Generic Positive Sequence Domain Model of Grid Forming Inverter Based Resource

3002021403

Generic Positive Sequence Domain Model of Grid Forming Inverter Based Resource

Technical Update, December 2021

EPRI Project Manager
E. Farantatos

ELECTRIC POWER RESEARCH INSTITUTE
3420 Hillview Avenue, Palo Alto, California 94304-1338 · PO BOX 10412, Palo Alto, California 94303-0813 · USA
800 313 3774 · 650 855 2121 · askepri@epri.com · www.epri.com

THE ELECTRIC POWER RESEARCH INSTITUTE (EPRI) PREPARED THIS REPORT.

This is an EPRI Technical Update report. A Technical Update report is intended as an informal report of continuing research, a meeting, or a topical study. It is not a final EPRI technical report.

NOTE

For further information about EPRI, call the EPRI Customer Assistance Center at 800 313 3774 or send an e-mail to askepri@epri.com.

ACKNOWLEDGMENTS

The Electric Power Research Institute (EPRI) prepared this report.

Principal Investigators:
D. Ramasubramanian
E. Farantatos

The concepts used in this model are based on research work carried jointly across EPRI, University of Washington (UW), University of Illinois Urbana Champaign (UIUC), and University of Minnesota (UMN) through a U.S. Department of Energy, Solar Energy Technologies Office project with Award Number DE-EE0009025 *A Scalable Control Architecture for 100% PV Penetration with Grid Forming Inverters*.

This report describes research sponsored by EPRI.

This publication is a corporate document that should be cited in the literature in the following manner:

Generic Positive Sequence Domain Model of Grid Forming Inverter Based Resource, EPRI, Palo Alto, CA: 2021. 3002021403

ABSTRACT

This document describes the performance of a generic grid forming (GFM) inverter-based resource (IBR) positive sequence model. The developed generic GFM model can represent four different types of GFM control methods that have been proposed in research literature. The behavior of the model has been compared against EMT domain simulations with satisfactory results.

Keywords

Grid forming, Inverter based resource, Positive Sequence

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1

INTRODUCTION

This document describes an inverter based resource (IBR) generic grid forming (GFM) positive sequence model developed at EPRI. The model is developed in positive sequence domain (in GE-PSLF^{TM}). The concepts used in this model are based on research work carried jointly across EPRI, University of Washington (UW), University of Illinois Urbana Champaign (UIUC), and University of Minnesota (UMN). Associated research results and further details are available in references [1–5].

The generic GFM model can represent, in a general way, four different types of GFM control methods that have been proposed in the literature. These methods are:

1. Droop based GFM
2. Virtual Synchronous Machine (VSM) based GFM
3. Dispatchable Virtual Oscillator (dVOC) based GFM
4. Phase Locked Loop (PLL) based GFM

An underlying structural similarity across the first three methods [1, 2] forms the basis for this generic model while an operational similarity with the fourth method [4,5] completes the model to allow it to cater to a wide variety of different IBR control representations. For the purpose of this document (and the corresponding simulation model), a grid following IBR is defined as a source that cannot maintain voltage *and* frequency within conventional limits (even though at a high level, the traces of active/reactive power and voltage may appear to be stable). A grid forming IBR, on the other hand, is defined as a source that can maintain both voltage and frequency within standard limits while also serving load and riding through faults in a robust manner [6].

Use of the positive sequence model associated with this document for system studies is to be accompanied with cautions, not limited to:

1. All models are still preliminary in nature and can continue to evolve over time to be made more robust with additional features.

2. The control structures and algorithms used for the grid forming IBR are by no means the only way to achieve a stable operation. There are numerous methods presently available and still more are continuously developed in research domain.

3. The use of these models for blackstart studies has not yet been tested.

4. Additional exhaustive testing, validation, and numerical robustness of the models have not yet been carried out.

5. Detailed comparison of the behavior of the models against original equipment behavior has not yet been carried out.

6. Validation of robustness of behavior for unbalanced events has not yet been extensively conducted.

7. The values of control gains presently used in the models may not be readily suitable or appropriate for,

   (a) Different ratings or operating conditions.
   (b) Different network topologies.
   (c) Different load dynamic characteristics.
   (d) Different number of varied source device characteristics

   However, for the above conditions, it can be possible to tune the control gains to bring about a satisfactory response.

2

MODEL SETUP AND RESULTS

The setup of the model in a positive sequence software platform will be described in this chapter along with bench marking results. The generic GFM model can be structured in a modular fashion like the existing WECC generic model suite as shown in Figure 2-1 and Figure 2-2. In these figures, variables in blue color indicate input variables from the network to the control structure, orange color indicates output variables from the control structure to the network, green color indicates variables that can pass between different models, purple color indicates input reference values, and red color indicates state variables. All other variables are either local variables or control gains/flag settings. The *xy* reference frame is the real – imaginary coordinate frame of the network while the *dq* reference frame is the coordinate frame of the control. The relative angle between these reference frames is denoted by the control variable θ_{inv}. Here in the figures, the model structure and interface with the network is shown in positive sequence/RMS frame. However, the same structure can be seamlessly applied in EMT domain, with *abc* quantities of voltage and current as inputs, which are subsequently transformed to the *dq* frame.

Due to the generalized nature of the model, not all control gains shown in the figures are direct inputs from the user. The inputs from the user for the independent parameters are tabulated in Table 2-1 while the values of the dependent parameters (i.e., the parameters derived from the user inputs based on the type of GFM control method chosen) are as tabulated in Table 2-2 with the value of K_2^{dvoc} derived as denoted in equation (2.1).

$$\frac{K_2^{dvoc}}{\omega_{drp}} = \frac{4 * 100^4}{100^4 - \left(2 * \left(100 - 100 * Q_{drp}\right)^2 - 100^2\right)^2} \tag{2.1}$$

Here, parameters ω_{drp} and Q_{drp} are the values of droop, however not expressed in percent. For example, an ω_{drp} of 5% will be represented as 0.05. In the model, all input reference values (purple colored variables) are specified in per unit on the MVA and kV rating of the IBR device. Further, input variables (blue colored variables) and output variables (orange colored variables) are also in per unit on the rating of the IBR device. It is expected that the software environment will handle the corresponding per unit conversions between the device and the network. Additionally in positive sequence environment, the network interface is modeled as a voltage source in a manner which is like the REGC_C model [7]. Due to the voltage source nature of the interface, there could also be

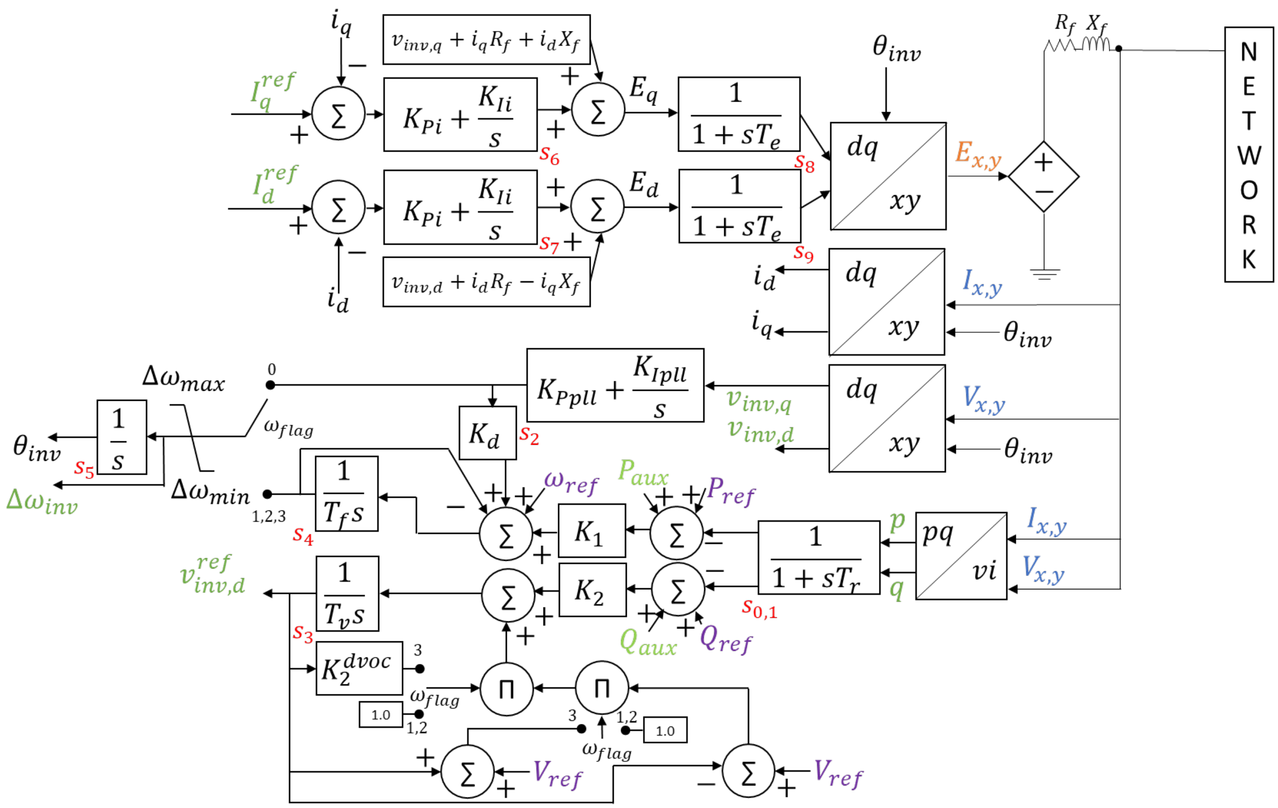

Figure 2-1
Generic grid forming (GFM) renewable generator/converter model

a need to ensure current limits are maintained during a fault in the positive sequence environment. This can require an algebraic iteration with the network solution at each time step during the fault. The method adopted in the REGC_C model and explained in detail in [7] is also adopted in this model.

The model also has the capability to receive signals P_{aux} and Q_{aux} from auxiliary control modules such as power oscillation dampers, automatic generation control blocks, plant controllers, etc. Additionally, when terminal voltage magnitude (i.e., $\sqrt{v^2_{(inv,d)} + v^2_{(inv,q)}}$) falls below the freeze threshold V_{dip}, states s_2, s_{10}, s_{11}, s_{12}, s_{13} are frozen.

Further, it can be noticed that the layout of the PLL based GFM mode option is like the existing REGC_C + REEC_D + REPC_A models. A detailed explanation of the use of the REGC_C + REEC_D + REPC_A models in grid forming mode (for non-blackstart scenarios) is provided in v1.0 of the EPRI GFM-PV software [8]. The code for the model developed in the positive sequence simulation software GE-PSLF is provided in Appendix 4 and the model can be represented in the dynamic data file as,

```
# GE-PSLF (TM) Parameterization of GFM IBR Model
epcgn1 1 "BUS 1 " 0.65 "1" : #9 mva=1060.000000 "generic_GFM.p" 6.0 "rsrc" 0.0015
    "xsrc" 0.15 "Cf" 0.0167 "Vdip" 0.8 "Imax" 1.2 "pqflag" 1 "control" 1 "m_f"
   0.15 "d_f" 30.0 "Tr" 0.002 "d_d" 0.1 "d_v" 22.22 "Kppll" 20.0 "Kipll" 700.0 "
```

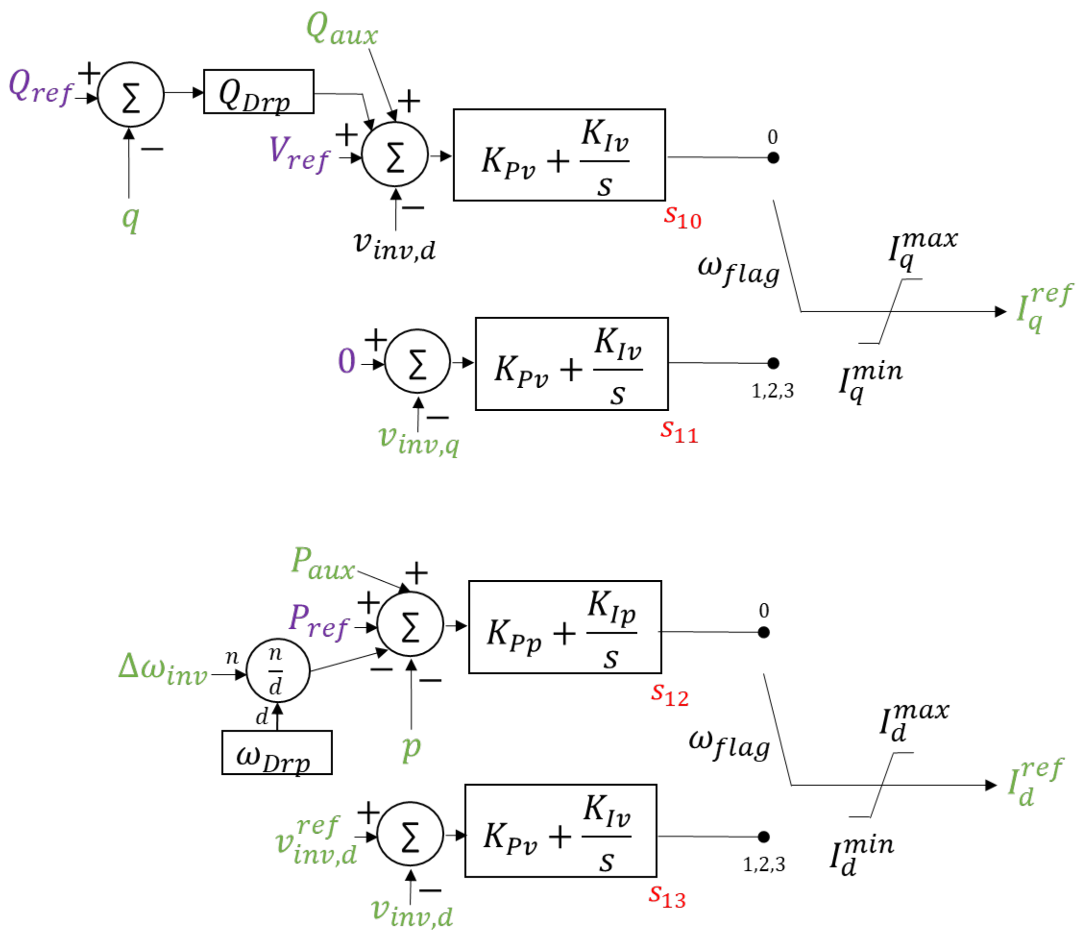

Figure 2-2
Generic grid forming (GFM) renewable electrical control model

```
K_Pv" 3.0 "K_Iv" 10.0 "K_Pi" 0.5 "K_Ii" 20.0 "K_Pp" 0.5 "K_Ip" 20.0 "K_Pvq"
0.5 "K_Ivq" 150.0 "Kpvq" 0.0 "Kivq" 0.0 "K_pod" 0.0 "Tw_pod" 0.01 "T1_pod"
0.01 "T2_pod" 0.005
```

Table 2-1
Input parameters entered by the user for the generic GFM model

Parameter	Description	Units	Default Value
MVA rating	IBR rating	MVA	100.0
R_f	Filter resistance	pu on MVA	0.0015
X_f	Filter reactance	pu on MVA	0.15
V_{dip}	State freeze threshold	pu	0.8
I_{max}	Maximum current magnitude	pu	1.2
PQflag	Current priority	-	0 - P priority 1 - Q priority
ω_{flag}	GFM control type	-	0 - PLL based 1 - Droop based 2 - VSM based 3 - dVOC based
$\Delta\omega_{max}$	Maximum value of frequency deviation	rad/s	75.0
$\Delta\omega_{min}$	Minimum value of frequency deviation	rad/s	-75.0
ω_{drp}	Frequency droop percent	-	0.033
Q_{drp}	Voltage droop percent	-	0.045
T_r	Transducer time constant	s	0.005
T_e	Output state time constant	s	0.01
m_f	VSM inertia constant	-	0.15
d_d	VSM damping factor	-	0.11
K_{Ppll}	PLL proportional gain	-	20.0
K_{Ipll}	PLL integral gain	-	700.0
K_{Pi}	Current controller proportional gain	-	0.5
K_{Ii}	Current controller integral gain	-	20.0
K_{Pv}	Voltage control proportional gain	-	0.5 (if $\omega_{flag} = 0$) 3.0 (if $\omega_{flag} \neq 0$)
K_{Iv}	Voltage control integral gain	-	150.0 (if $\omega_{flag} = 0$) 10.0 (if $\omega_{flag} \neq 0$)
K_{Pp}	Active power proportional gain	-	0.5
K_{Ip}	Active power integral gain	-	20.0

Table 2-2
Values of dependent parameters based on user input parameters for the generic GFM model

GFM Control	$\mathbf{T_f}$	$\mathbf{T_v}$	$\mathbf{K_d}$	$\mathbf{K_1}$	$\mathbf{K_2}$	$\mathbf{K_2^{dvoc}}$
PLL	N/A	N/A	0.0	N/A	N/A	N/A
Droop	0.0	0.0	0.0	ω_{drp}	Q_{drp}	N/A
VSM	$m_f * \omega_{drp}$	0.0	$d_d * \omega_{drp}$	ω_{drp}	Q_{drp}	N/A
dVOC	0.0	$1/\omega_0$	0.0	ω_{drp}/s_3^2	ω_{drp}/s_3	eqn (2.1)

Example results

The working of the generic GFM model is compared across both EMT model and positive sequence domain with simulations being carried out at a time step of $5\mu s$ in EMT domain and $1ms$ in positive sequence domain.

To showcase the model behavior, first a single inverter connected to an equivalent voltage source is considered. The setup of the network is as shown in Figure 2-3. At the start of the simulation the inverter is dispatched with P_{ref}=800MW and V_{ref}=1.035pu along with P_{load}=900MW and Q_{load}=210Mvar. Since the dispatch of the IBR is lower than the total load, the surplus power is provided by the equivalent voltage source. At t=5.0s the breaker connecting the equivalent source to the rest of the circuit is opened thereby creating a 100% inverter network. Following this at t=10.0s a solid to ground three phase fault is applied at the POI. A comparison of the response in EMT domain across all four grid forming modes of the model is shown in Figure 2-4 while a comparison of the response across all four grid forming modes in positive sequence domain is shown in Figure 2-5. From both figures the similarity of the responses can be observed. A one-to-one comparison of the behavior of the dVOC GFM mode across both EMT domain and positive sequence domain is shown in Figure 2-6. When the equivalent source is disconnected, initially there is a deficit in generation in the network as the IBR resource was dispatched at 800 MW. The deficit in generation both from active and reactive power results in voltage and frequency dropping. Subsequently, in all grid forming modes, frequency and voltage is controlled with an increase in active power and reactive power output. The response for a subsequent three phase solid to ground fault is also shown. It is seen that across all grid forming modes, both in EMT domain or positive sequence domain, the response of the generic model is similar and consistent with seamless translation of parameter values.

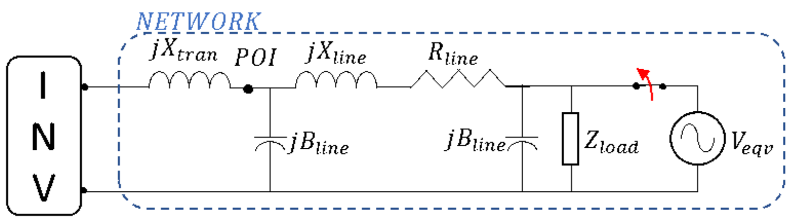

Figure 2-3
Single inverter-load-equivalent voltage source network to benchmark positive sequence and EMT model behavior

The take away from this model setup shows the behavior of the generic GFM model across difference control modes and simulation environments. This preliminary model can now be used as a starting point in many system studies.

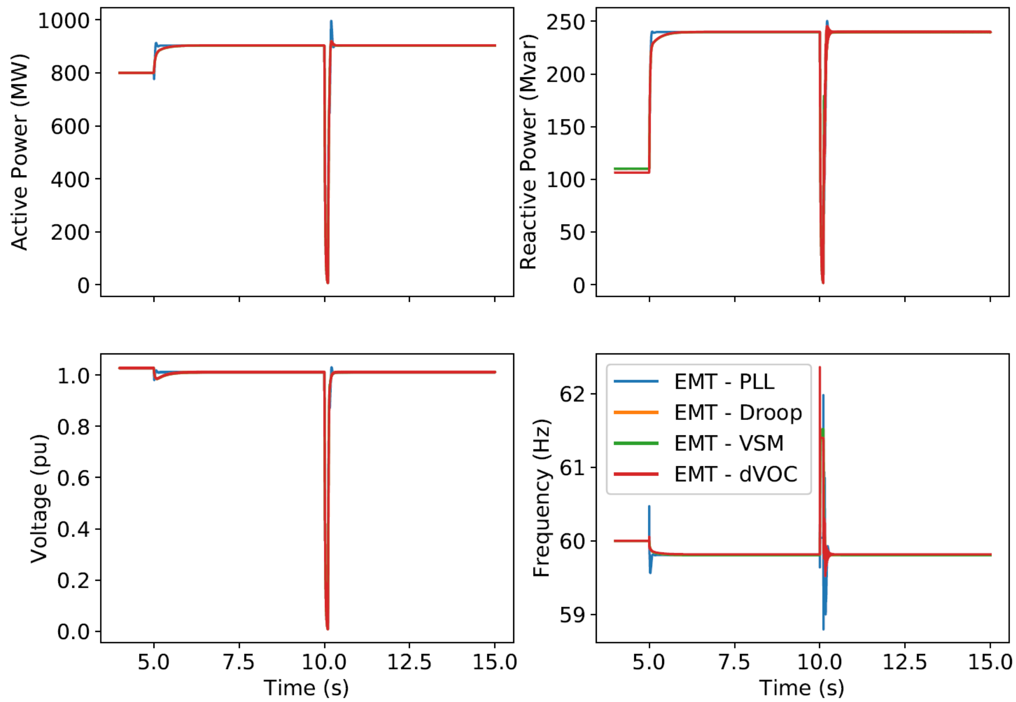

Figure 2-4
Comparison of EMT time domain response of the generic model in different GFM modes

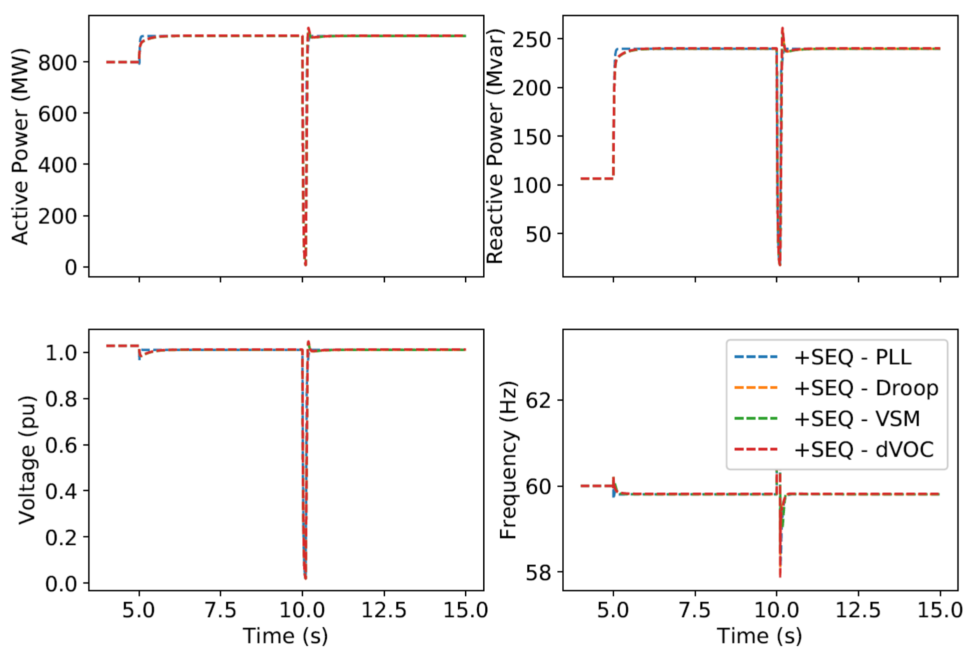

Figure 2-5
Comparison of positive sequence time domain response of the generic model in different GFM modes

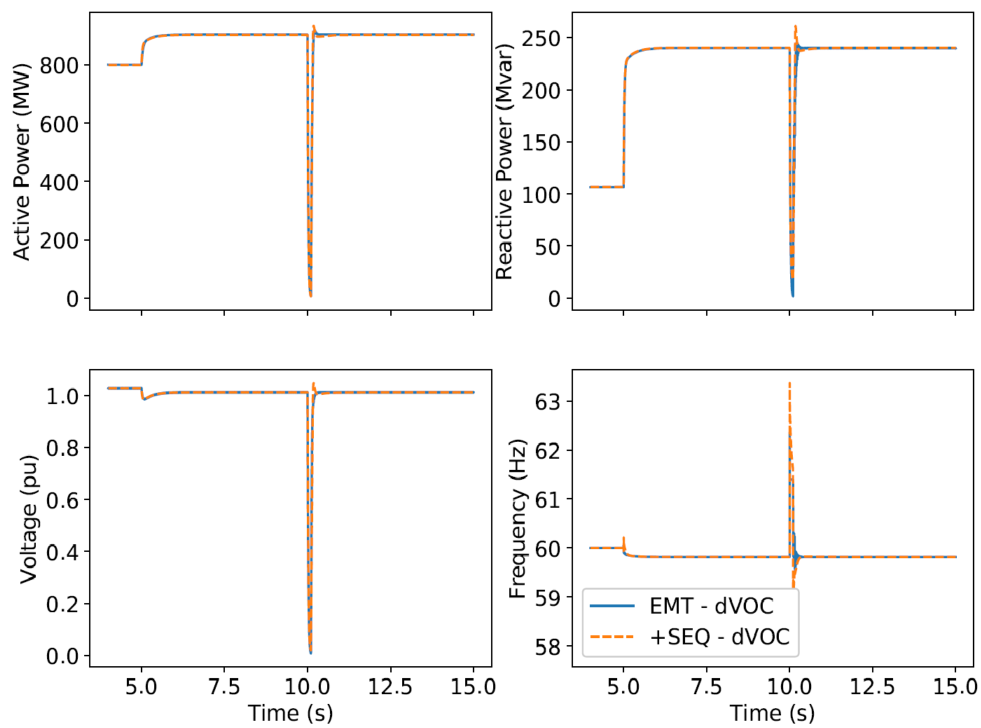

Figure 2-6
Comparison of EMT domain and positive sequence time domain response of the generic model in dVOC GFM mode

3

CONCLUSION

The models described in this document provide a starting point to study the behavior of grid forming inverter based resources in future power system. These models are not intended to be representative of exact equipment manufacturer controls and neither are they intended to be stable/operational in all system scenarios. As further research, testing, and validation is carried out, these models would continue to be updated.

4

GE-PSLF USER DEFINED MODEL CODE FOR GFM MODEL

The GE-PSLF code for this model is provided below.

```
/**************************************************/
/******* epcgn1 model for generic GFM ************/
/**************************************************/

/* Model comments and data description
Generic Positive sequence model of IBR plant in Grid Forming Mode.
Electric Power Research Institute (EPRI)
3420 Hillview Ave.
Palo Alto, CA 94304

Copyright © 2021 Electric Power Research Institute, Inc. All rights reserved.
This model depicts the converter as a controlled voltage source.
And also develops a generic GFM control structure for both PLL and non-PLL based

Model created at Electric Power Research Institute (EPRI)

As a user of this EPRI preproduction software, you accept and acknowledge that:
1: This software is a preproduction version which may have problems that could
   potentially harm your system
2: To satisfy the terms and conditions of the Master License Agreement or
   Preproduction License Agreement between EPRI and your company, you understand
   what to do with this preproduction product after the preproduction review
   period has expired
3: Reproduction or distribution of this preproduction software is in violation of
    the terms and conditions of the Master License Agreement or Preproduction
   License Agreement currently in place between EPRI and your company
4: Your company's funding will determine if you have the rights to the final
   production release of this product
```

```
5: EPRI will evaluate all tester suggestions and recommendations, but does not
   guarantee they will be incorporated into the final production product
6: As a preproduction tester, you agree to provide feedback as a condition of
   obtaining the preproduction software

Running this code, or running any simulation file that contains elements copied/
   transferred from this code, indicates acceptance and acknowledgement of the
   terms listed above.

Model developed by:
Deepak Ramasubramanian, EPRI
28 November 2021

Sample Input dynamic data record:

epcgn1 Busno. "Bus Name" kV "id" : #9 mva=100.0 "generic_GFM.p" 5.0 "rsrc" 0.0015
     "xsrc" 0.15 ...


***** End of comments */

define INIT 2
define SORC 3
define ALGE 4
define RATE 5
define OUTP 7
define NETW 8

@mx = dypar[0].cmi
@k = model[@mx].k
@bus=model[@mx].bus
@mode = dypar[0].mode
@kgen = genbc[@k].kgen

switch (@mode)

  case SORC:

        @E = sqrt((epcgn1[@mx].s11*epcgn1[@mx].s11)+(epcgn1[@mx].s12*epcgn1[@mx].
            s12))
        @Vdc = epcgn1[@mx].v10
        @VT = epcgn1[@mx].v11
        @wst = 2*(22/7)*60*dypar[0].time
        @m = @E/@VT

        if (@m>1.15)
          @m=1.0
```

```
       endif
       if (@m<0.2)
         @m=0.2
       endif

       if (epcgn1[@mx].s11>0.)
      @delt1=arctan(epcgn1[@mx].s12/epcgn1[@mx].s11)
        elseif (epcgn1[@mx].s11<0. and epcgn1[@mx].s12>=0.)
          @delt1=arctan(epcgn1[@mx].s12/epcgn1[@mx].s11)+(1.570796327*2)
        elseif (epcgn1[@mx].s11<0. and epcgn1[@mx].s12<0.)
          @delt1=arctan(epcgn1[@mx].s12/epcgn1[@mx].s11)-(1.570796327*2)
        elseif (epcgn1[@mx].s11=0. and epcgn1[@mx].s12>0.)
          @delt1=1.570796327
        elseif (epcgn1[@mx].s11=0. and epcgn1[@mx].s12<0.)
          @delt1=-1.570796327
        endif

       @Va = 0.5*@m*@Vdc*cos((@wst)+@delt1)
       @Vb = 0.5*@m*@Vdc*cos((@wst)+@delt1-2.094395102)
       @Vc = 0.5*@m*@Vdc*cos((@wst)+@delt1+2.094395102)

       @Ed = (2/3)*((@Va*cos(@wst))+(@Vb*cos(@wst-2.094395102))+(@Vc*cos(@wst
           +2.094395102)))
       @Eq = (-2/3)*((@Va*sin(@wst))+(@Vb*sin(@wst-2.094395102))+(@Vc*sin(@wst
           +2.094395102)))

       if (epcgn1[@mx].control = 0)
              epcgn1[@mx].angle = epcgn1[@mx].s3 + 1.570796327
       else
              epcgn1[@mx].angle = epcgn1[@mx].s6 + 1.570796327
       endif

       epcgn1[@mx].ed = @Ed
       epcgn1[@mx].eq = @Eq

break

case NETW:

      @delt = epcgn1[@mx].angle - 1.570796327

      @vq = (-volt[@bus].vr*sin(@delt))+(volt[@bus].vi*cos(@delt))
      @vd = (volt[@bus].vr*cos(@delt))+(volt[@bus].vi*sin(@delt))


  @g = epcgn1[@mx].rsrc/((epcgn1[@mx].rsrc*epcgn1[@mx].rsrc)+(epcgn1[@mx].xsrc*
      epcgn1[@mx].xsrc))
```

```
@b = epcgn1[@mx].xsrc/((epcgn1[@mx].rsrc*epcgn1[@mx].rsrc)+(epcgn1[@mx].xsrc*
    epcgn1[@mx].xsrc))

    @id = (((epcgn1[@mx].ed-@vd)*@g)+((epcgn1[@mx].eq-@vq)*@b))
@iq = (((epcgn1[@mx].eq-@vq)*@g)-((epcgn1[@mx].ed-@vd)*@b))

    @I = sqrt((@id*@id)+(@iq*@iq))

    if (@I > epcgn1[@mx].Imax)
      /* Setting current limits
     Adjusting priority based on flag setting*/
     @Imax = epcgn1[@mx].Imax
     @Ipmax_p = min(@Imax,epcgn1[@mx].v14*1.1)
     @Iqmax_q = min(@Imax,epcgn1[@mx].v14*1.1)
     if (epcgn1[@mx].pqflag = 0)
            /*------ P Priority --------*/
            if (@id > @Ipmax_p)
                    @id = @Ipmax_p
            endif
            if (@id < -@Ipmax_p)
                    @id = -@Ipmax_p
            endif

            @Iqmax_p = sqrt((@Imax*@Imax)-(@id*@id))
            if (@iq > @Iqmax_p)
                    @iq = @Iqmax_p
            endif
            if (@iq < -@Iqmax_p)
          @iq = -@Iqmax_p
            endif
     else
            /*------ Q Priority --------*/
       if (@iq > @Iqmax_q)
          @iq = @Iqmax_q
       endif
       if (@iq < -@Iqmax_q)
          @iq = -@Iqmax_q
       endif

       @Ipmax_q = sqrt((@Imax*@Imax)-(@iq*@iq));
       if (@id > @Ipmax_q)
         @id = @Ipmax_q
       endif
       if (@id < -@Ipmax_q)
         @id = -@Ipmax_q
       endif
     endif
```

```
 @ed = @vd+(@id*epcgn1[@mx].rsrc)-(@iq*epcgn1[@mx].xsrc)
 @eq = @vq+(@iq*epcgn1[@mx].rsrc)+(@id*epcgn1[@mx].xsrc)

 @E = sqrt((@ed*@ed)+(@eq*@eq))


 @Vdc = epcgn1[@mx].v10
 @VT = epcgn1[@mx].v11
 @wst = 2*(22/7)*60*dypar[0].time
 @m = @E/@VT

 if (@m>1.15)
   @m=1.15
 endif
 if (@m<0.2)
   @m=0.2
 endif

 if (@ed>0.)
@delt1=arctan(@eq/@ed)
  elseif (@ed<0. and @eq>=0.)
    @delt1=arctan(@eq/@ed)+(1.570796327*2)
  elseif (@ed<0. and @eq<0.)
    @delt1=arctan(@eq/@ed)-(1.570796327*2)
  elseif (@ed=0. and @eq>0.)
    @delt1=1.570796327
  elseif (@ed=0. and @eq<0.)
    @delt1=-1.570796327
  endif


 @Va = 0.5*@m*@Vdc*cos((@wst)+@delt1)
 @Vb = 0.5*@m*@Vdc*cos((@wst)+@delt1-2.094395102)
 @Vc = 0.5*@m*@Vdc*cos((@wst)+@delt1+2.094395102)

 @Ed = (2/3)*((@Va*cos(@wst))+(@Vb*cos(@wst-2.094395102))+(@Vc*cos(@wst
     +2.094395102)))
 @Eq = (-2/3)*((@Va*sin(@wst))+(@Vb*sin(@wst-2.094395102))+(@Vc*sin(@wst
     +2.094395102)))

 epcgn1[@mx].angle = @delt+1.570796327
 epcgn1[@mx].ed = @Ed
 epcgn1[@mx].eq = @Eq

endif
```

```
break

case ALGE:
     if (epcgn1[@mx].control = 0)
           genbc[@k].angle = epcgn1[@mx].s3 * 57.2958
     else
           genbc[@k].angle = epcgn1[@mx].s6 * 57.2958
     endif

break

case RATE:

      @mvabase = gens[@kgen].mbase
      @Vdip = epcgn1[@mx].Vdip
      @C_f = epcgn1[@mx].Cf
      @Imax = epcgn1[@mx].Imax

      @control = epcgn1[@mx].control
      @m_f = epcgn1[@mx].m_f
      @d_f = epcgn1[@mx].d_f
      @Tr = epcgn1[@mx].Tr
      @d_d = epcgn1[@mx].d_d
      @d_v = epcgn1[@mx].d_v

      @Kppll = epcgn1[@mx].Kppll
      @Kipll = epcgn1[@mx].Kipll
      @K_Pv = epcgn1[@mx].K_Pv
      @K_Iv = epcgn1[@mx].K_Iv
      @K_Pi = epcgn1[@mx].K_Pi
      @K_Ii = epcgn1[@mx].K_Ii
      @K_Pp = epcgn1[@mx].K_Pp
      @K_Ip = epcgn1[@mx].K_Ip
      @K_Pvq = epcgn1[@mx].K_Pvq
      @K_Ivq = epcgn1[@mx].K_Ivq
      @Kpvq = epcgn1[@mx].Kpvq
      @Kivq = epcgn1[@mx].Kivq
      @K_pod = epcgn1[@mx].K_pod
      @Tw_pod = epcgn1[@mx].Tw_pod
      @T1_pod = epcgn1[@mx].T1_pod
      @T2_pod = epcgn1[@mx].T2_pod

      @omega0 = 1.0
      @Vnom = epcgn1[@mx].v13

      @Pref = genbc[@k].pref/@mvabase
```

```
    @Qref = epcgn1[@mx].v12/@mvabase

@vmon = volt[@bus].vm

    if (@control = 0)
          /* SRF - PLL control */
          @tau_f = 0.002
          @K_v = 1.0/@d_v
    endif
    if (@control = 1)
          /* Droop control */
          @tau_f = 0.0
          @tau_v = 0.0
          @K_d = 0.0
          @K_f = 1.0/@d_f
          @K_v = 1.0/@d_v
    endif
    if (@control = 2)
          /* VSM control */
          @tau_f = @m_f/@d_f
          @tau_v = 0.0
          @K_d = @d_d/@d_f
          @K_f = 1.0/@d_f
          @K_v = 1.0/@d_v
    endif
    if (@control = 3)
          /* dVOC control */
          @tau_f = 0.0
          @tau_v = 1.0/377.0 /*@omega0*/
          @K_d = 0.0
          @K_1 = 1/@d_f
          @K_2 = @K_1/((pow(100,4) - pow((2*pow((100.0-(100.0/@d_v)),2) - pow
              (100,2)),2))/(4.0*pow(100,4)))
          @K_f = @omega0*@K_1
          @K_v = @K_1
    endif


    /* Measurement Transducer for Power*/
    @pout = (epcgn1[@mx].er*epcgn1[@mx].itr) + (epcgn1[@mx].ei*epcgn1[@mx].iti
        )
    @qout = (-epcgn1[@mx].er*epcgn1[@mx].iti) + (epcgn1[@mx].ei*epcgn1[@mx].
        itr)

    epcgn1[@mx].ds0 = (@pout-epcgn1[@mx].s0)/@Tr
    epcgn1[@mx].ds1 = (@qout-epcgn1[@mx].s1)/@Tr
```

```
/* PLL equations for SRF-PLL or VSM Control */
if (@control = 0)
      @vq_inv = (-epcgn1[@mx].er*sin(epcgn1[@mx].s3))+(epcgn1[@mx].ei*cos(
         epcgn1[@mx].s3))
      @vd_inv = (epcgn1[@mx].er*cos(epcgn1[@mx].s3))+(epcgn1[@mx].ei*sin(
         epcgn1[@mx].s3))
      @vq_PLL = @vq_inv
else
      @vq_inv = (-epcgn1[@mx].er*sin(epcgn1[@mx].s6))+(epcgn1[@mx].ei*cos(
         epcgn1[@mx].s6))
      @vd_inv = (epcgn1[@mx].er*cos(epcgn1[@mx].s6))+(epcgn1[@mx].ei*sin(
         epcgn1[@mx].s6))
      @vq_PLL = (-@vd_inv*sin(epcgn1[@mx].s3))+(@vq_inv*cos(epcgn1[@mx].s3
         ))
endif

if (@vmon < @Vdip)
      epcgn1[@mx].ds2 = 0.0
      @PLLdelt = epcgn1[@mx].s2 + @Kppll*@vq_PLL*sqrt(3)
else
      epcgn1[@mx].ds2 = @Kipll*@vq_PLL*sqrt(3)
      @PLLdelt = epcgn1[@mx].s2 + @Kppll*@vq_PLL*sqrt(3)
endif

@PLLmax = 12.0*2*2*1.570796327
if (@PLLdelt > @PLLmax)
      @PLLdelt = @PLLmax
endif
if (@PLLdelt < -@PLLmax)
      @PLLdelt = -@PLLmax
endif
epcgn1[@mx].ds3 = @PLLdelt

@freq_pll = ((2*2*1.570796327*60) + @PLLdelt)/(2*2*1.570796327)

/* Reactive power voltage control loop */
if (@control = 0 or @control = 1 or @control = 2)
      @Vd_ref = @K_v*(@Qref - epcgn1[@mx].s1) + @Vnom
      epcgn1[@mx].ds4 = 0.0
      @Vq_ref = 0.0
endif
if (@control = 3)
      epcgn1[@mx].ds4 = (1/@tau_v)*(@K_v*(@Qref - epcgn1[@mx].s1)/epcgn1[
         @mx].s4 + @K_2*(-(epcgn1[@mx].s4*epcgn1[@mx].s4)+(@Vnom*@Vnom))*
         epcgn1[@mx].s4)
      @Vd_ref = epcgn1[@mx].s4
      @Vq_ref = 0.0
```

```
endif

/* Active power frequency control loop */
if (@control = 0)
      @del_P = (@PLLdelt/(2*2*1.570796327*60.0))*@d_f
      epcgn1[@mx].ds5 = (1/@tau_f)*(-epcgn1[@mx].s5 + @Pref - @del_P)
      @w_ref = @freq_pll/60.0
endif
if (@control = 1)
      @w_ref = @omega0 + @K_f*(@Pref - epcgn1[@mx].s0)
      epcgn1[@mx].ds5 = 0.0
endif
if (@control = 2)
      epcgn1[@mx].ds5 = (1/@tau_f)*(-epcgn1[@mx].s5 + @omega0 + @K_f*(
          @Pref - epcgn1[@mx].s0) + @K_d*@PLLdelt)
      @w_ref = epcgn1[@mx].s5
endif
if (@control = 3)
      @w_ref = @omega0 + @K_f*(@Pref - epcgn1[@mx].s0)/(epcgn1[@mx].s4*
          epcgn1[@mx].s4)
      epcgn1[@mx].ds5 = 0.0
endif

/* Non PLL based GFM angle */
epcgn1[@mx].ds6 = (@w_ref - @omega0)*(2*2*1.570796327*60)

/* Voltage and power control loop */

if (@control = 0)
      @itiq = (-epcgn1[@mx].itr*sin(epcgn1[@mx].s3))+(epcgn1[@mx].iti*cos(
          epcgn1[@mx].s3))
      @itrd = (epcgn1[@mx].itr*cos(epcgn1[@mx].s3))+(epcgn1[@mx].iti*sin(
          epcgn1[@mx].s3))
else
      @itiq = (-epcgn1[@mx].itr*sin(epcgn1[@mx].s6))+(epcgn1[@mx].iti*cos(
          epcgn1[@mx].s6))
      @itrd = (epcgn1[@mx].itr*cos(epcgn1[@mx].s6))+(epcgn1[@mx].iti*sin(
          epcgn1[@mx].s6))
endif

if (@control = 0)
      if (@vmon < @Vdip)
             epcgn1[@mx].ds7 = 0.0
             epcgn1[@mx].ds8 = 0.0
      else
             epcgn1[@mx].ds7 = @K_Ip*(epcgn1[@mx].s5 - epcgn1[@mx].s0)
```

```
            epcgn1[@mx].ds8 = @K_Ivq*(@Vd_ref - sqrt((@vd_inv*@vd_inv)+(
                @vq_inv*@vq_inv)) + epcgn1[@mx].v9 + (@Kivq*@vq_inv))
      endif
      @Ipcmd = epcgn1[@mx].s7 + @K_Pp*(epcgn1[@mx].s5 - epcgn1[@mx].s0)
      @Iqcmd = (epcgn1[@mx].s8 + @K_Pvq*(@Vd_ref - sqrt((@vd_inv*@vd_inv)
          +(@vq_inv*@vq_inv)) + epcgn1[@mx].v9 + (@Kpvq*@vq_inv)))*-1.0
else
      if (@vmon < @Vdip)
            epcgn1[@mx].ds7 = 0.0
            epcgn1[@mx].ds8 = 0.0
      else
            epcgn1[@mx].ds7 = @K_Iv*(@Vd_ref - @vd_inv)
            epcgn1[@mx].ds8 = @K_Iv*(@Vq_ref - @vq_inv)
      endif

      @Vd_PI = epcgn1[@mx].s7 + @K_Pv*(@Vd_ref - @vd_inv)
      @Ipcmd = @Vd_PI - @C_f*@vq_inv

      @Vq_PI = epcgn1[@mx].s8 + @K_Pv*(@Vq_ref - @vq_inv)
      @Iqcmd = @Vq_PI + @C_f*@vd_inv
endif

/* Current limits */
@Ipmax_p = 1.0*@Imax
@Iqmax_q = 1.0*@Imax
if (epcgn1[@mx].pqflag = 0)
      /*------ P Priority --------*/
      if (@Ipcmd > @Ipmax_p)
            @Ipcmd = @Ipmax_p
      endif
      if (@Ipcmd < -@Ipmax_p)
            @Ipcmd = -@Ipmax_p
      endif

      @Iqmax_p = sqrt((@Imax*@Imax)-(@Ipcmd*@Ipcmd))
      if (@Iqcmd > @Iqmax_p)
            @Iqcmd = @Iqmax_p
      endif
      if (@Iqcmd < -@Iqmax_p)
    @Iqcmd = -@Iqmax_p
      endif

      epcgn1[@mx].v14 = abs(@Ipcmd)
else
      /*------ Q Priority --------*/
  if (@Iqcmd > @Iqmax_q)
    @Iqcmd = @Iqmax_q
```

```
  endif
  if (@Iqcmd < -@Iqmax_q)
     @Iqcmd = -@Iqmax_q
  endif

  @Ipmax_q = sqrt((@Imax*@Imax)-(@Iqcmd*@Iqcmd));
  if (@Ipcmd > @Ipmax_q)
    @Ipcmd = @Ipmax_q
  endif
  if (@Ipcmd < -@Ipmax_q)
    @Ipcmd = -@Ipmax_q
  endif

  epcgn1[@mx].v14 = abs(@Iqcmd)
endif

/* Current control loop */
epcgn1[@mx].ds9 = @K_Ii*(@Ipcmd - @itrd)
@Ip_PI = epcgn1[@mx].s9 + @K_Pi*(@Ipcmd - @itrd)

epcgn1[@mx].ds10 = @K_Ii*(@Iqcmd - @itiq)
@Iq_PI = epcgn1[@mx].s10 + @K_Pi*(@Iqcmd - @itiq)

if (@control = 0)
      @ed = @vd_inv + (@itrd*epcgn1[@mx].rsrc) - (@itiq*epcgn1[@mx].xsrc)
          - @Iq_PI
      @eq = @vq_inv + (@itiq*epcgn1[@mx].rsrc) + (@itrd*epcgn1[@mx].xsrc)
          + @Ip_PI
else
      @ed = @vd_inv + (@itrd*epcgn1[@mx].rsrc) - (@itiq*epcgn1[@mx].xsrc)
          + @Ip_PI
      @eq = @vq_inv + (@itiq*epcgn1[@mx].rsrc) + (@itrd*epcgn1[@mx].xsrc)
          + @Iq_PI
endif

epcgn1[@mx].ds11 = (@ed - epcgn1[@mx].s11)/0.002
epcgn1[@mx].ds12 = (@eq - epcgn1[@mx].s12)/0.002


/* Power oscillation damper control loop */
if (@control = 0)
      @Gain_pod = @K_pod*@Tw_pod
      epcgn1[@mx].ds13 = (@w_ref*(-@Gain_pod/@Tw_pod) - epcgn1[@mx].s13)/
          @Tw_pod
      @output_pod_1 = @w_ref*(@Gain_pod/@Tw_pod) + epcgn1[@mx].s13
```

```
            epcgn1[@mx].ds14 = (@output_pod_1*(1.0 - (@T1_pod/@T2_pod)) - epcgn1
                [@mx].s14)/@T2_pod
            @output_pod_2 = @output_pod_1*(@T1_pod/@T2_pod) + epcgn1[@mx].s14
      else
            epcgn1[@mx].ds13 = 0.0
            epcgn1[@mx].ds14 = 0.0
            @output_pod_2 = 0.0
      endif

      /*set variables for output file */
      epcgn1[@mx].v0 = @w_ref*60.0
      epcgn1[@mx].v1 = @Vd_ref
      epcgn1[@mx].v2 = @vd_inv
      epcgn1[@mx].v3 = @Vq_ref
      epcgn1[@mx].v4 = @vq_inv
      epcgn1[@mx].v5 = @Ipcmd
      epcgn1[@mx].v6 = @itrd
      epcgn1[@mx].v7 = @Iqcmd
      epcgn1[@mx].v8 = @itiq
      epcgn1[@mx].v9 = @output_pod_2

break

case INIT:

  @mvabase = gens[@kgen].mbase
     @control = epcgn1[@mx].control
     @C_f = epcgn1[@mx].Cf

     @K_pod = epcgn1[@mx].K_pod
     @Tw_pod = epcgn1[@mx].Tw_pod
     @T1_pod = epcgn1[@mx].T1_pod
     @T2_pod = epcgn1[@mx].T2_pod

     @pgen = gens[@kgen].pgen/@mvabase
     @qgen = gens[@kgen].qgen/@mvabase
     @vmon = volt[@bus].vm

     epcgn1[@mx].s0 = @pgen
     epcgn1[@mx].s1 = @qgen

     /* Calculating the inner voltage angle */

     if (epcgn1[@mx].er>0.)
   @delt=arctan(epcgn1[@mx].ei/epcgn1[@mx].er)
     elseif (epcgn1[@mx].er<0. and epcgn1[@mx].ei>=0.)
       @delt=arctan(epcgn1[@mx].ei/epcgn1[@mx].er)+(1.570796327*2)
```

```
elseif (epcgn1[@mx].er<0. and epcgn1[@mx].ei<0.)
  @delt=arctan(epcgn1[@mx].ei/epcgn1[@mx].er)-(1.570796327*2)
elseif (epcgn1[@mx].er=0. and epcgn1[@mx].ei>0.)
  @delt=1.570796327
elseif (epcgn1[@mx].er=0. and epcgn1[@mx].ei<0.)
  @delt=-1.570796327
endif

epcgn1[@mx].s2 = 0
if (@control = 0)
       epcgn1[@mx].s3 = @delt
else
       epcgn1[@mx].s3 = 0.0
endif

if (@control = 1 or @control = 2)
       epcgn1[@mx].s4 = 0.0
endif
if (@control = 3)
       epcgn1[@mx].s4 = @vmon
endif

if (@control = 0)
       epcgn1[@mx].s5 = @pgen
endif
if (@control = 1 or @control = 3)
       epcgn1[@mx].s5 = 0.0
endif
if (@control = 2)
       epcgn1[@mx].s5 = 1.0
endif

epcgn1[@mx].s6 = @delt

@vq=(-epcgn1[@mx].er*sin(@delt))+(epcgn1[@mx].ei*cos(@delt))
@vd=(epcgn1[@mx].er*cos(@delt))+(epcgn1[@mx].ei*sin(@delt))

@ipcmd = @pgen/@vmon
if (@control = 0)
       epcgn1[@mx].s7 = @ipcmd
else
       epcgn1[@mx].s7 = @ipcmd + @C_f*@vq
endif
epcgn1[@mx].s9 = 0.0

genbc[@k].pref = @pgen*@mvabase
```

```
@iqcmd = -@qgen/@vmon
if (@control = 0)
       epcgn1[@mx].s8 = -@iqcmd
else
       epcgn1[@mx].s8 = @iqcmd - @C_f*@vd
endif
epcgn1[@mx].s10 = 0.0

epcgn1[@mx].v12 = @qgen*@mvabase

epcgn1[@mx].v13 = @vmon

@ed=@vd+(@ipcmd*epcgn1[@mx].rsrc)-(@iqcmd*epcgn1[@mx].xsrc)
@eq=@vq+(@iqcmd*epcgn1[@mx].rsrc)+(@ipcmd*epcgn1[@mx].xsrc)

epcgn1[@mx].s11 = @ed
epcgn1[@mx].s12 = @eq

epcgn1[@mx].s13 = -1.0*@K_pod
epcgn1[@mx].s14 = 0.0

epcgn1[@mx].v14 = epcgn1[@mx].Imax


genbc[@k].speed = 1.0
genbc[@k].angle = @delt * 57.2958;


@E = sqrt((@ed*@ed)+(@eq*@eq))
@VT = @E/0.9
@Vdc = @E/(0.5*0.9)

epcgn1[@mx].v0 = 1.0
epcgn1[@mx].v1 = @vmon
epcgn1[@mx].v2 = @vd
epcgn1[@mx].v3 = 0.0
epcgn1[@mx].v4 = @vq
epcgn1[@mx].v5 = @ipcmd
epcgn1[@mx].v6 = @ipcmd
epcgn1[@mx].v7 = @iqcmd
epcgn1[@mx].v8 = @iqcmd
epcgn1[@mx].v9 = 0.0
epcgn1[@mx].v10 = @Vdc
epcgn1[@mx].v11 = @VT


epcgn1[@mx].ds0 = 0.0
```

```
    epcgn1[@mx].ds1 = 0.0
    epcgn1[@mx].ds2 = 0.0
    epcgn1[@mx].ds3 = 0.0
    epcgn1[@mx].ds4 = 0.0
    epcgn1[@mx].ds5 = 0.0
    epcgn1[@mx].ds6 = 0.0
    epcgn1[@mx].ds7 = 0.0
    epcgn1[@mx].ds8 = 0.0
    epcgn1[@mx].ds9 = 0.0
    epcgn1[@mx].ds10 = 0.0
    epcgn1[@mx].ds11 = 0.0
    epcgn1[@mx].ds12 = 0.0


    channel_head[0].type = "wref"
    channel_head[0].cmin = 0.
    channel_head[0].cmax = 150.0
    channel_head[1].type = "Vdcmd"
    channel_head[1].cmin = -50.0
    channel_head[1].cmax = 50.0
    channel_head[2].type = "vd"
    channel_head[2].cmin = 0.
    channel_head[2].cmax = 1.050
    channel_head[3].type = "Vqcmd"
    channel_head[3].cmin = 0.
    channel_head[3].cmax = 1.050
    channel_head[4].type = "vq"
    channel_head[4].cmin = 0.
    channel_head[4].cmax = 1.050
    channel_head[5].type = "Ipcmd"
    channel_head[5].cmin = 0.
    channel_head[5].cmax = 1.050
    channel_head[6].type = "ip"
    channel_head[6].cmin = 0.
    channel_head[6].cmax = 1.050
    channel_head[7].type = "Iqcmd"
    channel_head[7].cmin = 0.
    channel_head[7].cmax = 1.050
    channel_head[8].type = "iq"
    channel_head[8].cmin = 0.
    channel_head[8].cmax = 1.050
    channel_head[9].type = "pod"
    channel_head[9].cmin = 0.
    channel_head[9].cmax = 1.050


break
```

```
  case OUTP:

  break

endcase
end
```

5

REFERENCES

1. B. Johnson, T. Roberts, O. Ajala, A. D. Dominguez-Garcia, S. Dhople, D. Ramasubramanian, A. Tuohy, D. Divan, and B. Kroposki, "A generic primary-control model for grid-forming inverters: Towards interoperable operation & control," in *2022 55th Hawaii International Conference on System Sciences (HICSS)*, 2022.

2. O. Ajala, N. Baeckeland, S. Dhople, and A. D. Dominguez-Garcia, "Uncovering the kuramoto model from full-order models of grid-forming inverter-based power networks," in *IEEE Control and Decision Conference, 2021*, 2021.

3. D. Ramasubramanian, "Differentiating between plant level and inverter level voltage control to bring about operation of 100% inverter-based resource grids," *Electric Power Systems Research [under review]*.

4. D. Ramasubramanian and E. Farantatos, "Representation of grid forming virtual oscillator controller dynamics with wecc generic models," in *2021 IEEE PES General Meeting*, 2021.

5. D. Ramasubramanian, P. Pourbeik, E. Farantatos, and A. Gaikwad, "Simulation of 100% inverter-based resource grids with positive sequence modeling," *IEEE Electrification Magazine*, vol. 9, no. 2, pp. 62–71, 2021.

6. "Grid Forming Inverters: EPRI Tutorial (2021)," EPRI, Palo Alto, CA., Tech. Rep. 3002021722, 2021.

7. D. Ramasubramanian *et al.*, "Positive sequence voltage source converter mathematical model for use in low short circuit systems," *IET Generation, Transmission Distribution*, vol. 14, no. 1, pp. 87–97, 2020.

8. "EMT and Positive Sequence Domain Model of Grid Forming PV Plant (GFM-PV), Version 1.0 - Beta," EPRI, Palo Alto, CA., Tech. Rep. 3002021787, 2021.

The Electric Power Research Institute Inc., (EPRI, www.epri.com) conducts research and development relating to the generation, delivery and use of electricity for the benefit of the public. An independent, nonprofit organization, EPRI brings together its scientists and engineers as well as experts from academia and industry to help address challenges in electricity, including reliability, efficiency, affordability, health, safety and the environment. EPRI also provides technology, policy and economic analyses to drive long-range research and development planning, and supports research in emerging technologies. EPRI members represent 90% of the electricity generated and delivered in the United States with international participation extending to nearly 40 countries. EPRI's principal offices and laboratories are located in Palo Alto, Calif.; Charlotte, N.C.; Knoxville, Tenn.; Dallas, Texas; Lenox, Mass.; and Washington, D.C.

Together...Shaping the Future of Energy™

3002021403

Electric Power Research Institute
3420 Hillview Avenue, Palo Alto, California 94304-1338 · PO BOX 10412, Palo Alto, California 94303-0813 · USA
800 313 3774 · 650 855 2121 · askepri@epri.com · www.epri.com