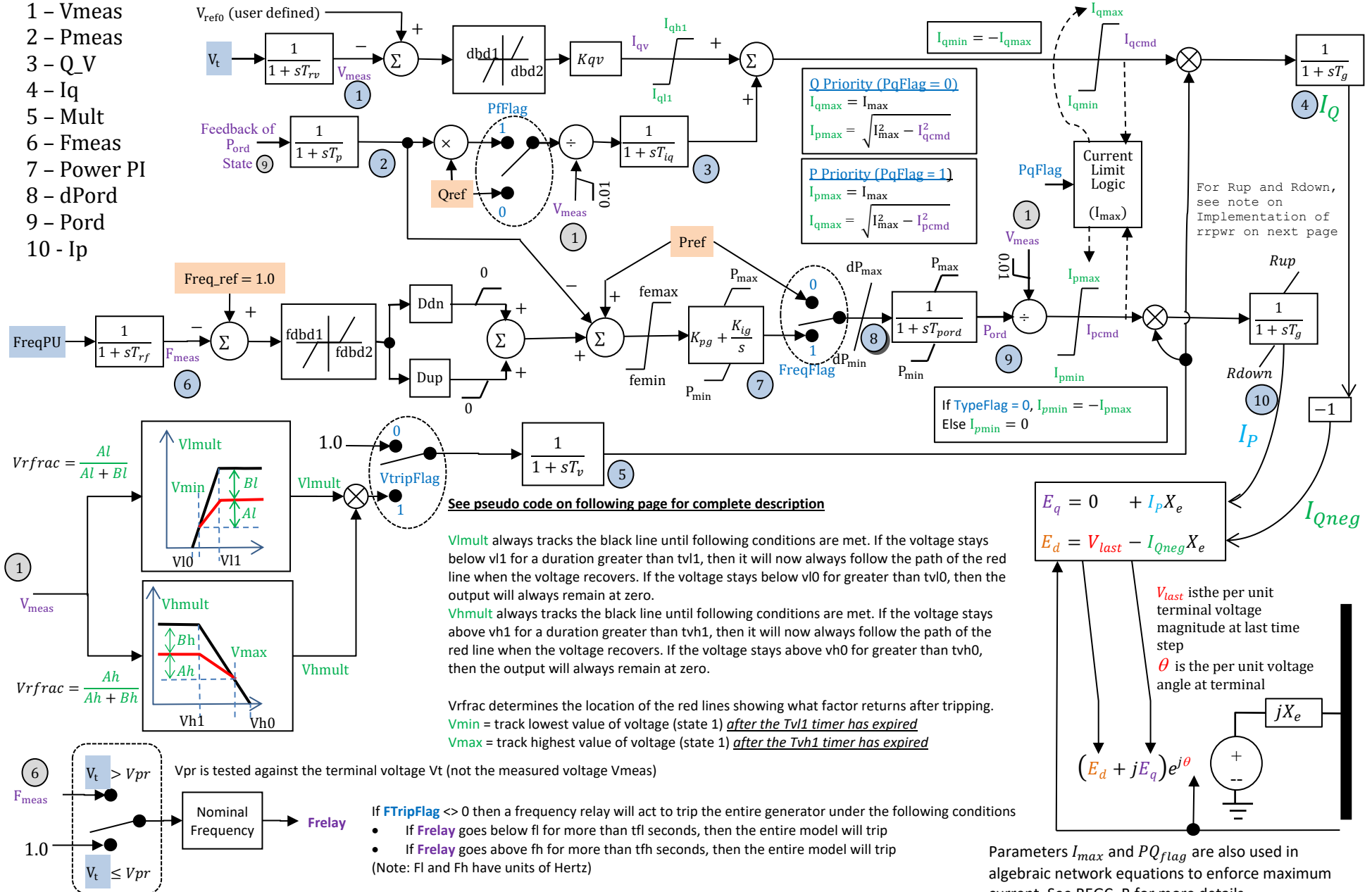


Machine Model DER_A

Model DER_A Distributed Energy Resource Aggregate model

States :

- 1 - Vmeas
- 2 - Pmeas
- 3 - Q_V
- 4 - Iq
- 5 - Mult
- 6 - Fmeas
- 7 - Power PI
- 8 - dPord
- 9 - Pord
- 10 - Ip



Pseudo Code for the Under and Over Voltage Tripping and Reconnection:

The following is performed each time step to model the low and high voltage tripping and reconnection.

Inputs Parameters

Vl0
Vl1
Vh0
Vh1
Tv10
Tv11
Tvh0
Tvh1

Simulation Inputs

theVolt
theTime

Results

Vlmult
Vhmult

Following
Variables are
internal to the
model

Numbers

Vmin
Vmax
Initialization Numbers
Vmin = Vl1
Vmax = Vh1

Timers

TimeBelowVl1
TimeAboveVh1
TimeBelowVl0
TimeAboveVh0

Boolean

(Initialize all Booleans to
FALSE)
ActiveTimerVl1
ActiveTimerVh1
ActiveFracLow
ActiveFracHigh
ActiveTimerVl0
ActiveTimerVh0
ActiveTripLow
ActiveTripHigh

// The variables come from the simulation
theVolt = voltage measurement (state 1)
theTime = present time of simulation

// Note: the pseudo code in this column is run
before the pseudo code on the right column

// Handle starting timers for low voltage

```
if (theVolt >= Vl1)
    ActiveTimerVl1 = false
elseif (not ActiveTimerVl1)
    ActiveTimerVl1 = true
    TimeBelowVl1 = theTime
endif
```

```
if (theVolt >= Vl0)
    ActiveTimerVl0 = false
elseif (not ActiveTimerVl0)
    ActiveTimerVl0 = true
    TimeBelowVl0 = theTime
endif
```

// Handle starting timers for high voltage

```
if (theVolt <= Vh1)
    ActiveTimerVh1 = false
elseif (not ActiveTimerVh1)
    ActiveTimerVh1 = true
    TimeAboveVh1 = theTime
endif
```

```
if (theVolt <= Vh0)
    ActiveTimerVh0 = false
elseif (not ActiveTimerVh0)
    ActiveTimerVh0 = true
    TimeAboveVh0 = theTime
endif
```

// Note: the pseudo code in the left column here is run before the pseudo code below

// Use Timers to see if on fractional restart curve

```
if (not ActiveFracLow) and ActiveTimerVl1 and ((theTime - TimeBelowVl1) >= Tv11)
    ActiveFracLow = true
endif
```

```
if (not ActiveFracHigh) and ActiveTimerVh1 and ((theTime - TimeAboveVh1) >= Tvh1)
    ActiveFracHigh = true
endif
```

// Use Timers to see if completely tripped

```
if (not ActiveTripLow) and ActiveTimerVl0 and ((theTime - TimeBelowVl0) >= Tv10)
    ActiveTripLow = true
endif
```

```
if (not ActiveTripHigh) and ActiveTimerVh0 and ((theTime - TimeAboveVh0) >= Tvh0)
    ActiveTripHigh = true
endif
```

// keep track of the minimum and the maximum voltage found after Fractional timers expire

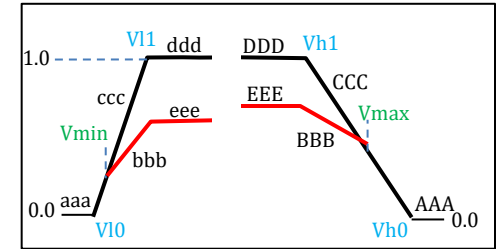
```
if (Vmin > theVolt) and ActiveFracLow
    Vmin = theVolt
    if Vmin < Vl0
        Vmin = Vl0
    endif
endif
if (Vmax < theVolt) and ActiveFracHigh
    Vmax = theVolt
    if Vmax > Vh0
        Vmax = Vh0
    endif
endif
```

// Calculate the Low Voltage multiplier

```
if (theVolt <= Vl0) or ActiveTripLow
    Vlmult = 0.0 // aaa
elseif (theVolt <= Vl1) and (theVolt > Vmin) and ActiveFracLow
    Vlmult = ((Vmin - Vl0) + Vrfrac * (theVolt - Vmin)) / (Vl1 - Vl0) // bbb
elseif (theVolt <= Vl1)
    Vlmult = (theVolt - Vl0) / (Vl1 - Vl0) // ccc
elseif (not ActiveFracLow)
    Vlmult = 1.0 // ddd
else
    Vlmult = ((Vmin - Vl0) + Vrfrac * (Vl1 - Vmin)) / (Vl1 - Vl0) // eee
endif
```

// Calculate the High voltage multiplier

```
if (theVolt >= Vh0) or ActiveTripHigh
    Vhmult = 0.0 // AAA
else if (theVolt >= Vh1) and (theVolt < Vmax) and ActiveFracHigh
    Vhmult = ((Vmax - Vh0) + Vrfrac * (theVolt - Vmax)) / (Vh1 - Vh0) // BBB
elseif (theVolt >= Vh1)
    Vhmult = (theVolt - Vh0) / (Vh1 - Vh0) // CCC
elseif (not ActiveFracHigh)
    Vhmult = 1.0 // DDD
else
    Vhmult = ((Vmax - Vh0) + Vrfrac * (Vh1 - Vmax)) / (Vh1 - Vh0) // EEE
endif
```



Calculation of Iqmax, Iqmin, Ipmax, and Ipmin

The values of Iqmax, Iqmin, Ipmax, and Ipmin are updated at the beginning of each time step. They are a function of the following.

1. Imax input parameter
2. PqFlag integer input parameter
3. TypeFlag integer input parameter
4. Ipcmd at last timestep
5. Iqcmd at last timestep

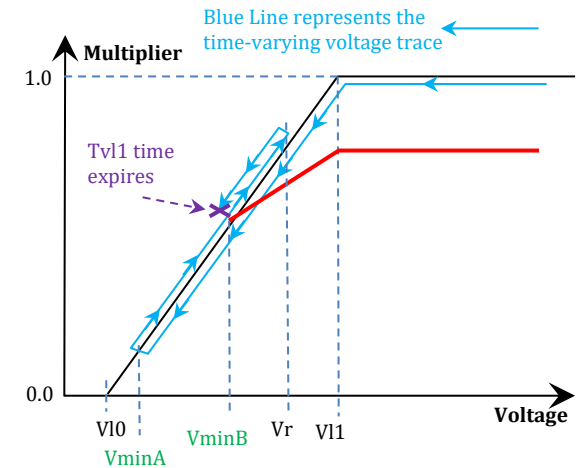
Pseudo-code for this logic is shown on the right.

```

if (PqFlag = 0)
    Iqmax = Imax
    Ipmax = sqrt(sqr(Imax) - sqr(IqCMD_LastTimeStep))
else
    Ipmax = Imax
    Iqmax = sqrt(sqr(Imax) - sqr(IpCMD_LastTimeStep))
endif
Iqmin = -Iqmax
if TypeFlag = 0
    Ipmin = -Ipmax
else
    Ipmin = 0
endif
    
```

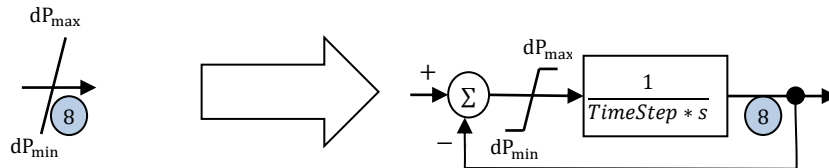
Clarification on the meaning of Vmin (and Vmax)

Vmin is initialized to V11. The value of Vmin will not then change until after the timer Tv11 expires. Vmin tracks the lowest value of voltage (state 1) after the Tv11 timer has expired. As an example, consider the blue line to the right representing a hypothetical time varying voltage tracing down the curve to VminA, then recovering up to Vr, then tracing back down to VminB at which time the timer Tv11 expires. The value of Vmin is thus set to VminB at the moment the timer expires. The voltage may have gone lower earlier in the simulation (such as down to VminA), but this does not matter. The value of Vmin is not tracked until after the timer expires.



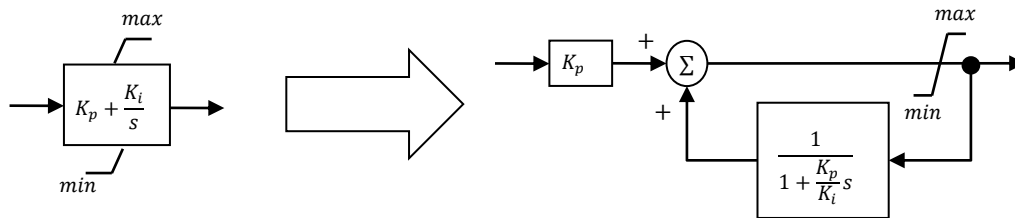
Implementation of the rate limit for dPmin and dPmax

The rate limit dPmin and dPmax are implemented using single time step integrator as follows.



Implementation of the PI controller with non-windup limits

When (Kpg <> 0) and (Kig <> 0), then the non-windup PI control is implemented using the following block diagram.



Implementation of the rrpwr Rate Limit

The parameter rrpwr represents an absolute value rate limit. It applies in the direction of the present sign of Ip. Thus it can be written as follows

If Ip >= 0 then Rup = +abs(rrpwr) Else Rup = +Infinity

If Ip <= 0 then Rdown = -abs(rrpwr) Else Rdown = -Infinity