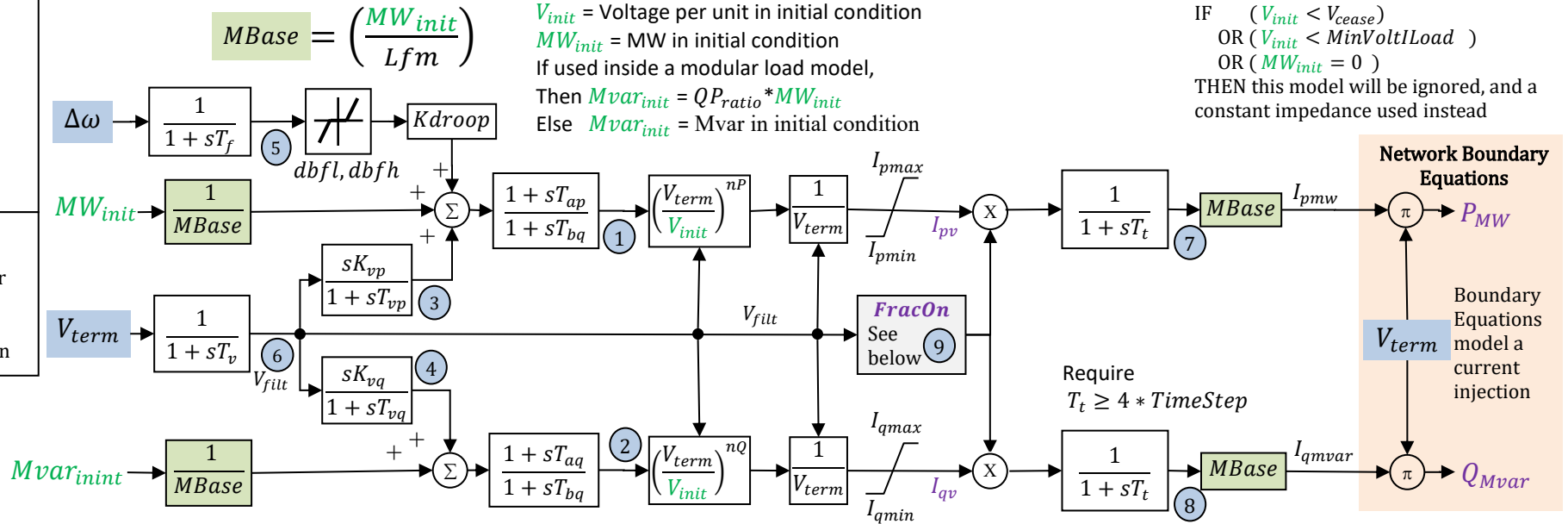


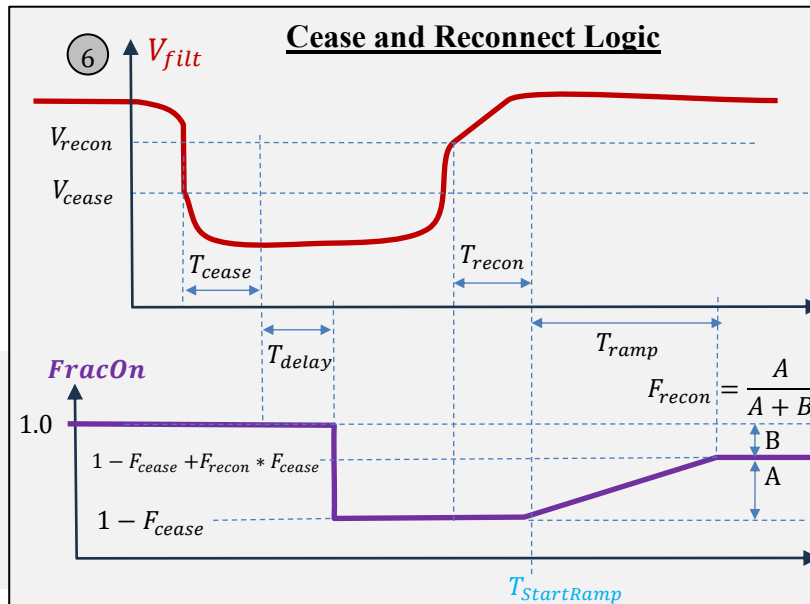
Load Characteristic PELCR2

Load Characteristic PELCR2 Power Electronic Load Model With Ceasing and Reconnecting

States
1 - Pctrl
2 - Qctrl
3 - dP
4 - dQ
5 - wfilt
6 - Vfilt
7 - Ip
8 - Iq
9 - Ramp
Others
1 - Pmw
2 - Qmvar
3 - Ipv
4 - Iqv
5 - FracOn



FracOn represents a fraction of the load that is operating based on the cease and reconnect logic.



Cease Logic

If $V_{filt} < V_{cease}$ for $T > T_{cease}$ seconds Then
 Start Delay timer.
 If Delay Timer $> T_{delay}$ then
 $FracOn = 1 - F_{cease}$

Reconnect Logic

If $V_{filt} > V_{recon}$ for $T > T_{recon}$ seconds
 Record reconnect time ($T_{StartRamp}$)

If $(T > T_{StartRamp} + T_{ramp})$ Then

$$FracOn = (1 - F_{cease}) + (F_{recon} F_{cease}) \quad (\text{and stop ramping})$$

Else

$$FracOn = (1 - F_{cease}) + (F_{recon} F_{cease}) \left(\frac{T - T_{StartRamp}}{T_{ramp}} \right)$$

Note: The actual logic is described in detail in pseudo-code below. It is possible for additional voltage dips to interrupt the reconnecting ramp.

Index	Parameter	Description
0	Lfm	Loading factor used to calculate MVABase of model as MWinit/Lfm. If Lfm < 0.001 then software will use a value of 1.000
1	QPratio	Q/P ratio for Q0 computation from P0 when used as a component of a CompLoad. When used as a stand-alone load model, Q0 = Qinit always.
2	Tfreq	Frequency measurement time constant [s]
3	Dbdl	Deadband on frequency response low (≤ 0) [pu]
4	Dbdh	Deadband on frequency response high (≥ 0) [pu]
5	Kdroop	Frequency droop [per unit]
6	Tvolt	Voltage measurement time constant [s]
7	Kvp	Proportional constant for active power washout
8	Tvp	Time constant for active power washout [seconds]
9	Kvq	Proportional constant for reactive power washout
10	Tvq	Time constant for reactive power washout [seconds]
11	Tap	Lead time constant for real power path [seconds]
12	Tbp	Lag time constant for real power path [seconds]
13	Taq	Lead time constant for reactive power path [seconds]
14	Tbq	Lag time constant for reactive power path [seconds]
15	nP	Active Power Exponential
16	nQ	Reactive Power Exponential
17	Fcease	Fraction that will cease (between 0 and 1)
18	Vcease	Voltage threshold for cease logic [per unit]
19	Tcease	Time delay for cease logic to be initiated for (≥ 0) [seconds]
20	Tdelay	Time delay to cease after detection for (≥ 0) [seconds]
21	Frecon	Of the Fraction that ceases, this is the fraction that then reconnects (between 0 and 1)
22	Vrecon	Voltage threshold to initiate power ramp reconnection logic [per unit]
23	Trecon	Time delay for ramp up logic to be initiated (≥ 0) [seconds]
24	Tramp	Ramp up time (≥ 0) [seconds]
25	Ipmax	Maximum Ip [per unit]
26	Ipmin	Minimum Ip [per unit]
27	Iqmax	Maximum Iq [per unit]
28	Iqmin	Minimum Iq [per unit]
29	Tt	Time delay for outputs [seconds] (If $< 4 \times \text{TimeStep}$ will be treated as $4 \times \text{TimeStep}$)

Pseudo code details for Cease and Reconnect Logic

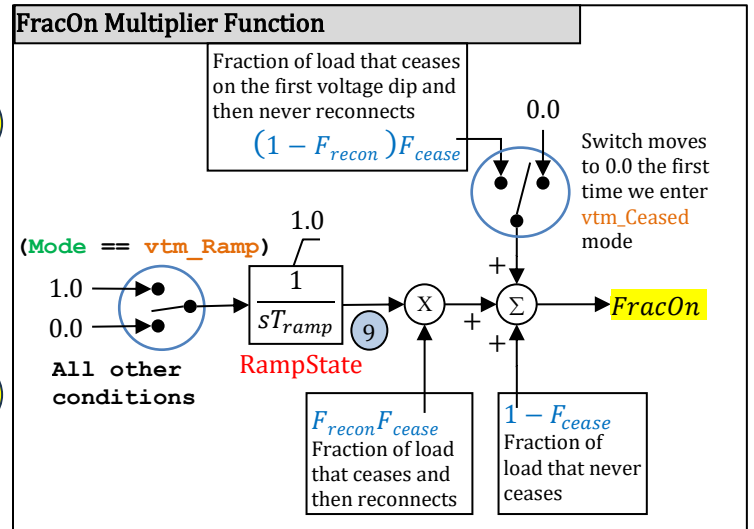
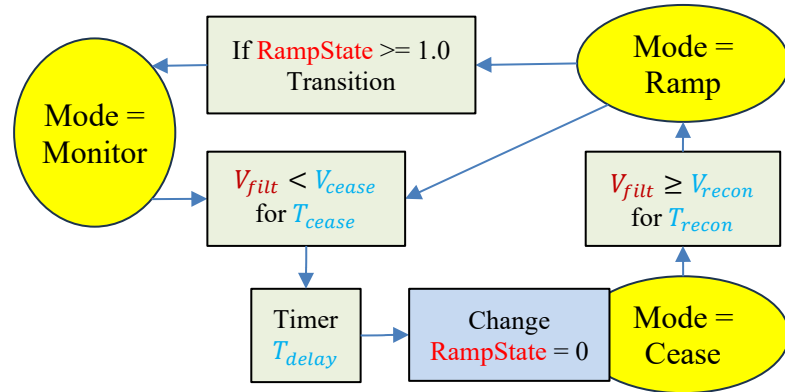
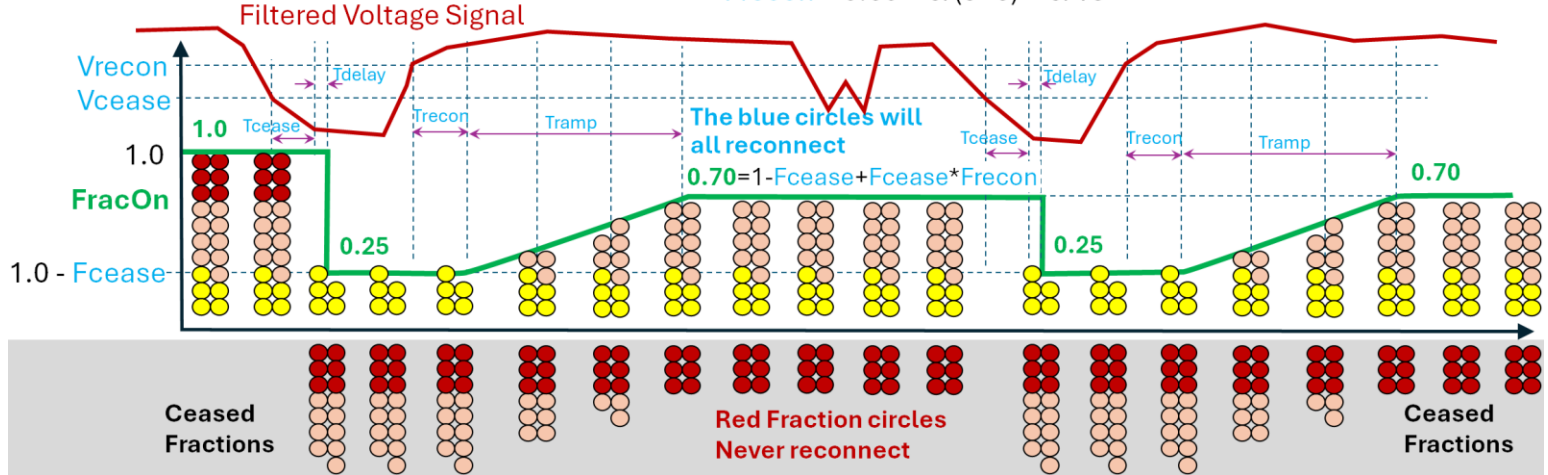
Cease and Reconnect logic uses a timer to control transitions between 3 modes : Monitor, Ceased and Ramp. While in the Ramp Mode, a dynamic integrator state is used to perform the ramp of reconnecting over a time of Tramp seconds. The output of the cease and reconnect logic is a multiplier **FracOn** the specifies what fraction of the respective portion of load remains online.

- 6 circles ● Ceases bus never reconnects ($F_{cease} * (1 - F_{recon})$)
- 9 circles ○ Ceases but reconnects ($F_{cease} * F_{recon}$)
- 5 circles ● Never cease ($1 - F_{cease}$)

FracOn Example

$$F_{cease} = 0.75 = (9+6)/(9+6+5) = 15/20$$

$$F_{recon} = 0.60 = 9/(9+6) = 9/15$$



Variable Names in the Pseudo-Code Below

Fcease, **Vcease**, **Tcease**, **Tdelay** : input parameters control Ceasing
Frecon, **Vrecon**, **Trecon**, **Tramp** : input parameters control Reconnecting
Vfilt : State variable of the filtered voltage
RampState : State variable varying between 0 and 1 to control reconnection ramping
FracOn : Multiplier representing the fraction of the load that remains connected
Mode : 3 possible modes: **vtm_Monitor**, **vtm_Ceased**, **vtm_Ramp**
TimerActive : 4 possibilities determine if any timer is active (only one time can be active)
vtt_None, **vtt_Tcease**, **vtt_Tdelay**, **vtt_Trecon**
Timer : Float number used to keep track of timing

Initialize the following before simulation starts

```
RampState = 1  
Mode = vtm_Monitor  
TimerActive = vtt_None  
Timer = 0
```

Run the following at each time step

```
If Fcease > 0 Then // Ignore all this stuff if Fcease <= 0  
  If Mode == vtm_Ramp Then // update the value of FracOn  
    If RampState > 1 Then RampState = 1 // don't let it go above this value  
    FracOn = 1 - Fcease + Frecon*Fcease*RampState  
  EndIf  
  
  // Tcease and Tdelay timers are counting up the same whether we are in Monitor or Ramp  
  If Mode <> vtm_Ceased Then // handled in both Monitor and Ramp modes  
    If TimerActive <> vtt_Tdelay Then  
      If (Vfilt >= Vcease) Then  
        TimerActive = vtt_None  
      ElseIf (TimerActive <> vtt_Tcease) Then  
        Timer = PresentTime  
        TimerActive = vtt_Tcease  
      EndIf  
    EndIf  
    If (TimerActive = vtt_Tcease) AND (PresentTime - Timer) >= Tcease Then  
      Timer = PresentTime  
      TimerActive = vtt_Tdelay  
    EndIf  
    If (TimerActive = vtt_Tdelay) AND (PresentTime - Timer) >= Tdelay Then  
      TimerActive = vtt_None  
      Mode = vtm_Cease  
      RampState = 0  
      FracOn = 1 - Fcease  
    EndIf  
  EndIf  
  
  If (Mode == vtm_Ramp) Then  
    If RampState >= 1 Then  
      RampState = 1  
      FracOn = 1 - Fcease + Frecon*Fcease  
      Mode = vtm_Monitor  
    End  
  ElseIf (Mode == vtm_Cease) Then  
    If (Vfilt < Vrecon) Then  
      TimerActive = vtt_None  
    ElseIf (TimerActive <> vtt_Trecon)  
      Timer = PresentTime  
      TimerActive = vtt_Trecon  
    EndIf  
    If (TimerActive = vtt_Trecon) AND (PresentTime - Timer) >= Trecon Then  
      TimerActive = vtt_None  
      Mode = vtm_Ramp  
    EndIf  
  EndIf  
EndIf // If Fcease > 0
```