

Load Characteristic PERC2

Load Characteristic PERC2 Power Electronic Load Model With Reconnecting and Ceasing

States

- 1 - Pctrl
- 2 - Qctrl
- 3 - dP
- 4 - dQ
- 5 - wfilt
- 6 - Vfilt
- 7 - Ip
- 8 - Iq
- 9 - Pcmd
- 10 - Qcmd

Others

- 1 - Pmw
- 2 - Qmvar
- 3 - Ipv
- 4 - Iqv
- 5 - FracOn

$$MBase = \left(\frac{MW_{init}}{Lfm} \right)$$

V_{init} = Voltage per unit in initial condition

MW_{init} = MW in initial condition

$Mvar_{init} = Q_{P_{ratio}} * MW_{init}$

If used inside a modular load model, then $I_{qvextra} = 0$ and mismatch of Mvar is handled during the initialization of the ComLoad components.

If used as a stand-alone model, then $I_{qvextra} = \frac{Q_{Mvar} - Mvar_{init}}{MBase * V_{init}}$

$$I_{pmaxused} = \min(I_{pmax}, +I_{max})$$

$$I_{pminused} = \max(I_{pmin}, -I_{max})$$

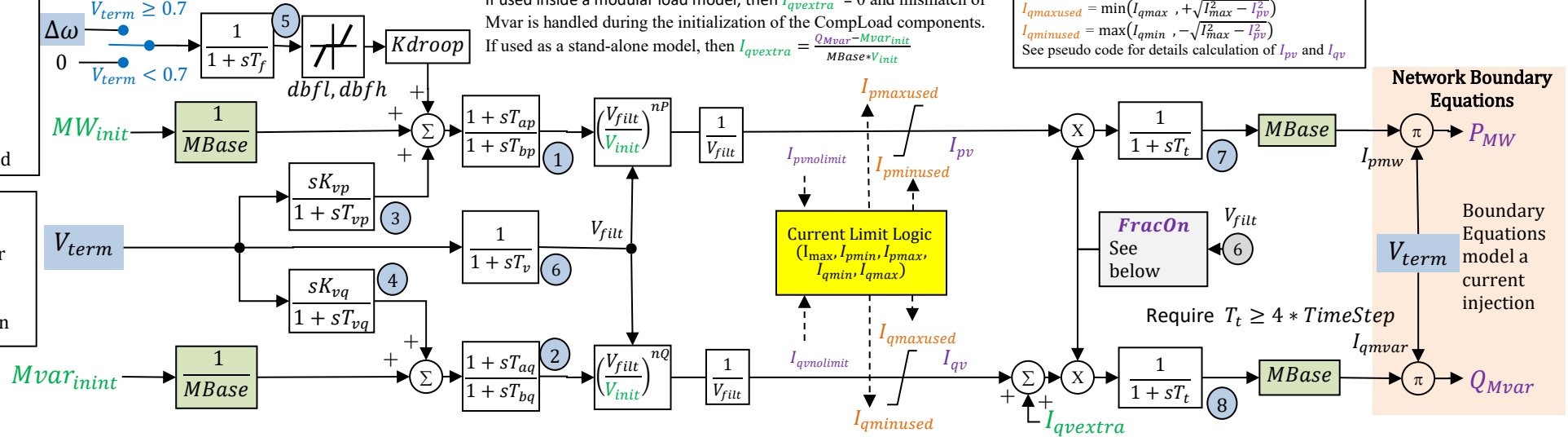
Enforce Ip limits first to get I_{pv} from $I_{pvlimit}$

$$I_{qmaxused} = \min(I_{qmax}, +\sqrt{I_{max}^2 - I_{pv}^2})$$

$$I_{qminused} = \max(I_{qmin}, -\sqrt{I_{max}^2 - I_{pv}^2})$$

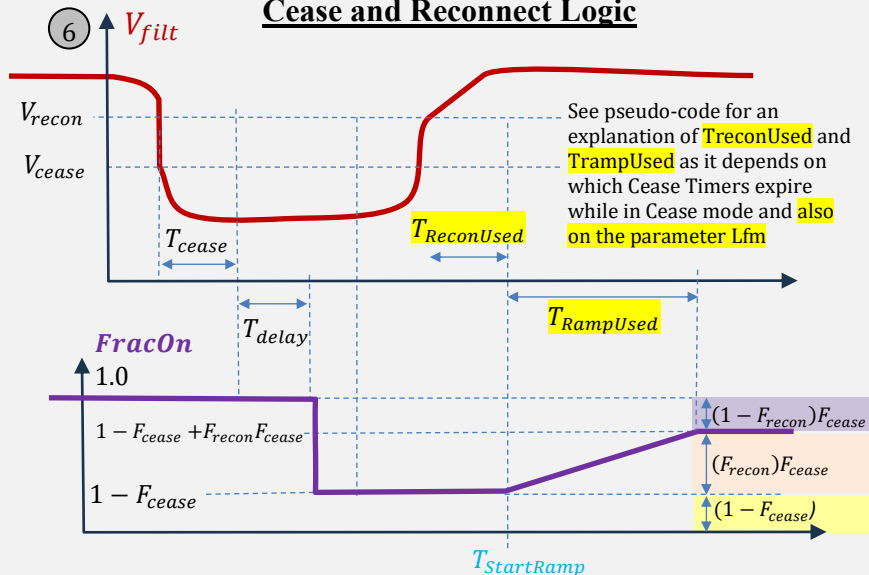
See pseudo code for details calculation of I_{pv} and I_{qv}

IF ($V_{init} < V_{cease}$)
OR ($V_{init} < MinVoltLoad$)
OR ($MW_{init} = 0$)
THEN this model will be ignored, and a constant impedance used instead



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

Cease and Reconnect Logic



Cease Logic

Actual logic is much more complex and involves various threshold/timer blocks

If ($V_{filt} < V_{cease}$) for $T > T_{cease}$

Then Start Delay timer.

If Delay Timer $> T_{delay}$

Then **FracOn** = $1 - F_{cease}$

Reconnect Logic

If ($V_{filt} > V_{Lrecon}$) AND ($V_{filt} < V_{Hrecon}$) for $T > T_{reconUsed}$ Then

Record reconnect time ($T_{startRamp}$) and start ramp

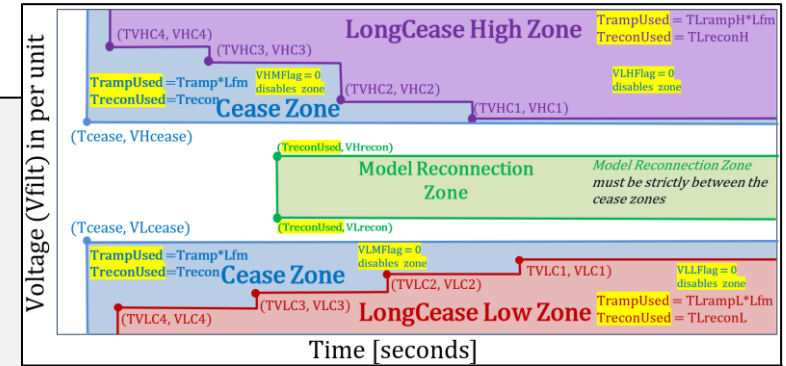
If ($T > T_{startRamp} + T_{RampUsed}$) Then

FracOn = $(1 - F_{cease}) + (F_{recon} F_{cease})$ (and stop ramp)

Else

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

Note: The actual logic is described in detail in pseudo-code below. It is possible for additional voltage dips to interrupt the reconnecting ramp. The threshold/timer which causing ceasing, determines the $TrampUsed$ and $TreconUsed$ values.

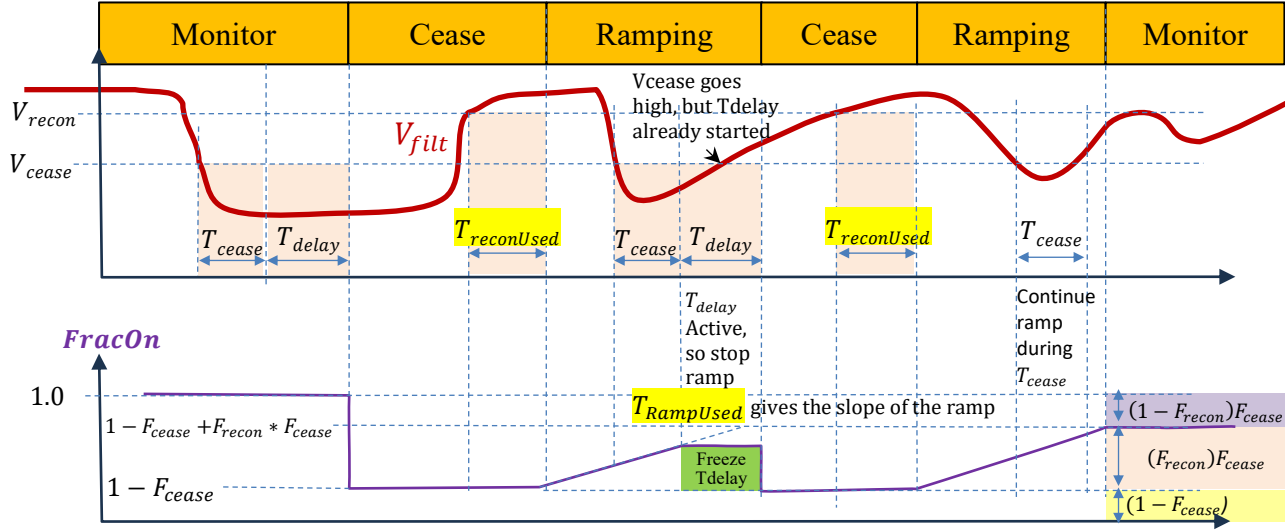


See pseudo code for details calculation of coordination of the various cease zones and the calculation of $TrampUsed$ and $TreconUsed$ from these zones

Index	Parameter	Description	Default
0	VLMflag	Low Voltage cease check (0 to not do the check; Else do the check)	1
1	VHMflag	High Voltage cease check (0 to not do the check; Else do the check)	1
2	VLLflag	Long Low Voltage cease check (0 to not do the check; Else do the check)	1
3	VLHflag	Long High Voltage cease check (0 to not do the check; Else do the check)	1
4	FLflag	Low Frequency cease check (0 to not do the check; Else do the check)	1
5	FHflag	High Frequency cease check (0 to not do the check; Else do the check)	1
6	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	0.80
7	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.	0.66
8	Tt	Time delay for outputs [seconds] (If < 4*TimeStep will be treated as 4*TimeStep)	0.02
9	Tv	Voltage measurement time constant [s]	0.02
10	Tf	Frequency measurement time constant [s]	0.02
11	Dbd1	Deadband on frequency response low (<=0) [pu]	0.00
12	Dbdh	Deadband on frequency response high (>=0) [pu]	0.00
13	Kdroop	Frequency droop [per unit]	10.00
14	Kvp	Proportional constant for active power washout	0.02
15	Tvp	Time constant for active power washout [seconds]	0.02
16	Kvq	Proportional constant for reactive power washout	0.50
17	Tvq	Time constant for reactive power washout [seconds]	0.10
18	Tap	Lead time constant for real power path [seconds]	0.00
19	Tbp	Lag time constant for real power path [seconds]	0.00
20	Taq	Lead time constant for reactive power path [seconds]	0.00
21	Tbq	Lag time constant for reactive power path [seconds]	0.00
22	nP	Active Power Exponential	0.00
23	nQ	Reactive Power Exponential	1.00
24	Ipmax	Maximum Ip [per unit]	1.00
25	Ipmin	Minimum Ip [per unit]	0.00
26	Iqmax	Maximum Iq [per unit]	0.66
27	Iqmin	Minimum Iq [per unit]	-0.66
28	Imax	Maximum total current [per unit]	2.00
29	Fcease	Fraction that will cease (between 0 and 1)	1.00
30	Frecon	Of the fraction that ceases, this is the fraction that then reconnects (>=0). Frecon > 1.0 allowed for a load that comes back above the initial value after reconnecting.	1.00
31	Tdelay	Time delay to cease after detection. This is the same for delay for Level Zones A, B and C (>= 0) [seconds]	0.00
32	VLrecon	Low Voltage threshold to initiate power ramp reconnection logic [per unit]	0.80
33	VHrecon	High Voltage threshold to initiate power ramp reconnection logic [per unit]	2.00
34	Tramp	Cease logic Ramp up time for portion of load that reconnects to go from 0 to 100% at full Mbase. (>= 0) [seconds]. TrampUsed = Tramp*Lfm	0.15
35	Trecon	Reconnection time delay for ramp up logic to be initiated for cease logic [per unit]	0.05
36	VLcease	Low Voltage threshold for the Cease logic [per unit]	0.80
37	VHcease	High Voltage threshold for the Cease logic [per unit]	2.00
38	Tcease	Low Time delay for Cease logic to be initiated (>= 0) [seconds]	0.01
39	TLrampL	Long Low Cease logic Ramp up time. (>= 0) [seconds]. TrampUsed = TrampLL*Lfm	5.00
40	TLreconL	Long Low Cease Time delay for ramp up logic to be initiated (>= 0) [seconds]	0.05
41	VLC1	First Low Voltage threshold for Long Cease [per unit]	0.90
42	TVLC1	First Low Time delay for Long Cease to be initiated (>= 0) [seconds]	2.00
43	VLC2	Second Low Voltage threshold for Long Cease [per unit]	0.80
44	TVLC2	Second Low Time delay for Long Cease to be initiated (>= 0) [seconds]	0.50
45	VLC3	Third Low Voltage threshold for Long Cease [per unit]	0.50
46	TVLC3	Third Low Time delay for Long Cease to be initiated (>= 0) [seconds]	0.25
47	VLC4	Fourth Low Voltage threshold for Long Cease [per unit]	0.35
48	TVLC4	Fourth Low Time delay for Long Cease to be initiated (>= 0) [seconds]	0.15
49	TLrampH	Long High Cease logic Ramp up time. (>= 0) [seconds]. TrampUsed = TrampLL*Lfm	5.00
50	TLreconH	Long High Cease Time delay for ramp up logic to be initiated (>= 0) [seconds]	0.05
51	VHC1	First High Voltage threshold for Long Cease [per unit]	1.10
52	TVHC1	First High Time delay for Long Cease to be initiated (>= 0) [seconds]	1.00
53	VHC2	Second High Voltage threshold for Long Cease [per unit]	2.00
54	TVHC2	Second High Time delay for Long Cease to be initiated (>= 0) [seconds]	99.00
55	VHC3	Third High Voltage threshold for Long Cease [per unit]	2.00
56	TVHC3	Third High Time delay for Long Cease to be initiated (>= 0) [seconds]	99.00
57	VHC4	Fourth High Voltage threshold for Long Cease [per unit]	2.00
58	TVHC4	Fourth High Time delay for Long Cease to be initiated (>= 0) [seconds]	99.00
59	FLC1	Low Frequency Threshold for Forever Cease [Hz]	57.00
60	TFLC1	Low Frequency Time Delay for Forever Cease to be initiated (>= 0) [seconds]	999.00
61	FHC1	High Frequency Threshold for Forever Cease [Hz]	63.00
62	TFHC1	High Frequency Time Delay for Forever Cease to be initiated (>= 0) [seconds]	999.00

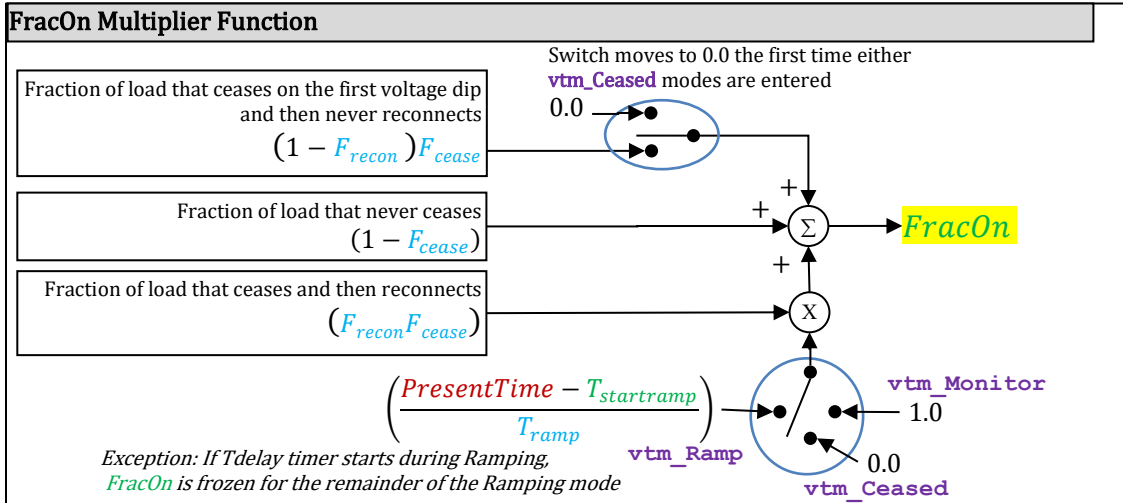
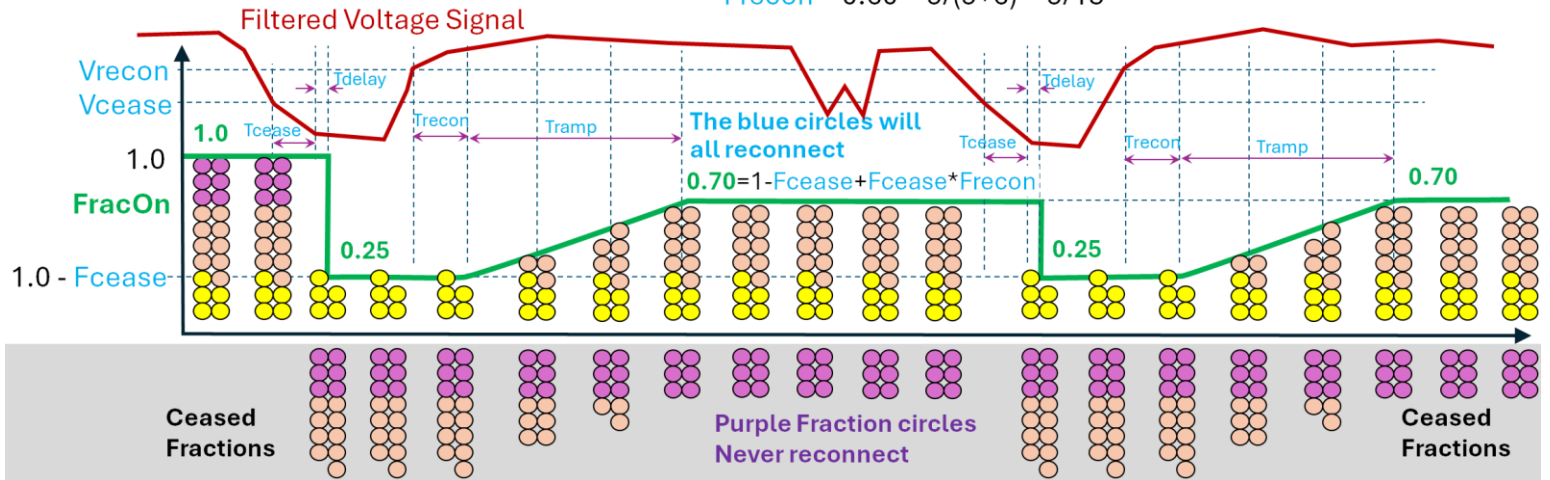
Pseudo code details for Cease and Reconnect Logic

Cease and Reconnect logic uses timers to control transitions between 3 modes: **Monitor**, **Ceased**, and **Ramp**. While in the Ramp Mode, the **FracOn** variable is updated each timestep to perform a ramp from the Ceased Mode value of $(1 - F_{cease})$ back up to $(1 - F_{cease} + F_{recon} * F_{cease})$. 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
 $F_{cease} = 0.75 = (9+6)/(9+6+5) = 15/20$
 $F_{recon} = 0.60 = 9/(9+6) = 9/15$



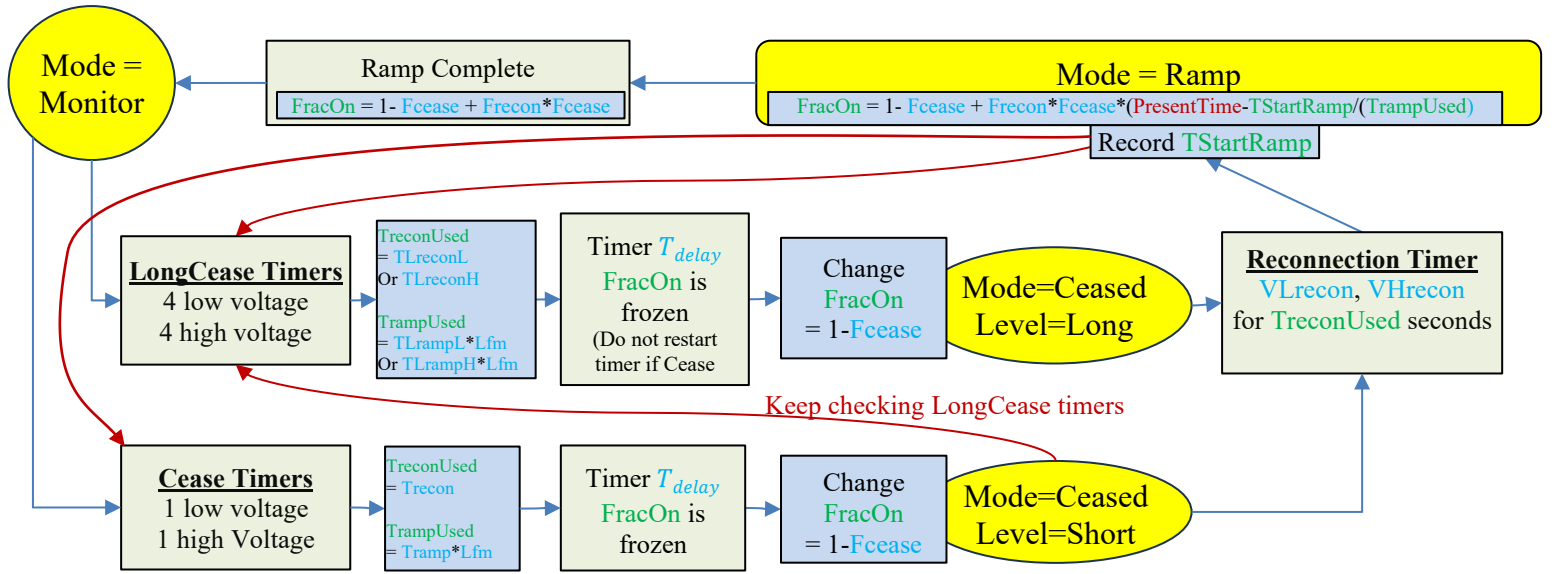
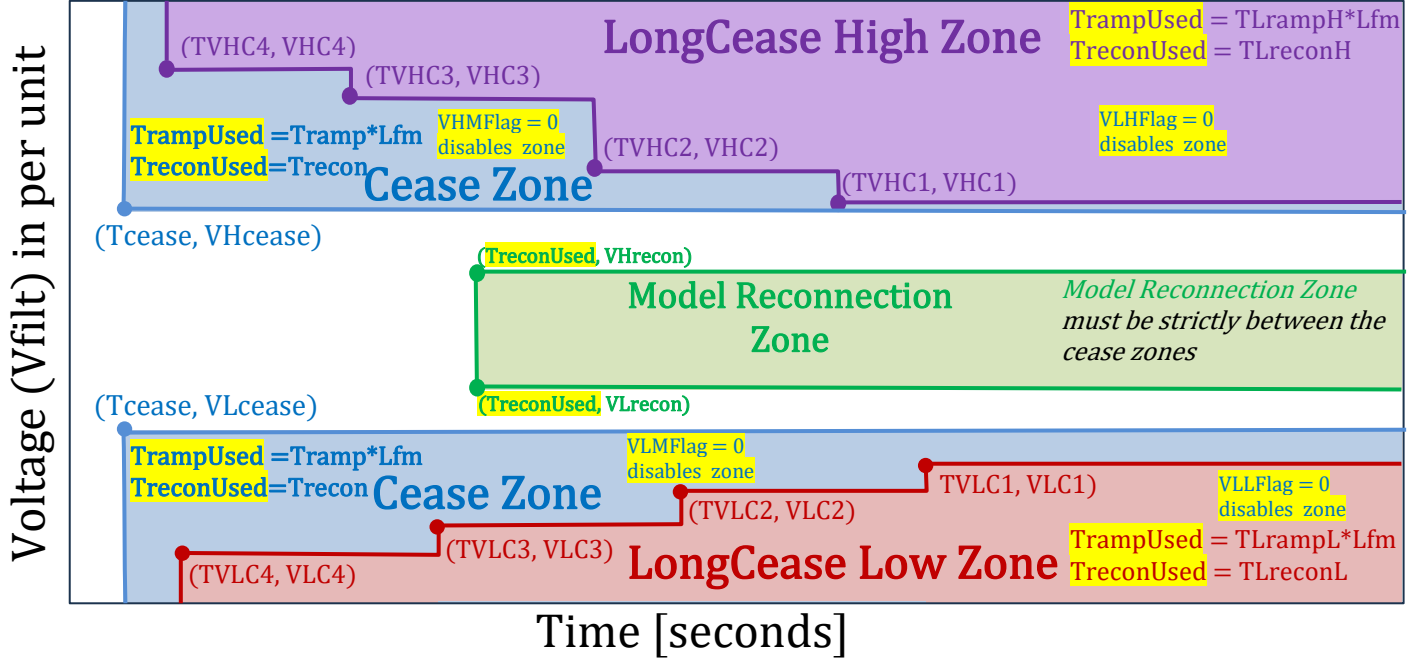
While in Cease mode, if the Red or Purple LongCease Zones are entered, then the reconnection and ramping time will be changed to

TreconUsed = TLreconL (or TLreconH) and TrampUsed = TLrampL*Lfm (or TLrampH*Lfm).

If however, the Blue Cease Zone is entered only, the reconnection time and ramping time will remain

(TreconUsed = Trecon) and (TrampUsed = Tramp*Lfm).

The expectation is Tramp and Trecon are smaller than the values for the Long Cease High and Low Zones.



Cease Level Long Low Timers 4 separate independent timers are used for determining a transition to Cease, Level Long. OR $[(V_{\text{filt}} < \text{VLC1}) \text{ for } \text{TVLC1} \text{ seconds}]$ OR $[(V_{\text{filt}} < \text{VLC2}) \text{ for } \text{TVLC2} \text{ seconds}]$ OR $[(V_{\text{filt}} < \text{VLC3}) \text{ for } \text{TVLC3} \text{ seconds}]$ OR $[(V_{\text{filt}} < \text{VLC4}) \text{ for } \text{TVLC4} \text{ seconds}]$ If any of these 4 timers expires, then we set $\text{TreconUsed} = \text{TLreconL}$ $\text{TrampUsed} = \text{TLrampL} * \text{Lfm}$ Keep checking the Never Reconnect Timers. Start the Tdelay timer if it hasn't started.	Cease Level Long High Timers 4 separate independent timers are used for determining a transition to Cease, Level Long. OR $[(V_{\text{filt}} > \text{VHC1}) \text{ for } \text{TVHC1} \text{ seconds}]$ OR $[(V_{\text{filt}} > \text{VHC2}) \text{ for } \text{TVHC2} \text{ seconds}]$ OR $[(V_{\text{filt}} > \text{VHC3}) \text{ for } \text{TVHC3} \text{ seconds}]$ OR $[(V_{\text{filt}} > \text{VHC4}) \text{ for } \text{TVHC4} \text{ seconds}]$ If any of these 4 timers expires, then we set $\text{TreconUsed} = \text{TLreconH}$ $\text{TrampUsed} = \text{TLrampH} * \text{Lfm}$ Keep checking the Never Reconnect Timers. Start the Tdelay timer if it hasn't started.
--	---

Similar complexity is not drawn for low and high frequency timers. After ceasing due to frequency, reconnection never occurs.

Cease Level Short Timers 1 timers is for determining a transition to Cease, Level Short. $[(V_{\text{filt}} < \text{Vcease}) \text{ for } \text{Tcease} \text{ seconds}]$ If any of these timers expires, then we set $\text{TreconUsed} = \text{Trecon}$ $\text{TrampUsed} = \text{Tramp} * \text{Lfm}$ While in Cease Level Short, continue checking Level Long and NeverReconnect Timers. Start the Tdelay timer if it hasn't started.	Never Reconnect Frequency Timers 2 separate independent timers are used for determining a transition to Cease Level Never Reconnect OR $[(F_{\text{filtHz}} < \text{FLC1}) \text{ for } \text{TFLC1} \text{ seconds}]$ OR $[(F_{\text{filtHz}} > \text{FHC1}) \text{ for } \text{TFHC1} \text{ seconds}]$ Start the Tdelay timer if it hasn't started. If any of these 2 timers expires, then we set $\text{TreconUsed} = 99999999$ (never reconnect) $\text{TrampUsed} = 99999999$ (never reconnect) Start the Tdelay timer if it hasn't started.	Reconnection Timers 1 independent timer is used for determining a transition to a Ramp Mode for reconnection. The voltage and frequency must both be between high and low reconnection thresholds for TreconUsed seconds for the model to transition to Ramping Model $[(V_{\text{filt}} \geq \text{VLrecon}) \text{ AND } (V_{\text{filt}} \leq \text{VHrecon})]$ for TreconUsed seconds Use filtered frequency state from block diagram and converter from per unit deviation to Hz $(F_{\text{filtHz}} = \text{Freq}_{\text{Nominal}} (1 + \text{State5}))$
--	--	--

Variable Names in the Pseudo-Code Below

Model Input Parameters are written in Blue Text

State Variable values are written in Maroon text

Model internal variables are shown in green text

VLMflag, VHMflag, VLLflag, VLHflag, FLflag, FHflag : integer flags
Fcease, Tdelay : input parameters control Ceasing
Frecon, Tramp, Trecon, TLrampL, TLreconL, TLrampH, TLreconH : input parameters control Reconnecting
VLrecon, VHrecon : Thresholds for normal region where we must return for reconnection
Vcease : Thresholds for high and low cease
Tcease : Thresholds Time delay for Cease high and low
TVLC1,TVLC2,TVLC3,TVLC4,TVHC1,TVHC2,TVHC3,TVHC4,TFLC1,TFHC1 : Time delays for 10 cease C timers
VLC1, VLC2, VLC3, VLC4, VHC1, VHC2, VHC3, VHC4, FLC1, FHC1 : Thresholds for 10 cease C timers

Vfilt : State variable of the filtered voltage
Vpu : Terminal Voltage
Ffilt : $60 \cdot (1 + w_{\text{Filt}})$: w_{Filt} is state variable of the filtered per unit frequency deviation

FracOn : Multiplier representing the fraction of the load that remains connected
Mode : 3 possible modes: vtm_Monitor, vtm_Ceased, vtm_Ramp
CeaseLevel : integer: -1=Not active; 0(Cease)=Blue; 1(LongCease)=Red or Purple zone; 2(Never Recon)
TimerDelayActiveLevel : integer: same options as CeaseLevel
TimerDelay : Float used for the Tdelay timer
TStartRamp : Float number used to keep track of timing during ramping
TrampUsed : Float number used to apply the appropriate ramping
depending on which timer has expired
TreconUsed : Float number used to apply the appropriate reconnection
timer depending on which timer has expired
TimerRecon : Float used for the TreconUsed timer
TimerReconActive : Booleans to indicate whether TimerRecon
TimerLV1Active, TimerLV2Active, TimerLV3Active, TimerLV4Active,
TimerHV1Active, TimerHV2Active, TimerHV3Active, TimerHV4Active,
TimerLF1Active, TimerHF1Active,
TimerVLCeaseActive, TimerVHCeaseActive : boolean
TimerLV1, TimerLV2, TimerLV3, TimerLV4,
TimerHV1, TimerHV2, TimerHV3, TimerHV4,
TimerLF1, TimerHF1,
TimerVLCease, TimerVHCease : Float number Timers

Initialize the following before simulation starts

FracOn = 1.0
Mode = vtm_Monitor
CeaseLevel = -1
TimerDelayActiveLevel = -1 // indicates timer not active
TimerDelay = 0
TrampUsed = Tramp*Lfm
TreconUsed = Trecon
TStartRamp = 0
TimerReconActive = FALSE TimerRecon = 0
TimerVLCeaseActive = FALSE TimerVLCease = 0
TimerVHCeaseActive = FALSE TimerVHCease = 0
TimerLV1Active = FALSE TimerLV1 = 0
TimerLV2Active = FALSE TimerLV2 = 0
TimerLV3Active = FALSE TimerLV3 = 0
TimerLV4Active = FALSE TimerLV4 = 0
TimerHV1Active = FALSE TimerHV1 = 0
TimerHV2Active = FALSE TimerHV2 = 0
TimerHV3Active = FALSE TimerHV3 = 0
TimerHV4Active = FALSE TimerHV4 = 0
TimerLF1Active = FALSE TimerLF1 = 0
TimerHF1Active = FALSE TimerLH1 = 0

Helper Procedure for LOCAL HandleTimer

```
Procedure LOCAL_HandleTimer(theMakeTimerActive: Boolean; Out theActive : boolean; Out theTimer : Float);
{
    If (NOT theMakeTimerActive) Then // within range so turn off timer
        If theActive Then
            theActive = FALSE
            // record transition event for user interface
        EndIf
    ElseIf (NOT theActive) Then
        theTimer = PresentTime
        theActive = TRUE
        // record transition event for user interface
    EndIf
}
```

Run the following at each time step

```
{
  If Fcease <= 0 Then Exit // Ignore all this stuff if Fcease <= 0
  Ffilt = NominalFrequency * (1+wFilt) // NominalFrequency = 60 in North America
  If (Mode == vtm_Ramp)
    AND (TimerDelayActiveLevel < 0) // FracOn is frozen during Tdelay, so don't change if active
  Then // update the value of FracOn
    If PresentTime >= TStartRamp + TrampUsed Then
      FracOn = 1 - Fcease + Frecon*Fcease
      Mode = vtm_Monitor
      TrampUsed = Tramp*Lfm // reset back
    Else
      FracOn = 1 - Fcease + Frecon*Fcease*(PresentTime - TStartRamp)/TrampUsed
    EndIf
  EndIf

  //-----
  // Note Various Tcease and Tdelay timers are counting the same when we are in Monitor or Ramp
  // Also depending on the CeaseLevel, some timers continue counting after transition to Cease.
  // There are multiple levels of Cease timers however: Never(2), LongCease(1) and Cease (0)
  // Never timers keep counting after Cease and LongCease timers expire
  // LongCease timers keep counting after Cease timers expire
  // This impacts the calculation of the TrampUsed and TreconUsed values. For example if the model
  // enters Cease mode due to Vcease timers, the Never and LongCease timers are still counting and
  // may change TrampUsed and TreconUsed. Once the TreconUsed timer expires and ramping begins the
  // TrampUsed is fixed the duration of the ramp.
  //-----

  //-----
  // Handle Cease Level Never Reconnect Timers (Level 2)
  //-----
  If ( (Mode <> vtm_Ceased) AND (TimerDelayActiveLevel < 2) ) OR // Not Ceased, but lower TDelayLevel
    ( (Mode == vtm_Ceased) AND (CeaseLevel < 2) ) Then // Ceased but lower Ceaselevel
    If (FLflag <> 0) Then LOCAL_HandleTimer(Ffilt < FLC1, TimerLF1Active, TimerLF1)
    If (FHflag <> 0) Then LOCAL_HandleTimer(Ffilt > FHC1, TimerLH1Active, TimerLH1)
    If ( (FLflag <> 0) AND (TimerLF1Active) AND (PresentTime - TimerLF1) >= TFLC1 ) OR
      ( (FHflag <> 0) AND (TimerHF1Active) AND (PresentTime - TimerHF1) >= TFHC1 )
    Then
      TrampUsed = 999999999 // force it to NEVER reconnect
      TreconUsed = -1 // -1 means will NEVER reconnect
      If (Mode == vtm_Ceased) then
        CeaseLevel = 2 // Already ceased, so change CeaseLevel
      Else
        If (TimerDelayActiveLevel < 0) Then TimerDelay = PresentTime // Only start Delay timer once
        TimerDelayActiveLevel = 2
      EndIf
    EndIf
  EndIf
EndIf

  //-----
  // Handle Cease Level Long High Timers (Level 1) The PURPLE Zone
  //-----
  If (VLHflag <> 0) AND
    ( ( (Mode <> vtm_Ceased) AND (TimerDelayActiveLevel < 1) ) OR // Not Ceased, but lower TDelayLevel
      ( (Mode == vtm_Ceased) AND (CeaseLevel < 1) ) ) Then // Ceased but lower Ceaselevel
    LOCAL_HandleTimer(Vfilt > VHC1, TimerLH1Active, TimerLH1)
    LOCAL_HandleTimer(Vfilt > VHC2, TimerLH2Active, TimerLH2)
    LOCAL_HandleTimer(Vfilt > VHC3, TimerLH3Active, TimerLH3)
    LOCAL_HandleTimer(Vfilt > VHC4, TimerLH4Active, TimerLH4)
    If ( (TimerHV1Active) AND (PresentTime - TimerHV1) >= TVHC1 ) OR
      ( (TimerHV2Active) AND (PresentTime - TimerHV2) >= TVHC2 ) OR
      ( (TimerHV3Active) AND (PresentTime - TimerHV3) >= TVHC3 ) OR
      ( (TimerHV4Active) AND (PresentTime - TimerHV4) >= TVHC4 )
    Then
      TrampUsed = TLrampH*Lfm // record what ramp time to use when reconnecting later
      TreconUsed = TLreconH // record what ramp reconnection to use when reconnecting later
      If (Mode == vtm_Ceased) then
        CeaseLevel = 1 // Already ceased, so change CeaseLevel
      Else
        If (TimerDelayActiveLevel < 0) Then TimerDelay = PresentTime // Only start Delay timer once
        TimerDelayActiveLevel = 1
      EndIf
    EndIf
  EndIf
EndIf
```

```

//-----
// Handle Cease Level Long Timers (Level 1) The RED Zone
If (VLLflag <> 0) AND
  ( ( (Mode <> vtm_Ceased) AND (TimerDelayActiveLevel < 1) ) OR // Not Ceased, but lower TDelayLevel
    ( (Mode == vtm_Ceased) AND (CeaseLevel < 1) ) ) Then // Ceased but lower Ceaselevel
  LOCAL_HandleTimer(Vfilt < VLC1, TimerLV1Active, TimerLV1)
  LOCAL_HandleTimer(Vfilt < VLC2, TimerLV2Active, TimerLV2)
  LOCAL_HandleTimer(Vfilt < VLC3, TimerLV3Active, TimerLV3)
  LOCAL_HandleTimer(Vfilt < VHC4, TimerLV4Active, TimerLV4)
  If ( (TimerLV1Active) AND (PresentTime - TimerLV1) >= TVLC1 ) OR
    ( (TimerLV2Active) AND (PresentTime - TimerLV2) >= TVLC2 ) OR
    ( (TimerLV3Active) AND (PresentTime - TimerLV3) >= TVLC3 ) OR
    ( (TimerLV4Active) AND (PresentTime - TimerLV4) >= TVLC4 )
  Then
    TrampUsed = TLrampL*Lfm // record what ramp time to use when reconnecting later
    TreconUsed = TLreconL // record what ramp reconnection to use when reconnecting later
    If Mode == vtm_Ceased Then
      CeaseLevel = 1 // Already ceased, so change CeaseLevel
    Else
      If (TimerDelayActiveLevel < 0) Then TimerDelay = PresentTime // Only start Delay timer once
      TimerDelayActiveLevel = 1
    EndIf
  EndIf
EndIf
//-----
// Handle Cease Level Timers (Level 0) The BLUE Zone
If ( (Mode <> vtm_Ceased) AND (TimerDelayActiveLevel < 0) ) OR // Not Ceased, but lower TDelayLevel
  ( (Mode == vtm_Ceased) AND (CeaseLevel < 0) ) Then // Ceased but lower Ceaselevel
  If (VLMflag <> 0) Then LOCAL_HandleTimer(Vfilt < VLcease, TimerVLCeaseActive, TimerVLCease) // Low Thresh
  If (VHMflag <> 0) Then LOCAL_HandleTimer(Vfilt > VHcease, TimerVHCeaseActive, TimerVHCease) // High Thresh
  If ( (VLMflag <> 0) AND (TimerVLCeaseActive) AND (PresentTime - TimerVLCease) >= Tcease ) OR
    ( (VHMflag <> 0) AND (TimerVHCeaseActive) AND (PresentTime - TimerVHCease) >= Tcease )
  Then
    TrampUsed = Tramp*Lfm // record what ramp time to use when reconnecting later
    TreconUsed = Trecon // record what ramp reconnection to use when reconnecting later
    If Mode == vtm_Ceased Then
      CeaseLevel = 0 // Already ceased, so change CeaseLevel
    Else
      If (TimerDelayActiveLevel < 0) Then TimerDelay = PresentTime // Only start Delay timer once
      TimerDelayActiveLevel = 0
    EndIf
  EndIf
EndIf
//-----
// Check for transition to ceased mode after delay timer expires
If (Mode <> vtm_Ceased) Then
  If (TimerDelayActiveLevel >= 0) AND ((PresentTime - TimerDelay) >= Tdelay) Then
    CeaseLevel = TimerDelayActiveLevel // Set CeaseLevel to whatever TimerDelayActiveLevel
    Mode = vtm_Ceased
    FracOn = 1 - Fcease
    TimerDelayActiveLevel = -1 // Reset back to not active again
  EndIf
//-----
// Monitoring for reconnection using TreconUsed (TreconUsed could change depending on CeaseLevel)
Else // (Mode == vtm_Ceased)
  LOCAL_HandleTimer( (Vfilt >= VLrecon) AND (Vfilt <= VHrecon), TimerReconActive, TimerRecon)
  If (TimerReconActive) AND
    (TreconUsed >= 0) AND // negative value means it will never reconnect
    (PresentTime - TimerRecon) >= TreconUsed )
  Then
    TimerReconActive = FALSE; // Reset reconnecting timers
    TimerVLCeaseActive = FALSE; TimerVHCeaseActive = FALSE; // Reset ceasing timers
    TimerLV1LActive = FALSE; TimerLV2LActive = FALSE; // Reset ceasing timers
    TimerLV3LActive = FALSE; TimerLV4LActive = FALSE;
    TimerHV1LActive = FALSE; TimerHV2LActive = FALSE;
    TimerHV3LActive = FALSE; TimerHV4LActive = FALSE;
    TimerLF1LActive = FALSE; TimerHF1LActive = FALSE;
    CeaseLevel = -1
    TreconUsed = Trecon // reset back
    If TrampUsed == 0 Then // skip over Ramp Mode and jump to Monitor
      Mode = vtm_Monitor
      FracOn = 1 - Fcease + Frecon*Fcease
    Else
      Mode = vtm_Ramp
      TStartRamp = PresentTime
    EndIf
  EndIf
EndIf

```

```
}
```

Psuedocode for Calculating Ipv and Iqv (Limit Enforcement)

Pctrl : This is state 1 and represents the Power being asked for by the device

Qctrl : This is state 2 and represents the Power being asked for by the device

Vfilt : This is state 6 and represents is the voltage after a delay

nQ, nP, Imax, Ipmín, Ipmín, Iqmin, Iqmax : Model input parameters

Ipv, Iqv: Algebraically calculated values in the model

local_Ipmax, local_Ipmin, local_Iqmax, local_Iqmin, local_temp : various local Floats

```
procedure CalculateIpvIqv(Pctrl, Qctrl, Vfilt : Float;  
    nQ, nP, Imax, Ipmín, Ipmín, Iqmin, Iqmax : Float;  
    Out Ipv, Iqv: Float)
```

```
{  
    // Calculate Values ignoring limits  
    If nP == 0 then Ipv = Pctrl  
    ElseIf nP == 1 then Ipv = Pctrl * (Vfilt/Vinit)  
    ElseIf nP == 2 then Ipv = Pctrl * sqr(Vfilt/Vinit)  
    Else Ipv = Pctrl * Power(Vfilt/Vinit, nP);  
    EndIf  
  
    If nQ == 0 then Iqv = Qctrl  
    ElseIf nQ == 1 then Iqv = Qctrl * (Vfilt/Vinit)  
    ElseIf nQ == 2 then Iqv = Qctrl * sqr(Vfilt/Vinit)  
    Else Iqv = Qctrl * Power(Vfilt/Vinit, nQ)  
    EndIf  
  
    // Convert from power to current, but prevent Divide by Zero  
    If Vfilt > 0.001 Then  
        Ipv := Ipv / Vfilt  
        Iqv := Iqv / Vfilt  
    Else  
        Ipv := Ipv / 0.001  
        Iqv := Iqv / 0.001  
    EndIf  
  
    // set the local variables for storing the max/min values  
    local_Ipmax = Ipmax  
    local_Ipmin = Ipmín  
    local_Iqmax = Iqmax  
    local_Iqmin = Iqmin  
  
    // Apply Limits Ipv first  
    If Imax > 0 Then // Ignore zero or negative Imax  
        If local_Ipmax > +Imax Then local_Ipmax = +Imax  
        If local_Ipmin < -Imax Then local_Ipmin = -Imax  
    EndIf  
    If Ipv > local_Ipmax Then Ipv = local_Ipmax  
    If Ipv < local_Ipmin Then Ipv = local_Ipmin  
  
    // Iqv then gets the current limit that is left over  
    If Imax > 0 Then // Ignore zero or negative Imax  
        local_temp := sqr(Imax) - sqr(Ipv);  
        If local_Temp < 1E-12 Then  
            local_Iqmax = 0  
            local_Iqmin = 0  
        Else  
            local_Temp = sqrt(local_Temp) // "leftover" current for Q  
            if local_Iqmax > +local_Temp then local_Iqmax = +local_Temp  
            if local_Iqmin < -local_Temp then local_Iqmin = -local_Temp  
        EndIf  
    EndIf  
    if Iqv > local_Iqmax then Iqv = local_Iqmax  
    if Iqv < local_Iqmin then Iqv = local_Iqmin  
}
```

Data modifications are run at the beginning of a simulation to modify input parameters temporarily

```
// Do not allow ceasing more than the initial load
If Fcease > 1 then Fcease = 1

// Negative Ramp or Reconnection times will be treated as zero
If Tramp < 0 then Tramp = 0
If TlrampL < 0 then TlrampL = 0
If TlrampH < 0 then TlrampH = 0
If Trecon < 0 then Trecon = 0
If TlrconL < 0 then TlrconL = 0
If TlrconH < 0 then TlrconH = 0

// Vlrcon must be larger than all the Low voltage ceasing thresholds
If Vlrcon < VLcease then Vlrcon = VLcease
If Vlrcon < VLC1 then Vlrcon = VLC1
If Vlrcon < VLC2 then Vlrcon = VLC2
If Vlrcon < VLC3 then Vlrcon = VLC3
If Vlrcon < VLC4 then Vlrcon = VLC4

// Negative or zero thresholds for high voltage ceasing will be ignored
If VHcease <= 0 then VHcease = 999
If VHC1 <= 0 then VHC1 = 999
If VHC2 <= 0 then VHC2 = 999
If VHC3 <= 0 then VHC3 = 999
If VHC4 <= 0 then VHC4 = 999

// VHrecon must be smaller than all the high voltage ceasing thresholds
if VHrecon > VHcease then VHrecon = VHcease
if VHrecon > VHC1 then VHrecon = VHC1
if VHrecon > VHC2 then VHrecon = VHC2
if VHrecon > VHC3 then VHrecon = VHC3
if VHrecon > VHC4 then VHrecon = VHC4

// VHrecon must be larger than Vlrcon. Just ignore VHrecon if that is not true
if VHrecon < Vlrcon then VHrecon = 999;
```