PowerSystemProtection

IEC 61131 Library for ACSELERATOR RTAC® Projects

SEL Automation Controllers

Table of Contents

Section 1: PowerSystemProtection
Introduction 3
Supported Firmware Versions 3
Enumerations 3
Function Blocks 4
Benchmarks 19
Examples 20
Release Notes 27

RTAC LIBRARY

PowerSystemProtection

Introduction

The power system protection library provides function blocks that implement various protection elements for power systems. This library is intended to be used in conjunction with the SEL-2245-42 AC Protection Module.

Special Considerations

- This library is not supported on the SEL-3505 class of RTACs. This is because the SEL-3505 does not support the Axion AC Protection Module. However, this library is not prevented from running on a SEL-3505, but is not tested and has no guarantee on expected behavior.

Supported Firmware Versions

You can use this library on any device configured using ACSELERATOR RTAC® SEL-5033 Software with firmware version R143 or higher.

Library version 3.5.0.0 is meant to be used with the AC Protection Module, which is not supported until RTAC firmware R137. However, there are no explicit checks to enforce this, and the library version 3.5.0.0 can be imported and used in logic in RTAC projects targeting firmware version R132 or later.

Enumerations

Enumerations make code more readable by allowing a specific number to have a readable textual equivalent.

enum_LineBusStates

Given the voltage levels on the bus side and line side of a breaker, the line side and bus side can each be either *live* or *dead*. This enumeration represents the four permutations that result from the two monitored locations, each in one of two states. There are also permutations provided for when one of the states can be confirmed, but not the other.

Enumeration	Description
DLDB	Dead-Line, Dead-Bus
DLLB	Dead-Line, Live-Bus
DLUB	Dead-Line, Undefined-Bus (bus voltage is above dead threshold, but not above live threshold)
LLDB	Live-Line, Dead-Bus
LLLB	Live-Line, Live-Bus
LLUB	Live-Line, Undefined-Bus (bus voltage is above dead threshold, but not above live threshold)
ULDB	Undefined-Line, Dead-Bus (line voltage is above dead threshold, but not above live threshold)
ULLB	Undefined-Line, Live-Bus (bus voltage is above dead threshold, but not above live threshold)
ULUB	Undefined-Line, Undefined-Bus (bus voltage and Line Voltage are both above dead threshold and below live threshold, or the computation is being blocked because of an error)

Function Blocks

fb_DefiniteTime (Function Block)

This function block implements instantaneous and definite-time overcurrent protection functionality.

The settings inputs (starting with *Set*) are read on the first call to the function block after *EN* becomes TRUE or on the first call to the function block. All changes to the settings inputs during runtime are ignored until the next rising edge of *EN* is detected.

Inputs

Name	IEC 61131 Type	Description
EN	BOOL	Enable this function block.
SetPickupSetting	REAL	Pickup current of the protection element. This is the minimum current required to initiate the operation of the protection element.

Inputs

Name	IEC 61131 Type	Description
SetDelayTime	TIME	Time delay for definite-time overcurrent protection element. This should be set to a multiple of the RTAC processing scan time on which this object is instantiated and represents the amount of time *OperatingQuantity* must exceed *PickupSetting* prior to asserting the protection element output.
OperatingQuantity	REAL	The fundamental magnitude of the current. This quantity may be phase (A, B, or C), ground, or negative-sequence current measurement.

Outputs

Name	IEC 61131 Type	Description
ENO	BOOL	The function block is enabled.
PickupSetting	REAL	Pickup current value used in the protection element, read from *SetPickupSetting*.
DelayTime	TIME	Definite-time overcurrent time delay value used in the protection element, read from *SetDelayTime*.
ElementPickup	BOOL	Instantaneous overcurrent protection element has operated. Set to `TRUE` when *OperatingQuantity* >= *PickupSetting*.
PickupTimeOut	BOOL	Definite-time overcurrent protection element has operated. Set to `TRUE` when *OperatingQuantity* > *PickupSetting* for a time longer than *DelayTime*.

Processing

- If *OperatingQuantity* >= *PickupSetting*, then set *ElementPickup* to `TRUE`.
- If ElementPickup has been `TRUE` for longer than *DelayTime*, then set *PickupTimeOut* `TRUE`.
- When OperatingQuantity < PickupSetting, reset the elapsed time in the timer to 0 and set the ElementPickup to `FALSE`.

fb_InverseTimeCurveUser (Function Block)

This function block defines a custom characteristic curve of an inverse-time overcurrent protection element. It implements the analytic equations for the operating time and reset time specified in section 4.2 of IEEE C37.112-1996, *IEEE Standard Inverse-Time Characteristic Equations for Overcurrent Relays*. This function block allows the user to apply constants to *Equation 1* and *Equation 2* to define an inverse-time overcurrent characteristic curve.

The actual operating time or reset time of the function block is guaranteed to be within $\pm 1\%$ of the calculated value or $\pm(2 \times ProcessingInterval_{RTAC})$.

The settings inputs (starting with *Set*) are read on the first call to the function block after *EN* becomes TRUE. All changes to the settings inputs during runtime are ignored until the next rising edge of *EN* is detected.

Equations

$$OperatingTime = TimeDial * (A + \frac{B}{M^C - 1}) \quad \text{(Equation 1)}$$

$$ResetTime = TimeDial * (\frac{R}{1 - M^2}) \quad \text{(Equation 2)}$$

where:

- M is the applied multiple of pickup current
  - for operating time, M > 1
  - for reset time, M < 1

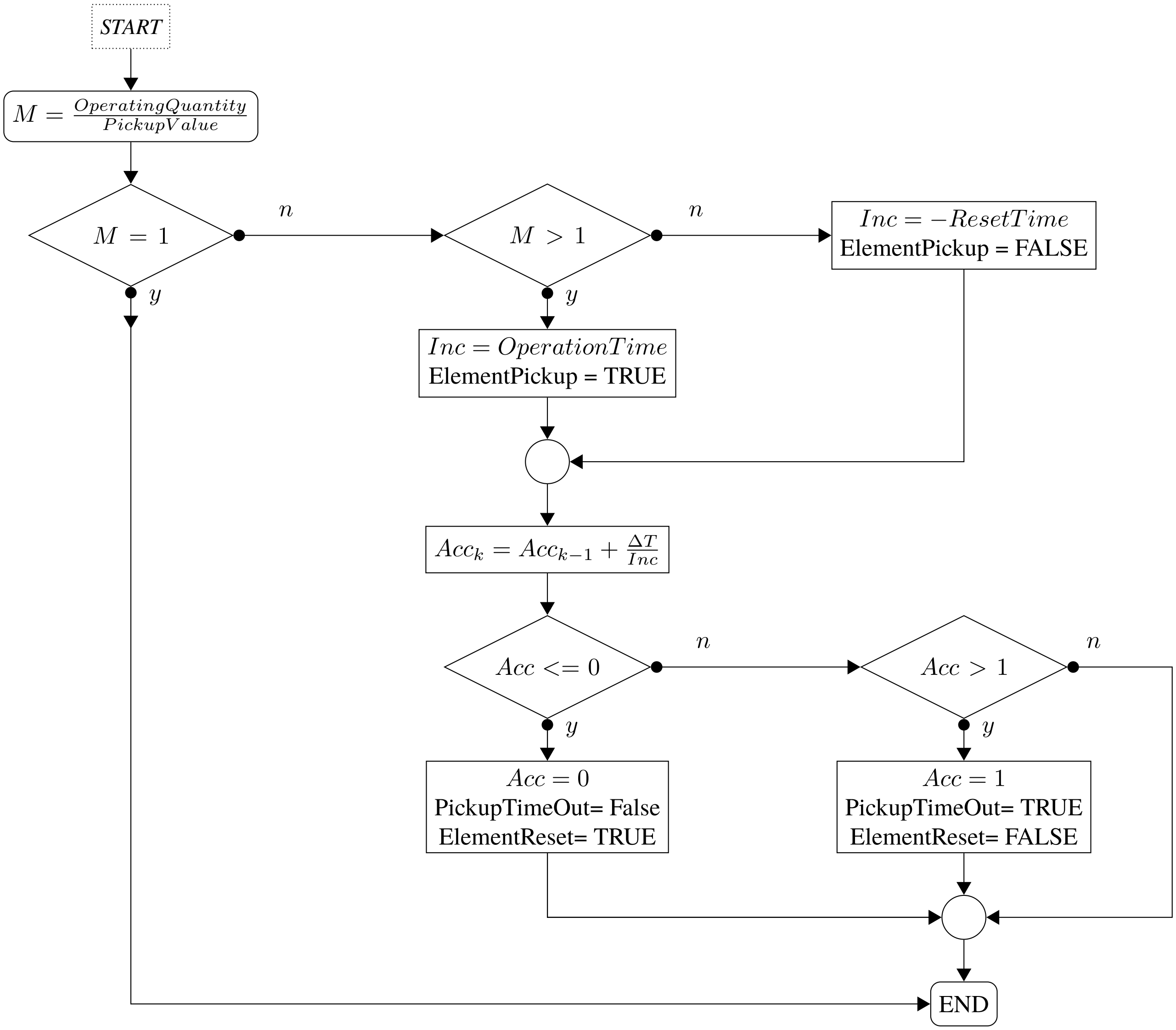
Inputs

Name	IEC 61131 Type	Description
EN	BOOL	Enable this function block.
SetPickupSetting	REAL	Pickup current magnitude of the protection element. If *OperatingQuantity* goes above this quantity, the protection element initiates its operation. This input is read once on first call to the function block. All changes to this input during runtime are ignored.
SetTimeDial	REAL	Time-dial setting. This input adjusts the protection element to a predetermined trip time at a specified current, as described in IEEE C37.112-1996, *IEEE Standard Inverse-Time Characteristic Equations for Overcurrent Relays*. This input is read once on first call to the function block. All changes to this input during runtime are ignored.
OperatingQuantity	REAL	The fundamental magnitude of the current. This quantity may be phase (A, B, or C), ground, or negative-sequence current measurement.
SetA	REAL	Sets user-defined A parameter for inverse curve calculation (see *Equation 1*).
SetB	REAL	Sets user-defined B parameter for inverse curve calculation (see *Equation 1*).
SetC	REAL	Sets user-defined C parameter for inverse curve calculation (see *Equation 1*).
SetR	REAL	Sets user-defined R parameter for inverse curve calculation (see *Equation 2*).

Outputs

Name	IEC 61131 Type	Description
ENO	BOOL	The function block is enabled.
PickupSetting	REAL	Pickup current value used in the protection element. This value is read from *SetPickupSetting* on first run of the function block.
TimeDial	REAL	Time-dial setting value used in the protection element. This value is read from *SetTimeDial* on first run of the function block.
ElementPickup	BOOL	The protection element has initiated operation. Set to TRUE if *OperatingQuantity* > *PickupSetting*.
PickupTimeOut	BOOL	The protection element has operated. Set to TRUE when the operating time expires.
ElementReset	BOOL	The protection element has reset (refer to section 3.2 of IEEE C37.112-1996, *IEEE Standard Inverse-Time Characteristic Equations for Overcurrent Relays*).
Aval	REAL	Value being used that was set from *SetA*.
Bval	REAL	Value being used that was set from *SetB*.
Cval	REAL	Value being used that was set from *SetC*.
Rval	REAL	Value being used that was set from *SetR*.

Processing



Predefined Inverse-Time Overcurrent Function Blocks

The function blocks listed below implement the U.S. and IEC inverse-time overcurrent curves, in accordance with IEEE C37.112-1996, *IEEE Standard Inverse-Time Characteristic Equations for Overcurrent Relays*.

The curve type, operating type, and reset time equations associated with each function block are listed in *Table 1* and *Table 2*.

The settings inputs (starting with *Set*) are read on the first call to the function block after *EN* becomes TRUE. All changes to the settings inputs during runtime are ignored until the next rising edge of *EN* is detected.

Table 1 IEC Equations Associated With Predefined Inverse-Time Overcurrent Function Blocks

Curve	Function Block Name	Operating Time (sec)	Reset Time (sec)
C1–Standard Inverse	fb_InverseTimeCurveC1	$TD * (\frac{0.14}{M^{0.02}-1})$	$TD * (\frac{13.5}{1-M^2})$
C2–Very Inverse	fb_InverseTimeCurveC2	$TD * (\frac{13.5}{M-1})$	$TD * (\frac{47.3}{1-M^2})$
C3–Extremely Inverse	fb_InverseTimeCurveC3	$TD * (\frac{80}{M^2-1})$	$TD * (\frac{80}{1-M^2})$
C4–Long-Time Inverse	fb_InverseTimeCurveC4	$TD * (\frac{120}{M-1})$	$TD * (\frac{120}{1-M})$
C5–Short-Time Inverse	fb_InverseTimeCurveC5	$TD * (\frac{0.05}{M^{0.04}-1})$	$TD * (\frac{4.85}{1-M^2})$

Table 2 U.S. Equations Associated With Predefined Inverse-Time Overcurrent Function Blocks

Curve	Function Block Name	Operating Time (sec)	Reset Time (sec)
U1–Moderately Inverse	fb_InverseTimeCurveU1	$TD * (0.0226 + \frac{0.0104}{M^{0.02}-1})$	$TD * (\frac{1.08}{1-M^2})$
U2–Inverse	fb_InverseTimeCurveU2	$TD * (0.180 + \frac{5.95}{M^2-1})$	$TD * (\frac{5.95}{1-M^2})$
U3–Very Inverse	fb_InverseTimeCurveU3	$TD * (0.0963 + \frac{3.88}{M^2-1})$	$TD * (\frac{3.88}{1-M^2})$
U4–Extremely Inverse	fb_InverseTimeCurveU4	$TD * (0.0352 + \frac{5.67}{M^2-1})$	$TD * (\frac{5.67}{1-M^2})$
U5–Short-Time Inverse	fb_InverseTimeCurveU5	$TD * (0.00262 + \frac{0.00342}{M^{0.02}-1})$	$TD * (\frac{0.323}{1-M^2})$

where:

- TD = time-dial setting
- M = applied multiple of pickup current
  - for operating time, M > 1;
  - for reset time, M < 1

Inputs

Name	IEC 61131 Type	Description
EN	BOOL	Enable this function block.
SetPickupSetting	REAL	Pickup current magnitude of the protection element. If *OperatingQuantity* goes above this quantity, the protection element initiates its operation. This input is read once on first call to the function block. All changes to this input during runtime are ignored.
SetTimeDial	REAL	Time-dial setting. This input adjusts the protection element to a predetermined trip time at a specified current, as described in IEEE C37.112-1996, *IEEE Standard Inverse-Time Characteristic Equations for Overcurrent Relays*. This input is read once on first call to the function block. All changes to this input during runtime are ignored.
OperatingQuantity	REAL	The fundamental magnitude of the current. This quantity may be phase (A, B, or C), ground, or negative-sequence current measurement.

Outputs

Name	IEC 61131 Type	Description
ENO	BOOL	The function block is enabled.
PickupSetting	REAL	Pickup current value used in the protection element. This value is read from *SetPickupSetting* on first run of the function block.
TimeDial	REAL	Time-dial setting value used in the protection element. This value is read from *SetTimeDial* on first run of the function block.
ElementPickup	BOOL	The protection element has initiated operation. Set to TRUE if *OperatingQuantity* > *PickupSetting*.
PickupTimeOut	BOOL	The protection element has operated. Set to TRUE when the operating time expires.
ElementReset	BOOL	The protection element has reset (refer to section 3.2 of IEEE C37.112-1996, *IEEE Standard Inverse-Time Characteristic Equations for Overcurrent Relays*).

Processing

The calculations for these predefined curves happens exactly the same as for the *fb_InverseTimeCurveUser (Function Block) on page 5*, except that the coefficients used are hard-coded instead of user-settable.

fb_LossOfPotential (Function Block)

Fuses often protect the secondary windings of the power system potential transformers. The loss-of-potential logic is used to detect blown potential transformer fuses. The output of this function block should be used to disable protection elements that rely on voltage inputs so that voltage-based protection is performed securely and does not cause tripping to occur when the transformer fuse is blown.

The function block declares a loss-of-potential condition if V1 drops in magnitude by at least ten percent and there is no corresponding change in the magnitude or angle of I1 or I0. A loss-of-potential condition persisting for 15 power cycles causes the loss-of-potential output to latch. The output resets when V1 returns to a level greater than 85 percent nominal voltage and V0 is less than 10 percent of the positive-sequence voltage (V1).

This function block is intended to be used with the SEL-2245-42 AC Protection Module.

When using the LossOfPotential function block, make sure that the RTAC task cycle time is set to 4 ms.

The settings inputs (starting with *Set*) are read on the first call to the function block after *EN* becomes TRUE. All changes to the settings inputs during runtime are ignored until the next rising edge of *EN* is detected.

Inputs

Name	IEC 61131 Type	Description
EN	BOOL	Enable this function block.
SetNominalFrequency	DINT(50..60)	The nominal frequency (Hz) of the power system (50 or 60). Recommend setting this input equal to the RTAC system tag: `SystemTags.Nominal_Frequency.stVal`
SetAmpsNominal	REAL	The maximum nominal amperage expected (usually 1 A or 5 A secondary).
SetVoltsNominal	REAL	The nominal line-to-line voltage.
VoltsPosSeq	vector_t	The positive-sequence fundamental voltage vector.
VoltsZeroSeq	vector_t	The zero-sequence voltage fundamental vector.
AmpsZeroSeq	vector_t	The zero-sequence current fundamental vector.
AmpsPosSeq	vector_t	The positive-sequence current fundamental vector.
AmpsNegSeq	vector_t	The negative-sequence current fundamental vector.
AmpsPhaseA	vector_t	The fundamental Phase A current.
AmpsPhaseB	vector_t	The fundamental Phase B current.
AmpsPhaseC	vector_t	The fundamental Phase C current.
FundQok	BOOL	The fundamental quantities above are reporting good quality. If values are being read from the SEL-2245-42 module, then enable the `QUALITY_FUND` tag on the module and assign that tag to this input.
FundTimeStamp	timestamp_t	The time stamp of the fundamental quantity measurements. Assign the `TIMESTAMP_FUND`. tag on the SEL-2245-42 module to this input.

Outputs

Name	IEC 61131 Type	Description
ENO	BOOL	The function block is enabled
ErrorDesc	STRING(80)	Displays an error description if one exists
NominalFrequency	DINT[50,60]	The nominal frequency of the power system (Hz).
AmpsNominal	REAL	The magnitude of the maximum nominal amperage expected, read once from *SetAmpsNominal*.
VoltsNominal	REAL	The magnitude of the nominal line-to-line voltage. This value is read once from *SetVoltsNominal*.
LossOfPotential	BOOL	Loss-of-potential condition detected. If *ENO* is `FALSE`, then this value returns to the default `TRUE` state. Use this output to disable protection elements that use voltage to trip.

Processing

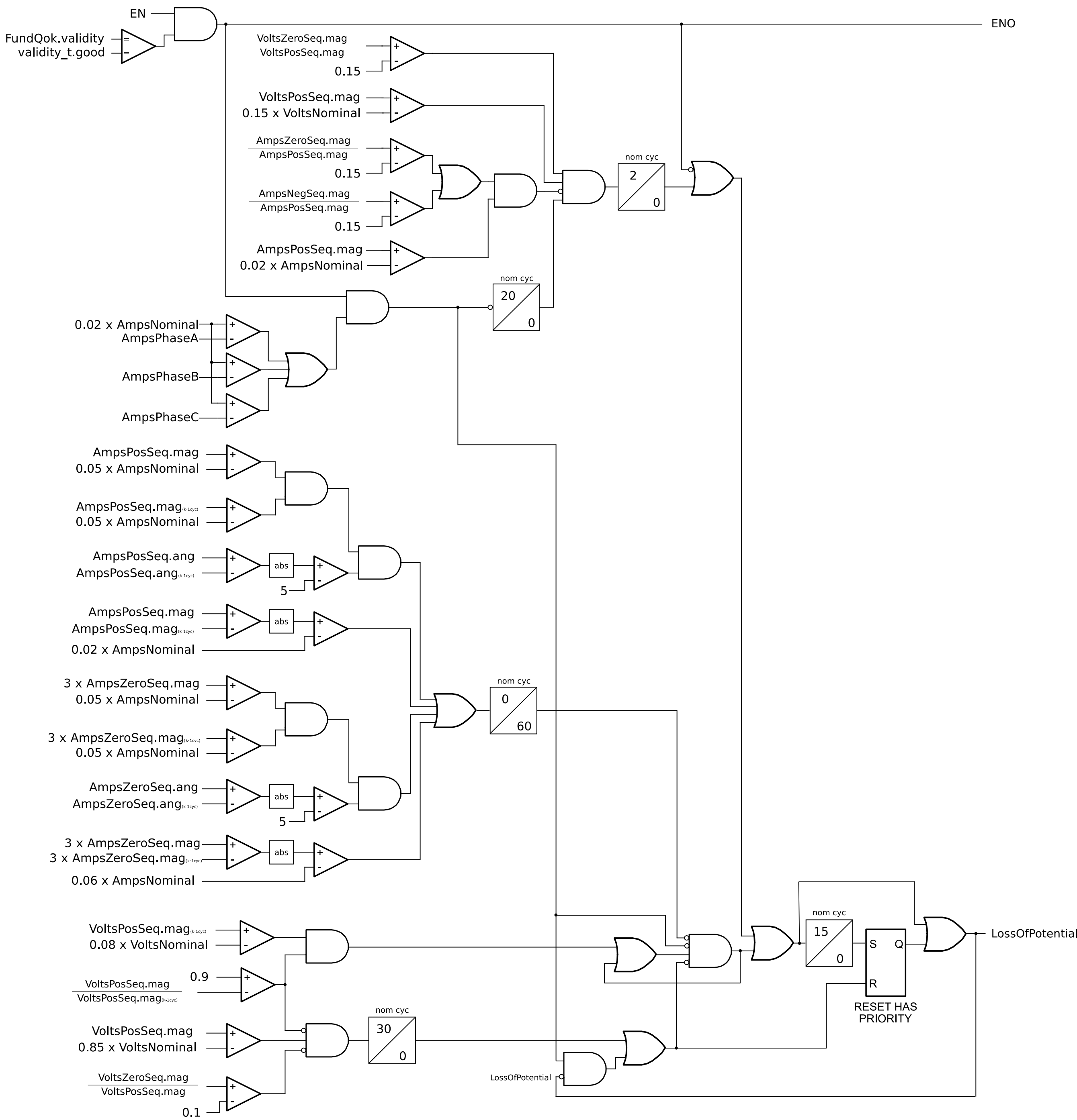


Figure 1 Loss-of-Potential Logic

fb_OverVoltage (Function Block)

Asserts *OverVoltage* when the measured voltage is higher than the threshold setting and the function block is enabled.

Inputs

Name	IEC 61131 Type	Description
EN	BOOL	Enable this function block. This should be set to FALSE if the quantity being measured is bad quality or a loss-of-potential condition is detected.
ThresholdVoltage	REAL	The voltage to compare the magnitude against.
MeasuredVoltage	REAL	The magnitude of the voltage to measure.

Outputs

Name	IEC 61131 Type	Description
ENO	BOOL	This function block is enabled.
OverVoltage	BOOL	The *MeasuredVoltage* is exceeding the voltage threshold.

Processing

The *OverVoltage* and *ENO* outputs are computed based on the input states according to the following logic diagram:

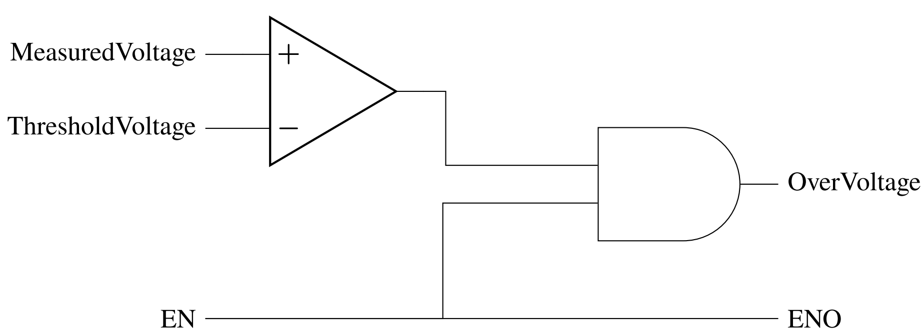


Figure 2 Overvoltage Logic

fb_UnderVoltage (Function Block)

Asserts *UnderVoltage* when the measured voltage is lower than the threshold setting and the function block is enabled.

Inputs

Name	IEC 61131 Type	Description
EN	BOOL	Enable this function block. This should be set to FALSE if the quantity being measured is bad quality or a loss-of-potential condition is detected.
ThresholdVoltage	REAL	The voltage to compare the magnitude against.
MeasuredVoltage	REAL	The magnitude of the voltage to measure.

Outputs

Name	IEC 61131 Type	Description
ENO	BOOL	This function block is enabled.
UnderVoltage	BOOL	The *MeasuredVoltage* is presently less than the threshold voltage.

Processing

The *UnderVoltage* and *ENO* outputs are computed based on the input states according to the following logic diagram:

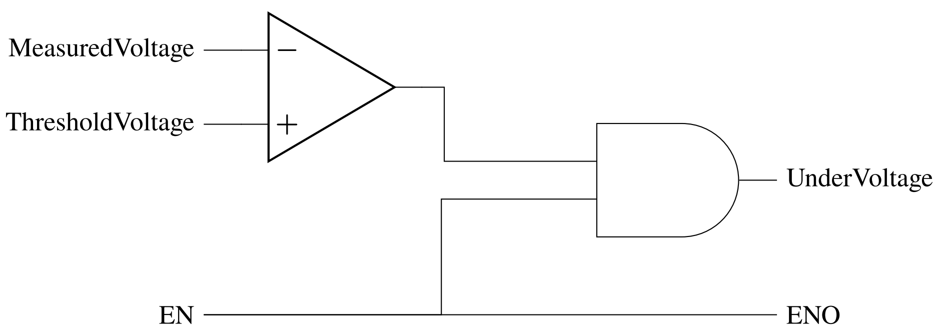


Figure 3 Undervoltage Logic

fb_BusLineVoltageCheck (Function Block)

Checks the single-phase voltage levels on the bus side and line side of a breaker. The line side and bus side can both be either “live” or “dead”, and this function block provides the logic to decide which permutation the voltages on each side of the breaker are in (see the *Enumerations on page 3* section for the listed enumeration).

Inputs

Name	IEC 61131 Type	Description
EN	BOOL	Enable this function block. Set this to FALSE if the quantity being measured is bad quality or a loss-of-potential condition is detected.
LineDeadThreshold	REAL	The voltage at which to declare the line dead.
LineLiveThreshold	REAL	The voltage at which to declare the line live.
BusDeadThreshold	REAL	The voltage at which to declare the bus dead.
BusLiveThreshold	REAL	The voltage at which to declare the bus live.
LineVoltage	REAL	The measured voltage of the line.
BusVoltage	REAL	The measured voltage of the bus.

Outputs

Name	IEC 61131 Type	Description
ENO	BOOL	This function block is enabled.
ErrorDesc	STRING(80)	Displays an error description, if one exists.
LineBusState	enum_LineBusStates	The decided permutation.

Processing

- If *LineDeadThreshold* > *LineLiveThreshold*, then an error is displayed in *ErrorDesc*.
- If *BusDeadThreshold* > *BusLiveThreshold*, then an error is displayed in *ErrorDesc*.
- When the following conditions are met, *ENO* is set to `TRUE`:
  - *LineDeadThreshold* ≤ *LineLiveThreshold*.
  - *BusDeadThreshold* ≤ *BusLiveThreshold*.
  - *EN* = `TRUE`.
- If *ENO* = `TRUE`, then the bus and line states are computed independently using this logic:
  - If the voltage is below the associated dead threshold, declare it dead.
  - If the voltage is above the associated live threshold, declare it live.
  - If neither of the above conditions are `TRUE`, declare it dead.
- If *ENO* = `FALSE`, then the *LineBusState* is set to ULUB (undefined bus, undefined line).

fb_ConductorThermalOverload (Function Block)

This function block estimates the temperature of a conductor by measuring the current flowing through that conductor and the ambient temperature and declares an overload condition when the approximated temperature exceeds the provided maximum temperature.

The settings inputs (starting with *Set*) are read on the first call to the function block after *EN* becomes TRUE or on the or first call to the function block. All changes to the settings inputs during runtime are ignored until the next rising edge of *EN* is detected.

Inputs

Name	IEC 61131 Type	Description
EN	BOOL	Enables the computations of this function block.
SetInitialTemp	REAL	Given in °C; the assumed initial temperature of the line. This initializes the output *CalculatedTemp* on the first scan that *EN* is set to TRUE.
SetTimeConstant	TIME	The time constant ($\mathcal{T}$) which represents the heating/cooling constant of the conductor associated with the thermal capacity of the line.
SetRatedCurrent	REAL	The amount of current which the conductor is rated to carry continuously. Units must match with the *MeasuredCurrent*.
SetReferenceAmbientTemp	REAL	The ambient temperature in °C at which *MaxConductorTemp* was measured.
SetMaxConductorTemp	REAL	Given in °C; the steady state temperature if rated current flows continuously.
MeasuredCurrent	REAL	The rms current presently measured flowing in the conductor. Units must match the *RatedCurrent* value provided.
AmbientTemp	REAL	The ambient dry-bulb temperature in °C. The highest measured air temperature experienced by the conductor over its span.

Outputs

Name	IEC 61131 Type	Description
ENO	BOOL	This function block is active.
TimeConstant	TIME	The value ($\mathcal{T}$) read from *SetTimeConstant*.
RatedCurrent	REAL	The value read from *SetRatedCurrent*.
ReferenceAmbientTemp	REAL	The value read from *SetReferenceAmbientTemp*.
MaxConductorTemp	REAL	The value read from *SetMaxConductorTemp*.
CalculatedTemp	REAL	Based on the parameters provided, this is the calculated temperature of the conductor, given in °C. Initialized by *SetInitialTemp*.
TimeToOverload	TIME	Assuming the inputs *MeasuredCurrent* and *AmbientTemp* remain at their present value, this is the amount of time remaining until an overload condition is declared.

Outputs

Name	IEC 61131 Type	Description
Overloaded	BOOL	*MaxConductorTemperature* has been reached.

Processing

This section describes the formulas used by this function block to calculate the outputs.

Input Validation Before Enabling

Inputs are read once and the *ENO* output is set to TRUE at the rising edge of the following combination of input conditions: $EN = TRUE$ AND $SetRatedCurrent > 0$.

Once set, the values used by the function block are displayed on the outputs and will not be read again until the rising edge described above is again detected.

Temperature Calculation

An internal model is employed which assumes that the thermal dissipation capacity of the conductor varies only with environmental temperature, which can either be entered manually, or a temperature sensor can be used to measure the actual environmental air temperature.

In the following, the uppercase Latin character T is used for temperature (°C) and the uppercase Greek $\mathcal{T}$ for time constant. Lowercase t is used to represent time.

Most of the internal temperatures used in the equations are relative to ambient temperature. These relative temperatures are represented by $T_{\triangle}$.

The temperature rise beyond ambient that is experienced by the conductor at steady-state when carrying the maximum rated current is given by *Equation 3*.

$$T_{\Delta n} = MaxConductorTemp - ReferenceAmbientTemp \qquad \text{(Equation 3)}$$

Where the *MaxConductorTemp* and *ReferenceAmbientTemp* are provided by the manufacturer. *MaxConductorTemp* is the temperature experienced by the conductor at steady-state when carrying rated current when the surrounding air is at the *ReferenceAmbientTemp*.

The assumed maximum line temperature for this scan is shown as $T_{CalculatedTemp_k}$, and the correlating calculated temperature is given in *Equation 4*.

$$T_{\Delta k} = CalculatedTemp - AmbientTemp \qquad \text{(Equation 4)}$$

Equation 5 shows the thermal differential equation, where h is the heat transfer coefficient of the conductor surface and A is the area of the surface of the conductor.

$$\frac{dT}{dt} = \frac{1}{\mathcal{T}} \left(\frac{I^2 R}{hA} - T \right) \qquad \text{(Equation 5)}$$

Equation 6 and *Equation 7* show the solution of *Equation 5* in time-domain and discrete domain, respectively.

$$T_\Delta = \frac{I^2R}{hA}\left(1 - e^{-\frac{t}{\mathcal{T}}}\right) + T_{\Delta Initial}e^{-\frac{t}{\mathcal{T}}} \quad \text{(Equation 6)}$$

$$T_{\Delta k} = \frac{\frac{\Delta t}{\mathcal{T}}\left(\frac{R}{\mathcal{T}hA}\right)I^2 + T_{\Delta k-1}}{1 + \frac{\Delta t}{\mathcal{T}}} \quad \text{(Equation 7)}$$

We can use a particular solution to find the relation between the coefficients R, h, and A: if rated current I_n flows for a long amount of time (infinite), the steady state temperature of the conductor is $T_{\Delta n}$. Substituting these values in *Equation 6*, the following equation can be obtained:

$$\frac{R}{hA} = \frac{T_{\Delta n}}{I_n^2} \quad \text{(Equation 8)}$$

Substituting *Equation 8* in *Equation 7*:

$$T_{\Delta k} = \frac{\frac{\Delta t}{\mathcal{T}}\left(\frac{I}{I_n}\right)^2 T_{\Delta n} + T_{\Delta k-1}}{1 + \frac{\Delta t}{\mathcal{T}}} \quad \text{(Equation 9)}$$

Equation 3 and *Equation 4* may be expressed as follows:

$$T_{\Delta k} = T_{Calculated_k} - T_{Ambient} \quad \text{(Equation 10)}$$

$$T_{\Delta k-1} = T_{k-1} - T_{Ambient} \quad \text{(Equation 11)}$$

$$T_{\Delta n} = T_{MaxConductor} - T_{Reference} \quad \text{(Equation 12)}$$

Substituting *Equation 10*, *Equation 11*, and *Equation 12* in *Equation 9*, we get the equation that is used by this function block to calculate the temperature of the conductor:

$$\boxed{T_{Calculated_k} = \frac{\left(\frac{I}{I_n}\right)^2 (T_{MaxConductor} - T_{Reference}) + \frac{\mathcal{T}}{\Delta t}T_{k-1} + T_{Ambient}}{1 + \frac{\mathcal{T}}{\Delta t}}} \quad \text{(Equation 13)}$$

where:

$$\begin{aligned} T_{Calculated_k} &\equiv CalculatedTemp \\ T_{MaxConductor} &\equiv MaxConductorTemp \\ T_{Reference} &\equiv ReferenceAmbientTemp \\ T_{Ambient} &\equiv AmbientTemp \\ I &\equiv MeasuredCurrent \\ I_n &\equiv RatedCurrent \end{aligned}$$

Time to Overload Calculation

When the generation of heat exceeds the estimated thermal dissipation capacity, the estimated temperature of the conductor is raised at a rate provided by the user (*TimeConstant*), which models the thermal capacity of the line.

On each execution of the function block, a new estimated line temperature - $T_{\Delta k}$ is calculated using *Equation 13*.

The *CalculatedTemp* is then compared to the MaxConductorTemp setting for this function block. If the new estimated temperature is above the value: *TempThreshold*, then *Overloaded* is set to True and *TimeToOverload* is set to 0.

Internally, the predicted steady-state temperature which will result if the inputs remain unchanged from their present state (T_{ss}) is calculated using *Equation 14*.

$$T_{ss} = \frac{I^2 T_{\Delta n}}{I_n^2} + T_{Ambient} \quad \text{(Equation 14)}$$

While the steady-state temperature $T_{ss} > T_{MaxConductor}$ and $T_{Calculated_k} < T_{MaxConductor}$ then there is an imminent overload condition. The time to the overload is calculated using *Equation 15*:

$$\boxed{TimeToOverload = \mathcal{T} \times \ln \frac{T_{ss} - T_{Calculated_k}}{T_{ss} - T_{MaxConductor}}} \quad \text{(Equation 15)}$$

If the internal steady-state temperature is lower than TempThreshold, the *TimeToOverload* output is set to the maximum value stored by type `TIME`.

Benchmarks

Benchmark Platforms

The benchmarking tests recorded for this library are performed on the following platforms.

- SEL-3530 / SEL-2241
  - R136 firmware
- SEL-3555
  - Dual-core Intel i7-3555LE processor
  - 4 GB ECC RAM
  - R136 firmware

Benchmark Test Descriptions

Execution Time of Overcurrent Function Blocks

The posted time is the average execution time of 1000 consecutive calls to the function blocks where the operating quantity alternates between a value that is barely above the pickup threshold, to a value of 0.

This test is performed on the following function blocks:

- fb_DefiniteTime

- fb_InverseTimeCurveUser (The performance results for this function block are also valid for any of the other inverse time curve blocks for the specific curves.)

Execution Time of Voltage Element Function Blocks

The posted time is the average execution time of 1000 consecutive calls to the function blocks.

This test is performed on the following function blocks:

- fb_BusLineVoltageCheck
- fb_OverVoltage
- fb_UnderVoltage

Execution Time of LossOfPotential Function Block

The posted time is the average execution time of 1000 consecutive calls to the fb_LossOf-Potential function block.

Execution Time of Thermal Overload Function Blocks

The posted time is the average execution time of 1000 consecutive calls to the function blocks.

This test is performed on the `fb_ConductorThermalOverload` function block.

Benchmark Results

Operation Tested	**Platform (time in μs)**	
	SEL-3530 SEL-2241	**SEL-3555**
fb_DefiniteTime	3.1	0.5
fb_InverseTimeCurveUser	12.6	0.9
fb_LossOfPotential	10.8	0.4
fb_BusLineVoltageCheck	5.2	0.3
fb_OverVoltage	0.9	0.1
fb_UnderVoltage	1.1	0.1
fb_ConductorThermalOverload	15.0	0.8

Examples

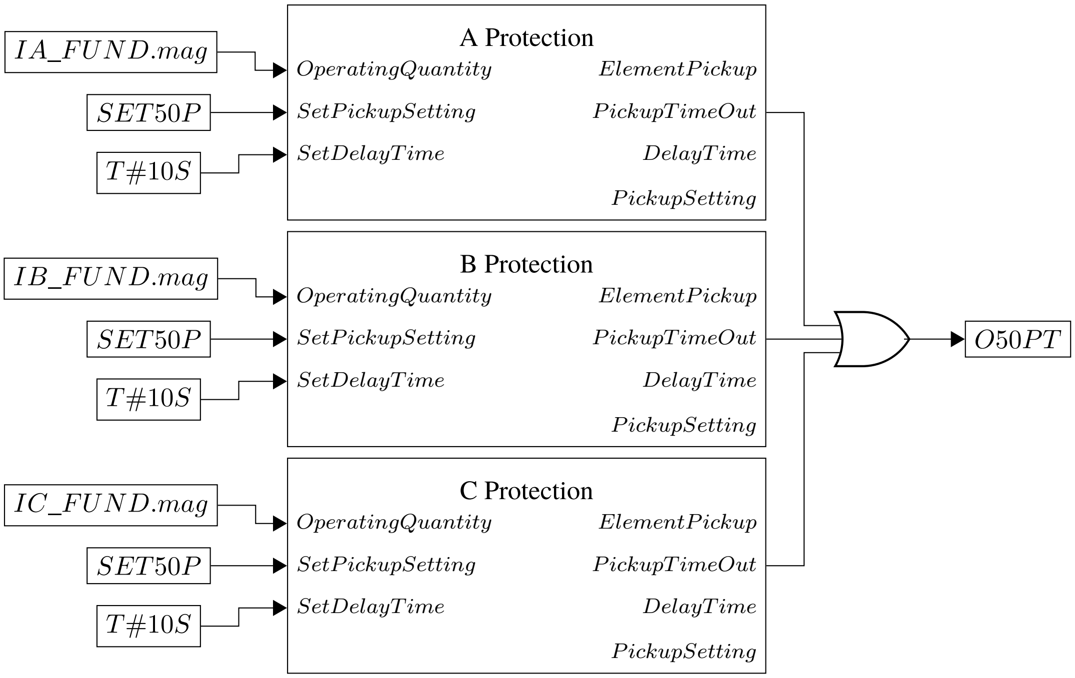
These examples demonstrate the capabilities of this library. Do not mistake them as suggestions or recommendations from SEL.

Implement the best practices of your organization when using these libraries. As the user of this library, you are responsible for ensuring correct implementation and verifying that the project using these libraries performs as expected.

Create a Three-Phase Definite-Time Overcurrent Element

Objective

The user would like to create a program that implements a Three-Phase Definite-Time Overcurrent element that asserts when any phase is detected overcurrent. The diagram is shown below.



Assumptions

This example assumes the user has an AC Protection Module setup to get current input measurements.

Solution

The user can create a program shown in *Code Snippet 1*:

Code Snippet 1 prg_50Element

```
PROGRAM prg_50Element
VAR
    //50 protection elements
    AProtection : PowerSystemProtection.fb_DefiniteTime;
    BProtection : PowerSystemProtection.fb_DefiniteTime;
    CProtection : PowerSystemProtection.fb_DefiniteTime;

    //User defined inputs to 50 element
    SET50P : REAL := 100;
    SET50PTD : TIME := T#1S;

    //Program Outputs
    O50PT : BOOL;
END_VAR
```

```
AProtection(OperatingQuantity := SEL_prCTPT_1_ECAT.IA_FUND.mag,
           SetPickupSetting := SET50P,
           SetDelayTime := SET50PTD);
BProtection(OperatingQuantity := SEL_prCTPT_1_ECAT.IB_FUND.mag,
           SetPickupSetting := SET50P,
           SetDelayTime := SET50PTD);
CProtection(OperatingQuantity := SEL_prCTPT_1_ECAT.IC_FUND.mag,
           SetPickupSetting := SET50P,
           SetDelayTime := SET50PTD);

O50PT := AProtection.PickupTimeout
      OR BProtection.PickupTimeout
      OR CProtection.PickupTimeout;
```

Create a Three-Phase Inverse-Time Overcurrent Element

Objective

The user would like to create a program with behavior of 51 Three-Phase Inverse-Time Overcurrent ORMODE outputs.

Assumptions

This example assumes the user has a SEL-2245-42 AC Protection Module configured in the RTAC project to obtain fundamental measurements.

NOTE: The module is assumed to have the name: "SEL_prCTPT_1_ECAT".

Solution

The user can create a program shown in *Code Snippet 2*:

Code Snippet 2 prg_51Element

```
PROGRAM prg_51Element
VAR
//Inverse Time Curve Function Blocks
Aphase : PowerSystemProtection.fb_InverseTimeCurveC2;
Bphase : PowerSystemProtection.fb_InverseTimeCurveC2;
Cphase : PowerSystemProtection.fb_InverseTimeCurveC2;
Nphase : PowerSystemProtection.fb_InverseTimeCurveC2;
//Phase Overcurrent Settings
Set51PPU : REAL := 0;
Set51PTD : REAL := 1.0;
//Neutral Overcurrent Settings
Set51NPU : REAL := 0;
Set51NTD : TIME := T#1S;
//OR Mode Outputs
O51PT : BOOL;
O51PR : BOOL;
O51P  : BOOL;
END_VAR
```

```
Aphase( OperatingQuantity :=SEL_prCTPT_1_ECAT.IA_FUND.mag,
       SetPickupSetting := Set51PPU,
       SetTimeDial := Set51PTD );

Bphase( OperatingQuantity :=SEL_prCTPT_1_ECAT.IB_FUND.mag,
       SetPickupSetting := Set51PPU,
       SetTimeDial := Set51PTD );

Cphase( OperatingQuantity :=SEL_prCTPT_1_ECAT.IC_FUND.mag,
       SetPickupSetting := Set51PPU,
       SetTimeDial := Set51PTD );

O51P := Aphase.ElementPickup
     AND Bphase.ElementPickup
     AND Cphase.ElementPickup;

O51PT := Aphase.PickupTimeOut
     AND Bphase.PickupTimeOut
     AND Cphase.PickupTimeOut;

O51PR := Aphase.ElementReset
     AND Bphase.ElementReset
     AND Cphase.ElementReset;
```

Detect a Loss-of-Potential Condition

Objective

The user would like to detect an overvoltage condition of Phase A voltage and, when one or more of the PT fuses or breakers is blown, prevent Phase A overvoltage detection from enabling LED 1 on an RTAC/Axion.

Assumptions

This example assumes the user has a SEL-2245-42 AC Protection Module configured in the RTAC project to obtain fundamental measurements.

For voltage Phase A overvoltage detection, the current and voltage phasors are assumed to be:

NOTE: The module is assumed to have the name: "SEL_prCTPT_1_ECAT".

```
Current Phase A: 4 A ∠ 0
Current Phase B: 4 A ∠ -120
Current Phase C: 4 A ∠ 120
Voltage Phase A: 70 V ∠ 0
Voltage Phase B: 67 V ∠ -120
Voltage Phase C: 67 V ∠ 120
```

A loss-of-potential condition can be triggered by using the following values and the LED 1 will turn off:

```
Current Phase A: 4 A ∠ 0
Current Phase B: 4 A ∠ -120
Current Phase C: 4 A ∠ 120
Voltage Phase A: 70 V ∠ 0
Voltage Phase B: 0 V ∠ -120
Voltage Phase C: 67 V ∠ 120
```

Solution

The user can create a program shown in *Code Snippet 3*:

Code Snippet 3 prg_LopDetection

```
PROGRAM prg_LopDetection
VAR
    LossOfPotential : PowerSystemProtection.fb_LossOfPotential;
    PhaseAOverVoltage : PowerSystemProtection.fb_OverVoltage;

    Initalized           : BOOL;
END_VAR
```

Code Snippet 3 prg_LopDetection (Continued)

```
IF NOT Initalized THEN
    // Assign all Set inputs and run once to initailize.
    LossOfPotential.SetAmpsNominal := 5;
    LossOfPotential.SetNominalFrequency :=
        SystemTags.Nominal_Frequency.stVal;
    LossOfPotential.SetVoltsNominal := 66.4;
    LossOfPotential.EN := TRUE;
    LossOfPotential();

    PhaseAOverVoltage.ThresholdVoltage := 69;
END_IF

LossOfPotential.FundQok := SEL_prCTPT_1_ECAT.QUALITY_FUND;
LossOfPotential.FundTimeStamp := SEL_prCTPT_1_ECAT.TIMESTAMP_FUND;

LossOfPotential.AmpsNegSeq := SEL_prCTPT_1_ECAT.I2_FUND;
LossOfPotential.AmpsPosSeq := SEL_prCTPT_1_ECAT.I1_FUND;
LossOfPotential.AmpsZeroSeq := SEL_prCTPT_1_ECAT.I0_FUND;

LossOfPotential.AmpsPhaseA := SEL_prCTPT_1_ECAT.IA_FUND;
LossOfPotential.AmpsPhaseB := SEL_prCTPT_1_ECAT.IB_FUND;
LossOfPotential.AmpsPhaseC := SEL_prCTPT_1_ECAT.IC_FUND;

LossOfPotential.VoltsPosSeq := SEL_prCTPT_1_ECAT.V1_FUND;
LossOfPotential.VoltsZeroSeq := SEL_prCTPT_1_ECAT.V0_FUND;

LossOfPotential();

// When loss of potential is detected, the overvoltage element will be
    disabled.
PhaseAOverVoltage.EN := NOT LossOfPotential.LossOfPotential;
PhaseAOverVoltage.MeasuredVoltage := SEL_prCTPT_1_ECAT.VA_FUND.mag;
PhaseAOverVoltage();

IF PhaseAOverVoltage.OverVoltage THEN
    SystemTags.Aux_LED_01.operSet.ctlVal := TRUE;
    SystemTags.Aux_LED_01.operClear.ctlVal := FALSE;
ELSE
    SystemTags.Aux_LED_01.operSet.ctlVal := FALSE;
    SystemTags.Aux_LED_01.operClear.ctlVal := TRUE;
END_IF
```

Detect a Thermal Overload for a Single Phase

Objective

The user would like to detect when Phase A has exceeded a maximum conductor temperature and turn on LED 1.

Assumptions

This example assumes the user has a SEL-2245-42 AC Protection Module configured in the RTAC project to obtain rms measurements.

The *AmbientTemperature* pin values can be forced by the user to observe functionality. Values greater than 20.3°C will cause a thermal overload condition when the *MeasuredCurrent* is set to 100 A.

NOTE: The module is assumed to have the name: "SEL_prCTPT_1_ECAT".

Solution

The user can create a program shown in *Code Snippet 4*:

Code Snippet 4 prg_ThermalElementExample

```
PROGRAM prg_ThermalElementExample
VAR
    ThermalOverload :
        PowerSystemProtection.fb_ConductorThermalOverload :=
        (SetInitialTemp        := 20,
        // This is an unrealistic time constant for a real system,
        // but will demonstrate the behavior of the function block on
        // the order of seconds for this example.
         SetTimeConstant       := T#1S,
         SetRatedCurrent       := 105,
         SetReferenceAmbientTemp := 20,
         SetMaxConductorTemp   := 24);

    // This value can be force changed to view behavior of function
        block.
    AmbientTemp : REAL := 21;
END_VAR
```

```
ThermalOverload.AmbientTemperature  := AmbientTemp;
ThermalOverload.MeasuredCurrent  := SEL_prCTPT_1_ECAT.IA_RMS;
ThermalOverload();

IF ThermalOverload.Overloaded THEN
     SystemTags.Aux_LED_01.operSet.ctlVal  := TRUE;
    SystemTags.Aux_LED_01.operClear.ctlVal  := FALSE;
ELSE
    SystemTags.Aux_LED_01.operSet.ctlVal  := FALSE;
    SystemTags.Aux_LED_01.operClear.ctlVal  := TRUE;
END_IF
```

Release Notes

Version	Summary of Revisions	Date Code
3.5.1.0	➤ Allows new versions of ACSELERATOR RTAC to compile projects for previous firmware versions without SEL IEC types "Cannot convert" messages. ➤ Must be used with R143 firmware or later.	20180619
3.5.0.0	➤ Initial release of time-overcurrent function block. ➤ Initial release of instantaneous-overcurrent function block. ➤ Initial release of loss-of-potential function block. ➤ Initial release of overvoltage function block. ➤ Initial release of undervoltage function block. ➤ Initial release of bus-line voltage check function block. ➤ Initial release of conductor thermal overload function block.	20170517