|  |  |  |
| --- | --- | --- |
| C:\Users\mteama\Desktop\800px-Valeo.svg.png |  | *Valeo Egypt*  *Valeo - GEEDS Business Group*  ***V****isibility &****L****ighting****S****ystems* |

**Validation and Testing  
Generic API User Guide**

|  | **Name** | **Function** | **Date** | **Visa** |
| --- | --- | --- | --- | --- |
| Written By | Mahmoud TEAMA |  |  |  |

Table of Contents

[1 Introduction 4](#_Toc507003243)

[2 Thermal Derating and NTC Defect Libraries 5](#_Toc507003244)

[2.1 Introduction 5](#_Toc507003245)

[2.2 System description and Porting 5](#_Toc507003246)

[2.2.1 Input & Output 5](#_Toc507003247)

[2.2.2 Mapping between the signal and the porting of testing platform: 6](#_Toc507003248)

[2.3 Decision table formulation 6](#_Toc507003249)

[2.4 Building Blocks Libraries 7](#_Toc507003250)

[2.4.1 TherDeratingState 7](#_Toc507003251)

[2.4.2 TherUnderatingState 9](#_Toc507003252)

[2.4.3 TherHystState 11](#_Toc507003253)

[2.5 JoinArray 13](#_Toc507003254)

[2.5.1 Inputs 13](#_Toc507003255)

[2.5.2 Outputs 13](#_Toc507003256)

[2.6 Generic One Test Case API: 14](#_Toc507003257)

[2.7 Use case of using the previous APIs 15](#_Toc507003258)

[2.7.1 Thermal derating scenario 15](#_Toc507003259)

[2.7.2 Specific Example (Java Code) 17](#_Toc507003260)

[2.7.3 NTC Defect Scenario 18](#_Toc507003261)

[3 Power Derating (Gen1): 20](#_Toc507003262)

[3.1 Introduction 20](#_Toc507003263)

[3.2 System Description and Porting 20](#_Toc507003264)

[3.2.1 Input & Output 20](#_Toc507003265)

[3.2.2 Mapping between the signal and the porting of testing platform: 20](#_Toc507003266)

[3.3 Decision table formulation 21](#_Toc507003267)

[3.4 Building Blocks Lib. (Power Derating) 21](#_Toc507003268)

[3.4.1 PwrCalcFn 21](#_Toc507003269)

[3.4.2 PwrDeratingState 22](#_Toc507003270)

[3.4.3 PwrUnDeratingState 23](#_Toc507003271)

[3.4.4 PwrHystState 25](#_Toc507003272)

[3.5 Use Case for power derating: 26](#_Toc507003273)

[3.5.1 Java Script 27](#_Toc507003274)

# Introduction

This Document is considered as a user guide for the generically developed Castle5 Validation API.

This APIs is designed to run throw castle5 (automatic test bench).

The API is divided into several types:

1- Building Blocks API: which is the building block for many test cases.

2- test case API: you only call a single API to perform a whole test case scenario.

3-Tools API: this APIs is used to serve over the previous two categorize (e.g. JoinArray API)

These API is tailored to fit in any project that adopts the same features. It’s developed to be generic and modular enough to be adopted in any similar project

Configuration Management URL: svn://bob1-sv00028/SW\_Tools/Testing\_Platform/Trunk/Testing\_Platform/TestingPlatform/TestingProcLayer/FBDexternalLib

## Impact of Valdiation:

making Generic APIs for validation and Testing which is considered as R&D contribution that saving many hours and used in many future project. And reorienting the tester scope to consider more corner cases and more sophisticated test cases. And he/she will have the appropriate tools to be used in developing those test cases

some of already projects that uses these APIs:

|  |
| --- |
| 1- Ford BBD CD539 |
| 2- Ford FBD CD539 |
| 3- Sierra\_Trio |
| 4-PSA\_PSA-P2X-FBD\_P10 |
| 5-Geely\_NL5\_BBD  6- Geely NL5-P0 |

# Thermal Derating and NTC Defect Libraries

## Introduction

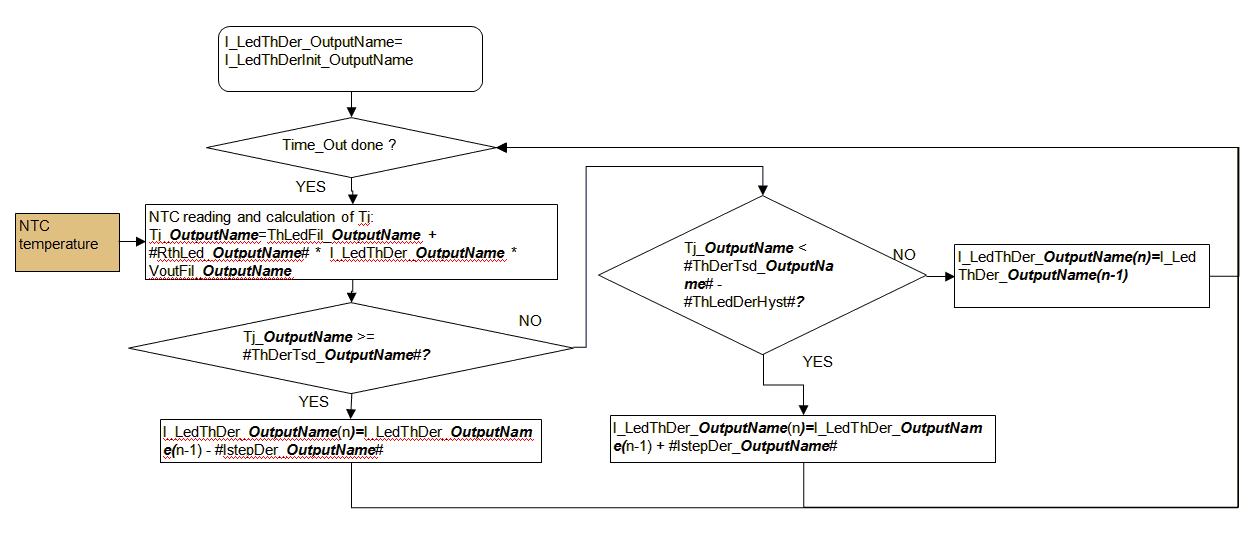
Thermal derating algorithm allows:

* Protecting the LED string against high temperature
* Since it reduces the LED current to maintain longer life

Configuration Management (SVN) URL:  
svn://bob1-sv00028/SW\_Tools/Testing\_Platform/Trunk/Testing\_Platform/TestingPlatform/TestingProcLayer/FBDexternalLib/ThermalDerating.java

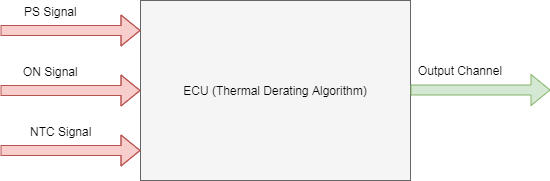
## System description and Porting

The following figure shows the thermal derating algorithm



### Input & Output

The following figure shows the inputs and output of the thermal derating system



#### Signal Description:

##### Inputs:

**1- PS Signal**: this is the power signal that powers up the ECU.

2- **ON Signal**: this is the signal that activates the corresponding function (LED Branch). For example this signal could be a *LIN signal, Digital Signal or power signal* (power signal means that when the volt of an input power channel reaches a certain level the function will be activated).  
e.g. at CD539 Ford BBD project all functions is activated by power signal.

3- **NTC Signal**: this is the input signal that comes from the temperature sensor. The type of the signal is analog. And there’s a mapping between the signal value (in volts) and the corresponding temperature value (in Kelvin or in Celsius).

##### Outputs:

1- **Output Channel**: this is the output signal that the thermal derating algorithm will apply on. For example it could be a *current channel* if the output of the ECU is derating current. Or it could be a *PWM channel* if the output of the ECU is derated value of duty cycle of the PWM signal sent to the driver.

**NOTE**:

The necessity of PS Signal and ON Signal is mainly to active the corresponding function.

### Mapping between the signal and the porting of testing platform:

The ports at the testing platform are written relative to Castle. That is if there’s an input to the ECU. It will be output relative to castle. And vice versa.

|  |  |
| --- | --- |
| Signal Name | Suggested Assigned port |
| PS Signal | Power Output |
| ON Signal | Digital Output, Power output,…etc |
| NTC Signal | DAC Channel |
| Output Channel | Current In Channel, PWM Input,….etc |

## Decision table formulation

The following table describes the descion table formulation

|  |  |  |  |
| --- | --- | --- | --- |
| Inputs | | | Outputs |
| PS Signal | **ON Signal** | **NTC Signal** | **Output Signal** |

## Building Blocks Libraries

This libraries is designed to be a building blocks for a wide variety of test cases considering Thermal derating algorithm.

Each building block can stimulate a certain state of thermal derating algorithm and assert on the output at these state.

**States of thermal derating:**

1. *Derating State:* at which the output is derated with certain characteristics
2. *Underating State*. at which the output is Underated with certain characteristics
3. *Maintain or Hysteresis State*: at which the output is maintained to the previous value.

Each API creates a decision table. Then we used another API (Join Array) to join all the generated Decision Table into one Decision table that can be run later on.

### TherDeratingState

#### Description

This API is simply stimulating the output to be derated by injecting high temperature. The output of the function is a decision table that contains the appropriate input and output. These inputs and outputs is created to inject the high temperature and assert on each step of the derated current.

#### Porting

The porting that’s used is the porting described at the previous section

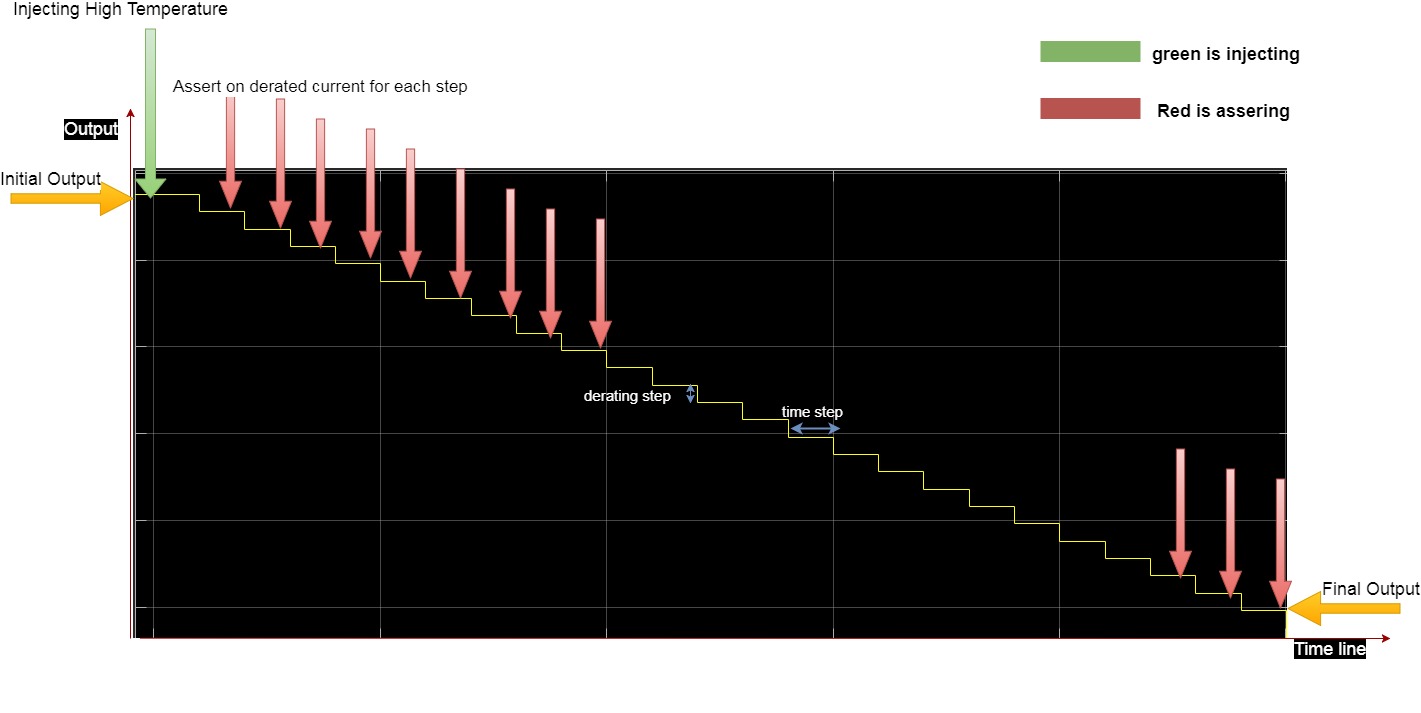
#### Arguments

##### Inputs

|  |  |  |
| --- | --- | --- |
| Parameter | Description | Unit |
| PSNominalVoltage | This is the voltage value required to start the ECU. this value will be place to **PS Signal** mentioned in the previous section | Volt |
| ONSignal | This is the value of the ON signal required to activate the function. e.g. input power or digital input.  This value will be placed to **ON Signal** mentioned in the previous section | The unit will depend on the type of the signal if the signal is input power then the unit will be volt |
| NTCHighTemp | High Temperature to stimulate derating state.  This value will be placed to **NTC Signal** mentioned in the previous section | mVolt |
| Initial\_Output | initial output at which derating will start | If the output is current then the units will be mAmp |
| Final\_Output | This is the final output that the API will stop asserting when reaching this output | If the output is current then the units will be mAmp |
| DeratingStep | It’s the value that the output will increase/decrease each derating/underrating step.  This is a configuration parameter. This parameter is related to the derating algorithm. | If the output is current then the units will be mAmp |
| TimeStep | The period of the step. | msec |

##### Output

The return of the API is a decision table formulated in the form of the porting mentioned in the previous section.



### TherUnderatingState

#### Description

This API is simply stimulate the output to be Underated by injecting Low temperature. The output of the function is a decision table that contains the appropriate input and output. These inputs and outputs is created to inject the Low temperature and assert on each step of the Underated current.

#### Porting

The porting that’s used is the porting described at the previous section

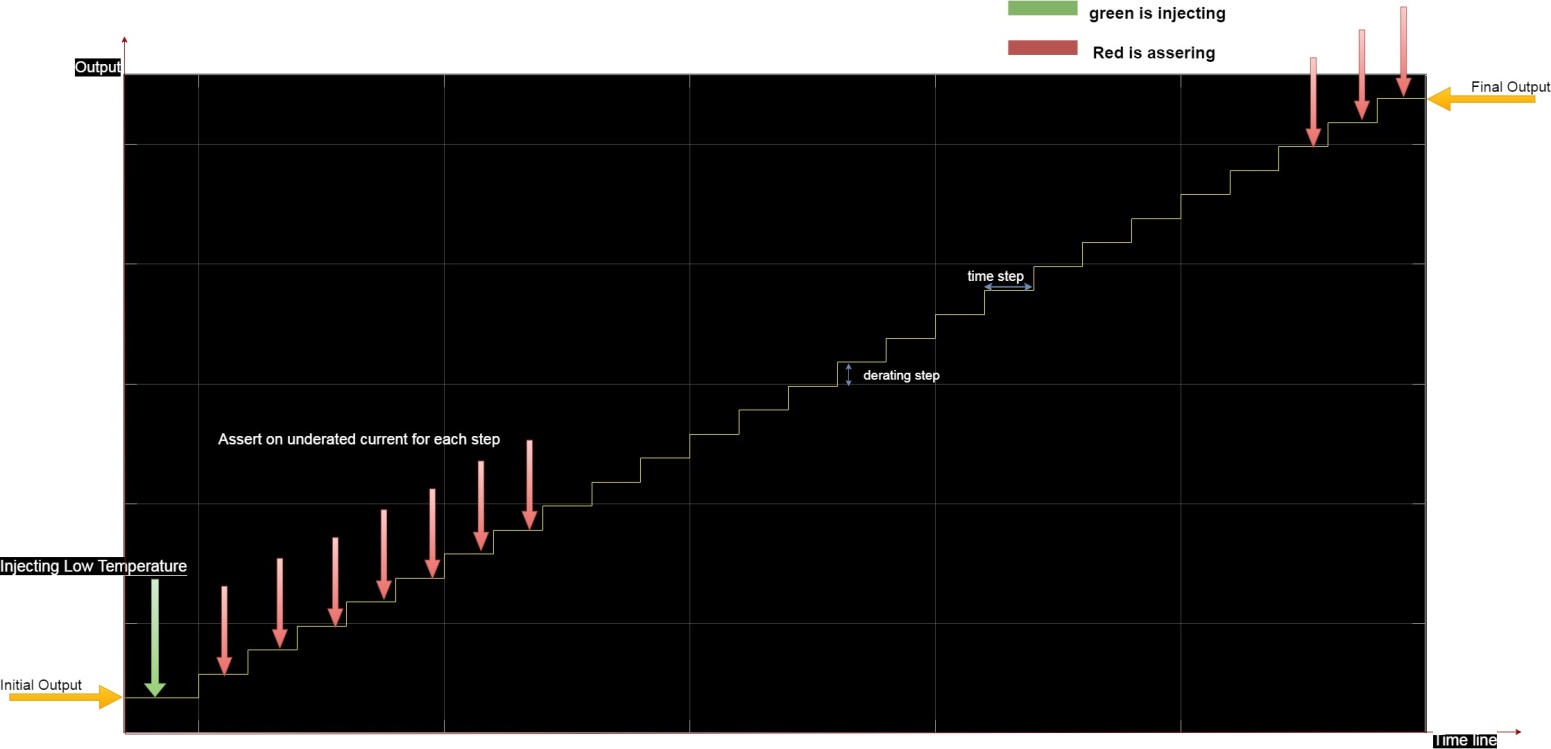
#### Arguments

##### Inputs

|  |  |  |
| --- | --- | --- |
| Parameter | Description | Unit |
| PSNominalVoltage | This is the voltage value required to start the ECU. this value will be place to **PS Signal** mentioned in the previous section | Volt |
| ONSignal | This is the value of the ON signal required to activate the function. e.g. input power or digital input.  This value will be placed to **ON Signal** mentioned in the previous section | The unit will depend on the type of the signal if the signal is input power then the unit will be volt |
| NTCNominalTemp | High Temperature to stimulate Underating state.  This value will be placed to **NTC Signal** mentioned in the previous section | mVolt |
| Initial\_Output | initial output at which Underating will start | If the output is current then the units will be mAmp |
| Final\_Output | This is the final output that the API will stop asserting when reaching this output | If the output is current then the units will be mAmp |
| DeratingStep | It’s the value that the output will increase/decrease each derating/underrating step.  This is a configuration parameter. This parameter is related to the derating algorithm. | If the output is current then the units will be mAmp |
| TimeStep | The period of the step. | msec |

##### Output

The return of the API is a decision table formulated in the form of the porting mentioned in the previous section.



### TherHystState

#### Description

This API is simply stimulate Hysteresis state by injecting temperature in the hyst. zone then wait for sufficient time to make sure that current is settle.

The same API could be used when asserting on the output (e.g. current) when it’s settled down or settled up by replacing hyst. Temperature by the previous temp

#### Porting

The porting that’s used is the porting described at the previous section

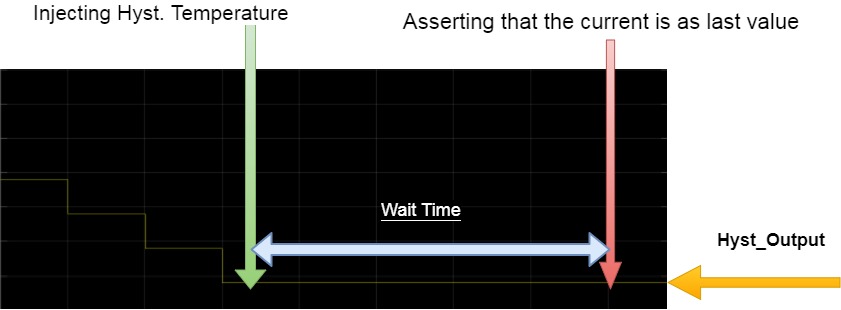
#### Arguments

##### Inputs

|  |  |  |
| --- | --- | --- |
| Parameter | Description | Unit |
| PSNominalVoltage | This is the voltage value required to start the ECU. this value will be place to **PS Signal** mentioned in the previous section | Volt |
| ONSignal | This is the value of the ON signal required to activate the function. e.g. input power or digital input.  This value will be placed to **ON Signal** mentioned in the previous section | The unit will depend on the type of the signal if the signal is input power then the unit will be volt |
| NTCHystTemp | High Temperature to stimulate Hyst  This value will be placed to **NTC Signal** mentioned in the previous section | mVolt |
| Hyst\_Output | settled output at which output will hold | If the output is current then the units will be mAmp |
| wait\_time | The period that is to be waited until asserting on the output | msec |

##### Output

The return of the API is a decision table formulated in the form of the porting mentioned in the previous section.



## JoinArray

This API is designed to join the various decision table generated by the previous API into one decision table that can be run later on

Configuration Management (SVN) URL:

svn://bob1-sv00028/SW\_Tools/Testing\_Platform/Trunk/Testing\_Platform/TestingPlatform/TestingProcLayer/FBDexternalLib/fBDTools.java

### Inputs

The various decision table.

### Outputs

The joined decision table.

## Generic One Test Case API:

Refer to:

<https://docs.google.com/spreadsheets/d/1Lr-FkIZH0-kQ5zhx8Q8mRlnU_wlT9An2KfowkSomraM/edit#gid=0>

## Use case of using the previous APIs

### Thermal derating scenario

One can use the previous API to design a wide variety of test cases.

For example let’s make this scenario.

1- Activate the function in normal state and expect the output to be nominal.

2-Stimulate High Temperature then the output will derate.

3-decrease the temperature to hysteresis zone the function will stop derating and hold the last value for the output.

4- Increase temperature to high temperature again then the function will start to derate again till it reach the lower saturation limit.

5- Wait for appropriate time to make sure that the output is settled down.

6- Decrease temperature to normal range and expect the output to underrate.

7-Increase temperature to hysteresis zone and expect the output to hold the previous value .

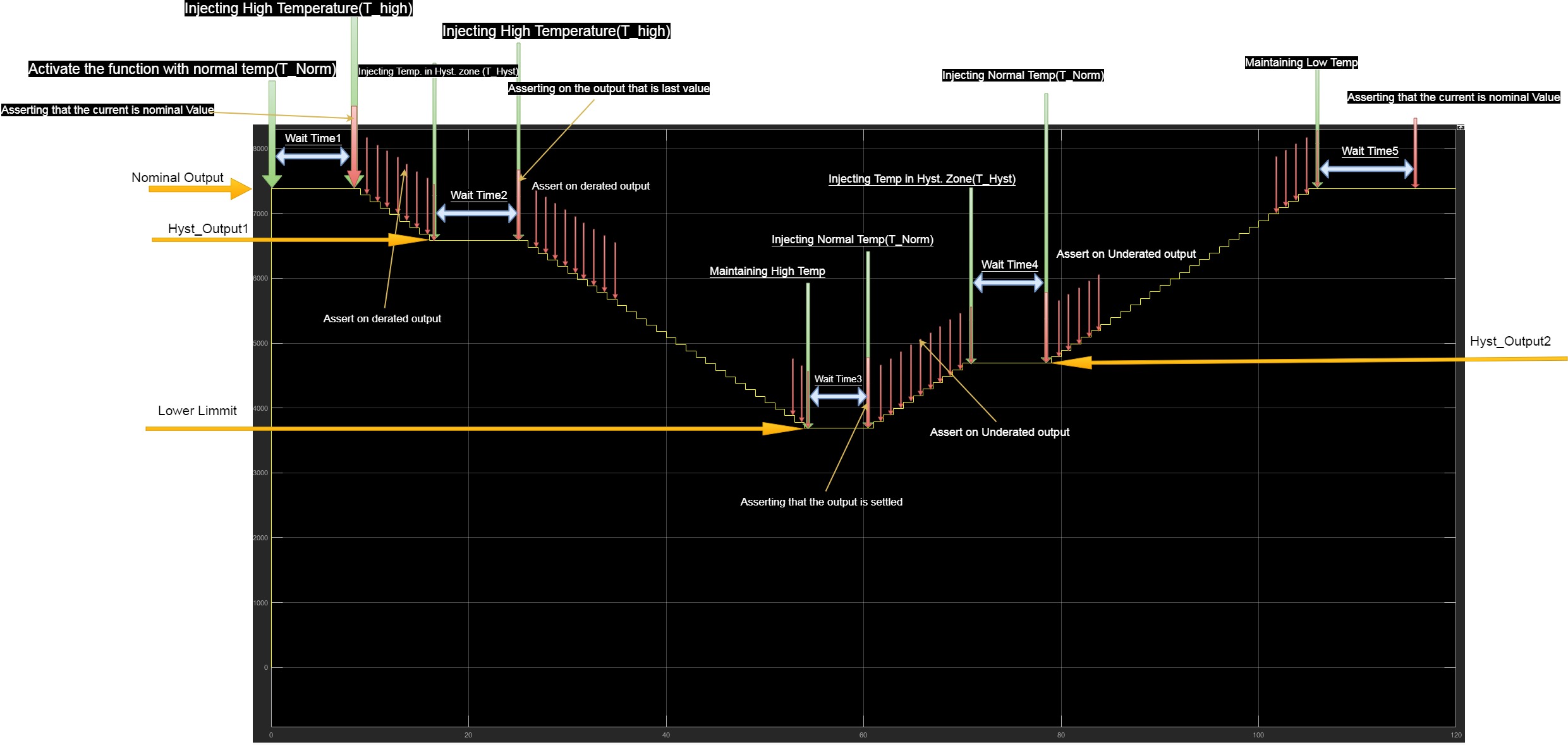
8-Decrease the temperature to normal range and expect output to continue derating until it reach the upper saturation limit i.e. the nominal output value.

9-wait for appropriate period of time and make sure that the output is settled up.

Configuration Management (SVN) URL:

svn://bob1-sv00028/Ford/L411N1601-C\_Ford\_CD539\_Mid-NA/Trunk/L411N1601-C\_Ford\_CD539\_Mid-NA/05-Software\_Tests/03-Validation/06-CASTLE5\_Test\_Projects/P1\_BBD\_P10\_FORD/tests/outputcontrol/LedThermalDerating/DRL\_ThermalDerating\_State.java

#### Graphical Explanation



### Specific Example (Java Code)

This example regarding an FBD project .

The function is DRL. And is activated with power. So the on signal is the same as the PS signal.



### NTC Defect Scenario

Another use case for thermal derating testing is NTC Defect. NTC is the Temperature Sensor for measuring LED temperature. When NTC is defected the SW shall derate the output to its min. value. You may consider this as the worst case.

One can use the building blocks lib. To test this feature also.

**NOTE:** the major difference between testing the thermal derating feature and NTC Defect is that instead of injecting high temperature, we will inject the value of the NTC Defect.

**NOTE:** the reversibility of the NTC defect in the following scenario is Irreversible

**Steps:**

1- Activate the function in normal state and expect the output to be nominal.

2-Stimulate NTC Defect (OC/SC) then the output will derate.

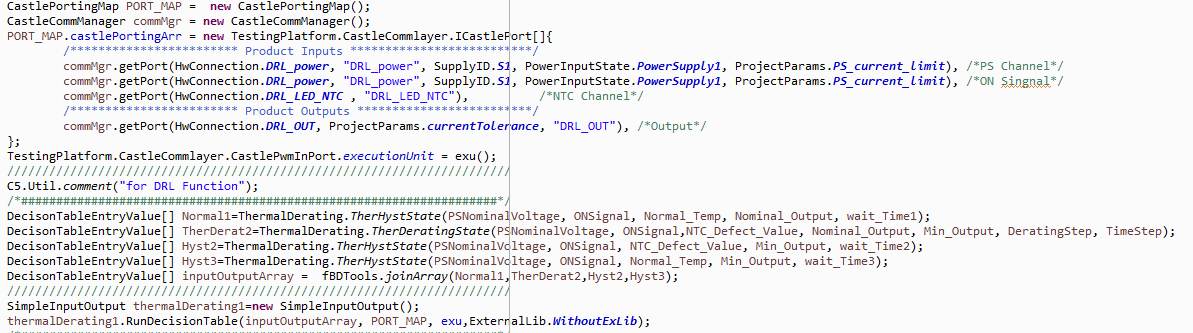
3- Wait for appropriate time to make sure that the output is settled down.

4-Remove NTC Defect by injecting normal NTC Temperature and wait for appropriate time and make sure that the output is still settled down and not underated.

Configuration Management (SVN) URL:

svn://bob1-sv00028/Ford/L411N1601-C\_Ford\_CD539\_Mid-NA/Trunk/L411N1601-C\_Ford\_CD539\_Mid-NA/05-Software\_Tests/03-Validation/06-CASTLE5\_Test\_Projects/P1\_BBD\_P10\_FORD/tests/outputcontrol/LedNTCFailure/DRL\_NTC\_OC\_Build.java

#### JAVA Code(NTC\_Defect)



# Power Derating (Gen1):

## Introduction

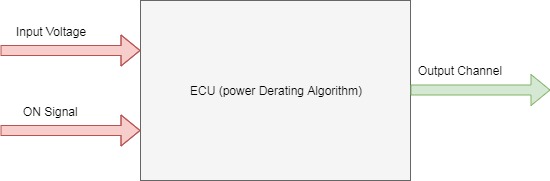
The Power Derating Management allows :

* not to exceed the maximum input current on each power feed of the FBD
* limit the output power of each DC/DC convertor versus the battery voltage.

## System Description and Porting

### Input & Output

The following figure shows the inputs and output of the thermal derating system



#### Signal Description:

##### Inputs:

**1- Input Voltage**: this is the input voltage for the DC/DC Converters.

2- **ON Signal**: this is the signal that activates the corresponding function (LED Branch). For example this signal could be a *LIN signal, Digital Signal or power signal* (power signal means that when the volt of an input power channel reaches a certain level the function will be activated).  
e.g. at CD539 Ford BBD project all functions is activated by power signal.

##### Outputs:

1- **Output Channel**: this is the output signal that the thermal derating algorithm will apply on. For example it could be a *current channel* if the output of the ECU is derating current. Or it could be a *PWM channel* if the output of the ECU is derated value of duty cycle of the PWM signal sent to the driver.

**NOTE**:

The necessity of PS Signal and ON Signal is mainly to active the corresponding function.

### Mapping between the signal and the porting of testing platform:

The ports at the testing platform are written relative to Castle. That is if there’s an input to the ECU. It will be output relative to castle. And vice versa.

|  |  |
| --- | --- |
| Signal Name | Suggested Assigned port |
| Input Voltage | Power Output |
| ON Signal | Digital Output, Power output,…etc |
| Output Channel | Current In Channel, PWM Input,….etc |

## Decision table formulation

The following table describes the descion table formulation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Inputs | | Outputs | | |
| PS Signal | **ON Signal** | |  | **Output Signal** | |

## Building Blocks Lib. (Power Derating)

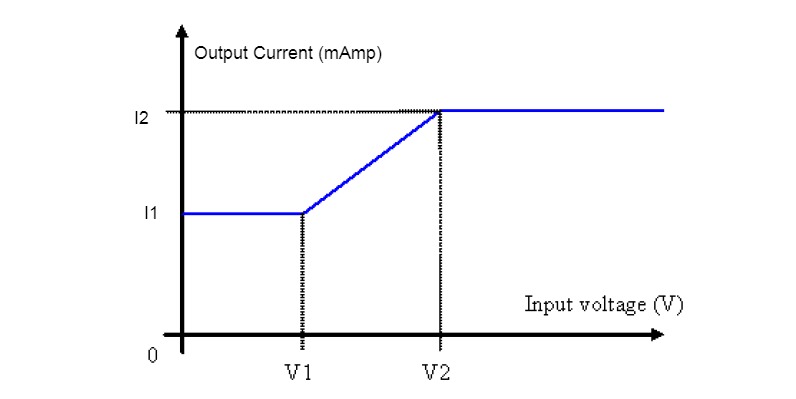
### PwrCalcFn

#### Description

This API is simply calculating the max. allowable output current. The output current is calculated according to the following figure.

The figure is simply a linear relation between the input voltage as the input and the output current as the output but there’s a saturation limit.

**NOTE:** if the input document (CRS) is shown this figure as a relation between the input voltage and the output power. You can simply obtain the output current by simply divide by the forward voltage for each function (LED Branch)



#### Porting

The porting that’s used is the porting described at the previous section

#### Arguments

##### Inputs

|  |  |  |
| --- | --- | --- |
| Parameter | Description | Unit |
| InputVoltage | This is the value of the input of voltage. | Volt |
| BINCurrent | This is the Nominal output current.  This is a configuration parameter | mAmp |
| V1 | Min. volt on curve before saturation (refer to previous figure) | Volt |
| V2 | Max. volt on curve before saturation (refer to previous figure) | Volt |
| I1 | Current Corresponding to V1 "Most properly Min. HW Current " | mAmp |
| I2 | Current Corresponding to V2 "Most properly Max. HW Current " | mAmp |

##### Output

The max allowable output current units(mAmp)

### PwrDeratingState

#### Description

The API is simply stimulate the output to be derated by lowering the input voltage to a point that the max. allowable current is lower than the nominal or existing one.

#### Porting

The porting that’s used is the porting described at the previous section

#### Arguments

##### Inputs

|  |  |  |
| --- | --- | --- |
| Parameter | Description | Unit |
| InputVoltage | This is the value of the input of voltage. | Volt |
| ONSignal | This is the value of the ON signal required to activate the function. e.g. input power or digital input.  This value will be placed to **ON Signal** mentioned in the previous section | The unit will depend on the type of the signal if the signal is input power then the unit will be volt |
| Initial\_Output | initial output at which derating will start | If the output is current then the units will be mAmp |
| DeratingStep | It’s the value that the output will increase/decrease each derating/underrating step.  This is a configuration parameter. This parameter is related to the derating algorithm. | If the output is current then the units will be mAmp |
| TimeStep | The period of the step. | msec |
| BINCurrent | This is the Nominal output current.  This is a configuration parameter | mAmp |
| V1 | Min. volt on curve before saturation (refer to previous figure) | Volt |
| V2 | Max. volt on curve before saturation (refer to previous figure) | Volt |
| I1 | Current Corresponding to V1 "Most properly Min. HW Current " | mAmp |
| I2 | Current Corresponding to V2 "Most properly Max. HW Current " | mAmp |

##### Output

The return of the API is a decision table formulated in the form of the porting mentioned in the previous section.

### PwrUnDeratingState

#### Description

The API is simply stimulate the output to be Underated by increasing the input voltage to a point that the max. allowable current is higher than existing one and lower than the Nominal Current

#### Porting

The porting that’s used is the porting described at the previous section

#### Arguments

##### Inputs

|  |  |  |
| --- | --- | --- |
| Parameter | Description | Unit |
| InputVoltage | This is the value of the input of voltage | Volt |
| ONSignal | This is the value of the ON signal required to activate the function. e.g. input power or digital input.  This value will be placed to **ON Signal** mentioned in the previous section | The unit will depend on the type of the signal if the signal is input power then the unit will be volt |
| Initial\_Output | initial output at which Underating will start | If the output is current then the units will be mAmp |
| DeratingStep | It’s the value that the output will increase/decrease each derating/underrating step.  This is a configuration parameter. This parameter is related to the derating algorithm. | If the output is current then the units will be mAmp |
| TimeStep | The period of the step. | msec |
| BINCurrent | This is the Nominal output current.  This is a configuration parameter | mAmp |
| V1 | Min. volt on curve before saturation (refer to previous figure) | Volt |
| V2 | Max. volt on curve before saturation (refer to previous figure) | Volt |
| I1 | Current Corresponding to V1 "Most properly Min. HW Current " | mAmp |
| I2 | Current Corresponding to V2 "Most properly Max. HW Current " | mAmp |

##### Output

The return of the API is a decision table formulated in the form of the porting mentioned in the previous section.

### PwrHystState

#### Description

This API could be used when asserting on the output (e.g. current) when it’s settled down or settled up by replacing simply wait a certain time then asserting on the output and make sure that the output didn’t change

#### Porting

The porting that’s used is the porting described at the previous section

#### Arguments

##### Inputs

|  |  |  |
| --- | --- | --- |
| Parameter | Description | Unit |
| PSNominalVoltage | This is the value of the input of voltage | Volt |
| ONSignal | This is the value of the ON signal required to activate the function. e.g. input power or digital input.  This value will be placed to **ON Signal** mentioned in the previous section | The unit will depend on the type of the signal if the signal is input power then the unit will be volt |
| Hyst\_Output | settled output at which output will hold | If the output is current then the units will be mAmp |
| wait\_time | The period that is to be waited until asserting on the output | msec |

##### Output

The return of the API is a decision table formulated in the form of the porting mentioned in the previous section.

## Use Case for power derating:

Configuration Management (SVN) URL: svn://bob1-sv00028/GEELY/L281A1605-C\_Geely\_Nl-5\_FBD\_P0/Trunk/L281A1605-C\_Geely\_Nl-5\_FBD\_P0/05-Software\_Tests/03-Validation/06-CASTLE5\_Test\_Projects/Geely\_Nl-5\_FBD\_P0/tests/protection/PWR\_drating/test\_case.java

**Steps:**

1- Activate the function on Normal Voltage and expect Nominal Output.

2-Lower the input voltage so that the function starts to derate then assert on each step until complete derating i.e. the output current reaches a value that corresponds to input voltage from power derating diagram.

3- Wait an appropriate amount of time then assert on the output current that it didn’t change.

4-increase the input voltage again then the output current will start to underate then assert on each step until complete underrating.

5- waint an appropriate amount of time then assert on the output current that’s not change

**NOTE:**

* each derating or underrating API needs the initial output of the output. .
* The initial output for a next stage is the final output of the previous one.
* And to calculate the final output of the previous state we can use PwrCalcFn (mentioned in previous section)
* Then the input of the PwrCalcFn will be the same input voltage feed to previous stage.

### Java Script

### C:\Users\mteama\Desktop\PwrDeratGen1_Java.PNG