|  |  |
| --- | --- |
| Software Development Process Specification for NPe M2 Inverter Development  Prepared By:  KPIT Technologies Ltd.  R.0032999.001, PES - KPIT  Pune, India. |  |
| CONFIG ID : NPe\_M2\_ SDPS  VERSION : 4.3  DATE : 22-Oct-21  Prepared For:  Eaton E-Mobility  Eaton Inc., USA  Eaton E-Mobility |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | | |
|  |  | | |
|  |  |  |  |
|  |
|  |
|  |  | | |

Revision History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ver. | Change Description | Prepared/  Modified By | Approved By | Date |
| 1.0 | Baseline version | Upendra Tambe | Kaushik Shah | 13-Sep-2019 |
| 2.0 | Revision for Scope update V&V, BSW under CR4 | Amol Jarag | Kaushik Shah | 15-Nov-2019 |
| 3.0 | Safety template update.  Merge of ASW and BSW Uniphases in Software.  Change Tracking tool from KAP to Unipro | Sameer Manohar | Kaushik Shah | 13-Feb-2020 |
| 4.0 | Change Req#40   1. Latest Safety Template Update 2. Alignment to VF-1 release Proposal, Schedule etc. and Products. 3. Alignment of Output products as per latest execution and deliveries. Updated output products section for Software, Systems and VNV Uniphases. 4. Updated SDPS to add Product KPI in Appendix-D. BSW product acceptance criteria are defined in agreement with customer. Added a note in Screen Section 5.4.3 – Downstream Products 5. Updated Section 3.2- Added KPI Monitoring SVN path. Added KPI monitoring table is to monitor Product KPI for every release to customer. 6. Updated Section 4.5 - To add Review strategy document to align with PET initiative. The same section is referenced in all Uniphases Process->Activity Description. | Sameer Manohar, Jitendra Darji, Rajiv Bansode | Bimal Tripathi | 13-Oct-2020 |
| 4.1 | Chg Request # 65  SDPS Migration in new template  Update the SDPS for   1. Review Management Strategy 2. Architecture Review by Practice SME 3. Backup Plan for Projects 4. Katapult refinement Parameters Updates. | Jitendra Darji | Bimal Tripathi | 2-Feb-2021 |
| 4.2 | Update the SDPS for   1. Audit dates CR#108 2. WSR Activity Removal 3. team org structure update 4. CM checklist added 5. LIST OF CONFIGURATION ITEMS updated 6. LGM validation reports added 7. GIT Config tool added | Amol Jarag | Bimal Tripathi | 2-Aug-2021 |
| 4.3 | SDPS Updated for :   1. Purpose about safety 2. Scope about safety 3. References: ISO 26262:2011 Road Vehicle-Functional safety updated 4. Project Acceptance criteria 5. Team Organization: Managers for Eaton and KPIT updated 6. Communication plan: Updated for Eaton and KPIT 7. Engineering methods 8. Development and testing Environment 9. Decision Analysis and Resolution 10. Software reuse components 11. Configuration Ids for merged and unmerged documents 12. CR 124 raised for same. | Sukhada More | Bimal Tripathi | 15 Sept 21 |

Table of Contents

[1 Introduction 6](#_Toc86244326)

[1.1 Purpose 6](#_Toc86244327)

[1.2 Scope 6](#_Toc86244328)

[1.3 References 8](#_Toc86244329)

[1.4 Project Acceptance Criteria 8](#_Toc86244330)

[2 Project EPMN 9](#_Toc86244331)

[3 Project Management 10](#_Toc86244332)

[3.1 Management and Control 10](#_Toc86244333)

[3.2 KPI & Measurements 11](#_Toc86244334)

[3.3 Project Schedule 11](#_Toc86244335)

[3.4 People & Skills 11](#_Toc86244336)

[3.5 Team Organization 11](#_Toc86244337)

[3.6 Risk Management 12](#_Toc86244338)

[3.7 Communication Plan 12](#_Toc86244339)

[3.8 Intragroup Coordination 13](#_Toc86244340)

[3.9 Tailoring 14](#_Toc86244341)

[4 Engineering Management 17](#_Toc86244342)

[4.1 Standards 17](#_Toc86244343)

[4.2 Engineering Methods 17](#_Toc86244344)

[4.3 Development and Testing environment 17](#_Toc86244345)

[4.4 Review & Testing Strategy 18](#_Toc86244346)

[4.5 Decision Analysis and Resolution 18](#_Toc86244347)

[4.6 Software Reuse Components 19](#_Toc86244348)

[5 Project Uniphases 20](#_Toc86244349)

[5.1 Planning Uniphase 20](#_Toc86244350)

[5.2 SYSTEM Uniphase 25](#_Toc86244351)

[5.3 FUSA Uniphase 29](#_Toc86244352)

[5.4 HARDWARE Uniphase 42](#_Toc86244353)

[5.5 BSW Uniphase 48](#_Toc86244354)

[5.6 ASW Uniphase 77](#_Toc86244355)

[**5.7** **VNV Uniphase** 83](#_Toc86244356)

[6 Configuration Management 101](#_Toc86244357)

[6.1 Strategy 101](#_Toc86244358)

[6.2 List of Configuration Items 102](#_Toc86244359)

[6.3 Configuration Environment 103](#_Toc86244360)

[6.4 Change Management 105](#_Toc86244361)

[6.5 Problem Management 105](#_Toc86244362)

[7 Quality Assurance 107](#_Toc86244363)

[7.1 QA Strategy 107](#_Toc86244364)

[7.2 QA Activities, R&R 107](#_Toc86244365)

[7.3 Red Flag Reporting Mechanism 108](#_Toc86244366)

[7.4 Timing Plan 109](#_Toc86244367)

[8 EPMN Definitions, Abbreviations and Acronyms 111](#_Toc86244368)

[9 APPENDIX 113](#_Toc86244369)

[9.1 Appendix A: ASIL METHODS IN THE PROJECT 113](#_Toc86244370)

[9.2 Appendix B: EPMN File 117](#_Toc86244371)

[9.3 APPENDIX C: SUPPORTING DOCUMENTS 117](#_Toc86244372)

[9.4 APPENDIX D: Product KPI and Acceptance Goals 118](#_Toc86244373)

[9.5 Appendix E : PMRP 120](#_Toc86244374)

[ 120](#_Toc86244375)

[9.6 APPENDIX F : Acceptance 120](#_Toc86244376)

[ 120](#_Toc86244377)

# Introduction

## Purpose

KPIT Quality Management System and Safety Management System supports dedication to deliver the highest quality engineering products and services to its customers on time by providing the processes for each organization and project.

Software Development Process Methodology (SDPM) is a proprietary methodology used by KPIT for all engineering product development and services. SDPM provides a framework for projects to develop, plan and execute unique and specific processes suitable to individual needs.

<https://sdpm.kpit.com/>

## Scope

|  |
| --- |
| Background:  Eaton has a new business segment called eMobility which brings together Eaton’s Electrical, Transportation and Electrical Power EMEA businesses with parts of Eaton Vehicle business to design, manufacture, market and supply electrical and hybrid xEV solutions for automotive and commercial vehicles  M3 program artifacts will be reused to align to M2 specifications. In this phase of the project, delta requirements between M3 and M2 will be updated and M2 specific new requirements will be added. The scope does not include design/development activities.  SVN path:  [https://gitlab.kpit.com/epowertrain/inverter\_projects/m2\_dualcore\_r2.5/-/blob/master/Program\_Management/KPIT\_Commercial\_Documents/Proposals/M2 Inverter Proposal for 2020\_V10.pdf](https://hjph3svn01.kpit.com/svn/R0032999/Program_Management/KPIT_Commercial_Documents/Proposals/M2%20Inverter%20Proposal%20for%202020_V10.pdf)  for VC proposal refer :  https://gitlab.kpit.com/epowertrain/inverter\_projects/m2\_dualcore\_r2.5/-/blob/master/Program\_Management/KPIT\_Commercial\_Documents/Proposals/KPIT\_Proposal\_Eaton\_NPe\_M2\_Inverter\_Program\_V3.0.pdf  Refer Page 46 to 58 from Proposal – for deliverables list. |

|  |  |
| --- | --- |
| In-Scope:  Responsibility KPIT:  **Electrical Hardware Engineering:**   * Inverter Hardware Architecture (“Control Electronics board” Electronics) * Hardware Safety Analysis/Evaluation * PCB Layout for Control Board * Schematics for Control Electronics * Control board Verification and Testing * Hardware FMEA for Control Electronics Board * Hardware Requirements Management 400 Level for Control Electronics * CPLD hardware and logic design * Hardware Control Electronics and Power Electronics Integration Testing (Joint Responsibility)   **Systems & Requirements Engineering**   * Inverter Module Functional and Safety Requirements Specifications (EEHW, SW, Safety) * System requirement specification - Interface Requirements   Functional and Performance (E/E functions)  Integration & Validation (E/E functions)   * Sub-systems requirements specification - Software, PCB, Power Circuits * System FMEA * Hardware Software Interface specifications * System Fault Reaction Strategy/Plan * CAN signal list   **Functional Safety**   * System FTAs * Technical Safety Requirements and ASIL allocations * Technical Safety Concept * HW Safety Requirements * Safety Review HSIS * HW FMEDA * HW FTA * Software Safety Requirements * SW Safety Analysis * SW Tool Qualification   **ASW and BSW**   * Inverter Software Requirements Specifications (300 Level) * Inverter Software Component Level Requirements Specifications (400 Level) * AUTOSAR compliant Inverter Software Architecture Design (BSW) * Software design with Technical Safety Requirements allocated to software (SSR) * AUTOSAR BSW * AUTOSAR migration for Eaton SWC’s (Eaton specific ASW modules) * Complex Device Drivers * AUTOSAR Software BSW and ASW Integration * AUTOSAR Integration Validation * Inverter “Controller Electronics board” Board HW & SW Integration * Calibration interface using ETAS xETK   **VNV**   * MIL/SIL testing for Model based Inverter Application Software Components (Input processing, output processing and Derating) * BSW PIL testing for Complex Device Drivers * HIL Test Case Design Development * System level basic manual Integration testing (Open loop testing) * Test strategy and Test Cases Development | Out-Of-Scope:  Responsibility Out of Scope KPIT:  **Electrical Hardware Engineering:**   * Inverter Hardware Architecture (Power Electronics) * PCB Printing and Assembly for Control Board * Key Power Electronics components selection * Power Electronics Schematics & PCB Layout * Power Board Verification and Testing * Hardware Requirements Management 400 Level for Power Electronics   **Systems & Requirements Engineering**   * Inverter performance design targets and Motor characteristics * High level design requirements and safety goal * System requirement specification -   Functional and Performance (Non-E/E functions)  Integration & Validation (Non-E/E functions)   * System requirement specification - Data and Modelling requirements Operational requirements Constraints requirements   **Functional Safety**   * Functional Safety Management Plan   **ASW and BSW**   * AUTOSAR compliant Inverter Software Architecture Design (ASW compositions and components along with ECU extract) * AUTOSAR MCAL modules availability for Configuration and integration by KPIT * Application layer software related to torque control & safety related torque demands * Model based Inverter Application Software Components for 3 phase AC motor control, thermal control * Testing Application Layer Functional Software Unit Testing   **VNV**   * Dyno Testing Automation and Test execution * HIL Testing Automation & Test Execution * Vehicle Functional Testing |

## References

|  |  |  |  |
| --- | --- | --- | --- |
| Ref. No. | Reference Title | Version | Date |
| 1 | KPIT M2 Proposal Document | KPIT\_Proposal\_Eaton\_NPe\_M2\_Inverter\_Program\_V1.2 | 07-Aug-2019 |
| 2 | KPIT M2 Proposal Document | KPIT\_Proposal\_Eaton\_NPe\_M2\_Inverter\_Program\_V2.0 | 25-Sep-2019 |
| 3 | KPIT M2 Proposal Document | KPIT\_Proposal\_Eaton\_NPe\_M2\_Inverter\_Program\_V3.0 | 01-Oct-2019 |
| 4 | KPIT M2 Proposal Document | M2 Inverter Proposal for 2020\_V10.pdf | 12-May-2020 |
| 5 | Store Guidelines for Configuration Management | M2 Inverter\_configuration\_management\_plan\_V1.0 | 19-Aug-21 |
| 6 | ISO 26262:2011 Road Vehicle-Functional safety | ISO\_26262\_6\_functional\_safety\_V1.0 | 19-Aug-21 |

## Project Acceptance Criteria

Detailed acceptance criteria also defined in M2 Proposal document.

GIT Path:

<https://gitlab.kpit.com/epowertrain/inverter_projects/m2_dualcore_r2.5/-/blob/master/Program_Management/KPIT_Commercial_Documents/Proposals/M2%20Inverter%20Proposal%20for%202020_V10.pdf> : Slide 45

|  |  |  |
| --- | --- | --- |
| Parameter | Acceptance Criteria | Remarks |
| Fusa Sprint | As per ISO 26262 recommendation |  |
| For all other uniphases | Please refer above link |  |

# Project EPMN

Project lifecycle model is defined for Iterative model.

NPe M2 Inverter Development.



# Project Management

## Management and Control

|  |  |
| --- | --- |
| Responsible Person (Management) | Narendra Saraswat (Overall Program) |
| Ankit Verma, Jaydip Mehta for FUSA |
| Sukhada More, Jaydip Mehta for BSW |
| Sukhada More, Jaydip Mehta for ASW |
| Sukhada More, Jaydip Mehta for Hardware |
| Deepak Mishra for VnV |
| Responsible Person (Functional Safety Consultant) | Manjunatha |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Meeting | Frequency | Mode of Communication/Meeting | Stakeholders | Report |
| MLR-L1 | Weekly | In-Person Meeting | * PL/PM, * PRG.M * PC | MLR-L1 Report |
| MLR-L2 | Weekly | In-Person Meeting | * PRG.M * DM | MLR-L2 Report |
| Screen Meeting | End of Screen | Teams Meeting, Unipro, In-Person Meeting | * Screen Team   PC | Screen Activity Report |
| Daily Standup | Daily | CM repository | * Team members * PM | * MOM on MS Teams |
| MSR | Monthly | In-Person Meeting | * Project Team * PC * PRG.M | * MSR Report |
| Root Cause Analysis | For following instances -Customer reported defects, (Will not be applicable for Test Build releases) only for fin not Low CSAT, Metrics variance, Repetitive defect trends | In-Person Meeting | * Project Team | RCA Report |
| Milestone Kickoff Meeting | Start of the milestone | Teams Meeting , In-Person Meeting | * Project Team * PC * PRG.M * PM | Milestone Kick Off PPT |
| Sub System Sync Up | Weekly or Need Basis | Teams Meeting | * PRG.M * DM * Tech Lead * PM | WSR |
| Quarterly Review | Quarterly | Teams Meeting | * Account Manager * BL | BRM |

## KPI & Measurements

Refer “Project Measurement and Resource Plan” for KPIs/Metrics, Measures and detailed Estimation in below path:

<https://gitlab.kpit.com/epowertrain/inverter_projects/m2_dualcore_r2.5/-/commit/948f53b8bf3baeb7d95405691e325fc7391770eb>

Product KPI (Attached in Appendix) – Product KPI Monitoring Table – to be maintained per release. Below is the path:

<https://gitlab.kpit.com/epowertrain/inverter_projects/m2_dualcore_r2.5/-/commit/948f53b8bf3baeb7d95405691e325fc7391770eb>

## Project Schedule

Refer “Project Measurement and Resource Plan” for work schedules, milestones and list of deliverables in below path :

<https://gitlab.kpit.com/epowertrain/inverter_projects/m2_dualcore_r2.5/-/commit/948f53b8bf3baeb7d95405691e325fc7391770eb>

## People & Skills

Refer “Project Measurement and Resource Plan” for people & skills in below path:

<https://gitlab.kpit.com/epowertrain/inverter_projects/m2_dualcore_r2.5/-/commit/948f53b8bf3baeb7d95405691e325fc7391770eb>

## Team Organization

Safety Team

KPIT

Customer

Scot Streeter (President)

Ranganathan M

Account Manager

Berisch Gabe (Prg.Mgr)

Nishant Tholiya Business Lead

Bimal Tripathi Delivery Manager

Sukhada More

Proj Lead

Sachin Auti

Tech Leader

Prasad Oraskar

Practice SME

Narendra S.

Program Manager

Ankit Verma

Safety Manager

Project Team

Configuration Manager

Rajesh Janwadkar

SBU Head

Tilottamma Kulkarni

Process

Consultant

Sudhir S.

Safety Consultant

AdiNarayan M.

Quality

Shashidarshan Pai

PEG Head

## Risk Management

### Risk Management R&R

The table below describes the risk management process activities and their RASIC. Detailed risk management process is defined in the SDPM.

| Risk Management Activities |  | |  | |  | RASIC | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| DM | Prg.M | | SM | | | PM | TM | CM | SC | PC |
| Identify Risks | R | R | | R | | | R | R | S | S | S |
| Analyze Risk and Plan Risk Response | I | A | | C | | | R | S | C | C | C |
| Perform Mitigation/ Contingency Action | I | A | | I | | | R | S | S | S | S |
| Risk Reporting and Closure | I | A | | I | | | R | I | S | S | S |

### Risk Management Process & Methods

|  |  |
| --- | --- |
| Internal/External Risks | Risk Management Report |
| Frequency | Weekly |
| Detailed Guidelines | Risk Management Guidelines (CGSPGAPA PA 244) |

## Communication Plan

SVN Path:

<https://hjph3svn01.kpit.com/svn/R0032999/Program_Management/KPIT_Quality_Process/SDPS/Communication_Plan.pptx>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| KPIT Contacts | | | | |
| Name of Person | Designation | Contact | Escalation Level | Remarks |
| Narendra Saraswat | Program Manager | [Narendra.Saraswat@kpit.com](mailto:Narendra.Saraswat@kpit.com)  +918055363042  SEZ Unit-2, Plot no- 17, Phase 3, Hinjewadi Rajiv Gandhi Infotech Park, Hinjawadi, Pune, Maharashtra 411057 | Level 1 | Project Level |
| Bimal Tripathi | Delivery Manager | [Bimal.Tripathi@kpit.com](mailto:Bimal.Tripathi@kpit.com)  +91 9987247299  SEZ Unit-2, Plot no- 17, Phase 3, Hinjewadi Rajiv Gandhi Infotech Park, Hinjawadi, Pune, Maharashtra 411057 | Level 2 | Delivery Level |
| Anjana Reddie | BL | Anjana.Reddie@kpit.com  +91 9822477231  SEZ Unit-2, Plot no- 17, Phase 3, Hinjewadi Rajiv Gandhi Infotech Park, Hinjawadi, Pune, Maharashtra 411057 | Level 3 | Business Level |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Customer Contacts** | | | | |
| Name of Person | Designation | Contact | Escalation Level | Remarks |
| Jeff Dunfee II | Project Manager | [JeffSDunfeeII@eaton.com](mailto:JeffSDunfeeII@eaton.com)  26201 Northwestern Hwy, Southfield, MI 48076, USA  [Phone](https://www.google.com/search?rlz=1C1GCEU_enIN844IN844&q=eaton+corporation+southfield+phone&ludocid=16175259936648413909&sa=X&ved=2ahUKEwjBwOWLvcziAhXn7HMBHZomCqIQ6BMwGnoECBIQDg): +1 248-226-6200 | Level 1 | Project Level |
| Yeidei Wang | Program Manager | [YeideiWang@eaton.com](mailto:YeideiWang@eaton.com)  26201 Northwestern Hwy, Southfield, MI 48076, USA  [Phone](https://www.google.com/search?rlz=1C1GCEU_enIN844IN844&q=eaton+corporation+southfield+phone&ludocid=16175259936648413909&sa=X&ved=2ahUKEwjBwOWLvcziAhXn7HMBHZomCqIQ6BMwGnoECBIQDg): +1 248-226-6200 | Level 2 | Program Level |
| Scot Streeter | President, EATON e-Mobility | [ScotMStreeter@eaton.com](mailto:ScotMStreeter@eaton.com)  26201 Northwestern Hwy, Southfield, MI 48076, USA  [Phone](https://www.google.com/search?rlz=1C1GCEU_enIN844IN844&q=eaton+corporation+southfield+phone&ludocid=16175259936648413909&sa=X&ved=2ahUKEwjBwOWLvcziAhXn7HMBHZomCqIQ6BMwGnoECBIQDg): +1 248-226-6200 | Level 3 | Business Level |

* Holiday Calendar

|  |  |  |  |
| --- | --- | --- | --- |
| KPIT | | Customer | |
| Time Zone: IST GMT + 5.30 Hrs | | Time Zone: Eastern Daylight Time GMT - 4:00 Hrs | |
| Holiday | Description | Holiday | Description |
| 01.01.2021, Friday | New Year’s Day | 01.01.2021--Friday | New Year |
| 26.01.2021, Tuesday | Republic Day | 06.01.2021--Wednesday | Epiphany |
| 29.03.2021, Monday | Holi | 02.04.2021—Friday | Good Friday |
| 02.04.2021, Friday | Good Friday | 05.04.2021—Monday | Easter Monday |
| 13.04.2021, Tuesday | Gudi Padwa | 01.05.2021—Saturday | Labor Day |
| 10.09.2021, Friday | Ganesh Chaturthi | 13.05.2021—Thursday | Ascension Day |
| 15.10.2021, Friday | Vijaya Dashami | 24.05.2021—Monday | Whit Monday |
| 04.11.2021, Thursday | Diwali – Laxmi Pooja | 03.06.2021—Thursday | Corpus Christi |
| 05.11.2021, Friday | Diwali - Extended | 01.11.2021—Monday | All Saints Day |
| 27.12.2021, Monday till  31-12.2021, Friday | Global Shutdown |  |  |
|  |  |  |  |

* Internal Communication

|  |  |
| --- | --- |
| Internal Issue Tracking Method | KIT path:  <https://kit.kpit.com/redmine/projects/m2_npe_inverter-program_pes> |
| Project Group Email Address | NA |

* Customer Communication

|  |  |
| --- | --- |
| Communication Frequency | Weekly |
| Communication Tools | Microsoft Teams |
| Delivery/Document Recipient | Vasilios Tsourapas  [vasiliostsourapas@eaton.com](file:///C:\Users\SNEHALD7\AppData\Local\Temp\vasiliostsourapas@eaton.com) |
| Delivery Medium | KPIT SVN, GIT |
| Customer Issue Tracking Method | KIT and/or DNG |
| Communication after closure | [customer-relations@kpit.com](mailto:customer-relations@kpit.com) |
| Other information | NA |

* Meetings

Refer [Section 3.1](#_Management_and_Control)

## Intragroup Coordination

* Travel

Travel related information will be available in the SVN without CR.

SVN Path:

<https://hjph3svn01.kpit.com/svn/R0032999/Program_Management/KPIT_Quality_Process/SDPS/Intragroup_Coordination.xlsx>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| From Date | To Date | Destination | Visa Requirement | No. of People |
| please refer above SVN path |  |  |  |  |

* Procurements

Procurements related information will be available in the SVN without CR.

SVN Path:

<https://hjph3svn01.kpit.com/svn/R0032999/Program_Management/KPIT_Quality_Process/SDPS/Intragroup_Coordination.xlsx>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| From Date | To Date | Item Type | Item Description | Vendor Details |
| please refer above SVN path | - | - | - | - |

* Facilitization

SVN path:

<https://hjph3svn01.kpit.com/svn/R0032999/Program_Management/KPIT_Quality_Process/SDPS/Intragroup_Coordination.xlsx>

|  |  |  |
| --- | --- | --- |
| Type of Facilitization | Facilitization Description | Due Date |
| Please refer above SVN path | - | - |

## Tailoring

There are no tailoring with regard to safety as activities are planned in accordance with the ISO 26262 product development for software which is in KPIT scope, except mapping ISO 26262 activities to KPIT specific different uniphases (FUSA, BSW, V&V)

|  |  |  |  |
| --- | --- | --- | --- |
| Process/Activity | Tailoring Description | Alternate Mechanism/Justification | References |
| Estimation | Estimation Sheet Modification | Current standard estimation templates don’t provide detailed level estimates based on milestone level for many subsystems.  Standard template does not have size measurement “velocity”.  Hence customized template for estimation which includes size template will be used in the Project. | [https://hjph3svn01.kpit.com/svn/R0032999/Program\_Management/KPIT\_Quality\_Process/Templates/Estimation Sheet.xlsm](https://hjph3svn01.kpit.com/svn/R0032999/Program_Management/KPIT_Quality_Process/Templates/Estimation%20Sheet.xlsm) |
| MLR | MLR template modification | Standard template does not have time hours for sprint level for individual resource.  Hence added Time log sheet and Customized training plan will be used to address project needs based on milestones. | <https://hjph3svn01.kpit.com/svn/R0032999/Program_Management/KPIT_Quality_Process/Templates/MLR_L1.xlsx> |
| CM Checklist | Project specific checklist added | To be used for all software releases | <https://gitlab.kpit.com/epowertrain/inverter_projects/m2_dualcore_r2.5/-/blob/master/Program_Management/KPIT_Quality_Process/CM/CM_Release_Checklist.xlsx> |
| MSR | MSR template Modification | Standard template does not have graphical views for effort, schedule, and quality for different subsystems.  Hence added Graphs (Effort View, Quality View, Schedule View, Effort Summary and Sprint Efforts) for each subsystem. | <https://hjph3svn01.kpit.com/svn/R0032999/Program_Management/KPIT_Quality_Process/Templates/Monthly_Status_Report_Excel.xlsx> |
| PAR | WSR (PAR) template modification | Standard template doesn’t have Customer recommended specific information (Milestone Monitor, Milestone Plan, Action Items, Risk, Dependencies, SW-HW Tools) which is added in the customized WSR template. WSR is Not done with Customer. | <https://hjph3svn01.kpit.com/svn/R0032999/Program_Management/KPIT_Quality_Process/Templates/WSR.xlsm> |
|  | Separate WSR (PAR) report for M2 and M3 Program | For M2 and M3 programs Separate WSR (PAR) report will be used for weekly reviews with customer | <https://hjph3svn01.kpit.com/svn/R0032999/Program_Management/KPIT_Quality_Process/Templates/WSR.xlsm>  \*\* From 17th March 21 onwards – Customer has requested to cancel the WSR. Instead share it as a summary on every Wednesday. |
| Technical Review Checklist | Technical review checklists are modified | Project specific points added in the checklist for all uniphases (Planning, BSW, ASW, Hardware, Systems, VnV) | <https://hjph3svn01.kpit.com/svn/R0032999/Program_Management/KPIT_Quality_Process/Templates/Technical_review_checklist> |
| Technical Artifacts Templates  UTP  UTR  ITP  ITR | Usage of project specific/ customer mandated templates checklists etc. | Template Received from customer | <https://gitlab.kpit.com/epowertrain/inverter_projects/m2_dualcore_r2.5/-/tree/master/Program_Management/KPIT_Quality_Process/Templates> artifacts which will be available in the DNG. |
| Uniphase kickoff | Uniphase kick-off mapped with milestone kick-off | Because milestone kick is capturing all required details in Uniphase kickoff | [https://hjph3svn01.kpit.com/svn/R0032999/Program\_Management/KPIT\_Quality\_Process/Templates/Milestone Kick Off.pptx](https://hjph3svn01.kpit.com/svn/R0032999/Program_Management/KPIT_Quality_Process/Templates/Milestone%20Kick%20Off.pptx) |
| Release notes | Software Uniphase | Release note will be shared in the customer provided format. Template Attached. |  |
| HSI document | HSI document is maintained by customer | Inputs from KPIT are provided to customer and will be reviewed by the customer to check the completeness and the adequacy of it for correct control and usage of the hardware |  |
| Synch up meeting | Customer call for discussing the project status | Discuss the project status with client thrice a week. |  |

# Engineering Management

## Standards

Identify all standards with which this activity must comply. These may include KPIT Standards, Customer Specific Standards, Standards Organizations, Regulatory, Government, etc.

|  |  |  |  |
| --- | --- | --- | --- |
| Uniphase | Process | Screen | Store |
| SYSTEM | [Refer](https://hjph3svni02.kpit.com:8443/svn/R0019494/Release/System/Milestone-S5/200_System%20Requirements%20Specification_v5.0.doc) Eaton DNG 200L Systems Requirements.  Section 10.2: PSA Standards and Applicable Standards | KPIT | KPIT |
| FUSA | 1. ISO26262 2. IEC62380 Reliability Handbook | KPIT | KPIT |
| Hardware | Refer Eaton DNG HW Requirements Section :  Refer Section 3.5: Automotive Standards for component selection  Refer Section 1.0: References | KPIT | KPIT |
| BSW | 1. AUTOSAR 4.2.2   C coding standards document | KPIT | KPIT |
| ASW | KPIT Standard | KPIT | KPIT |
| VNV | 1. Guidelines for Software Requirement Analysis 2. Integration Testing Guideline 3. ISO26262-6 4. Unit Testing Guidelines   Refer SVN path:  [https://hjph3svni02.kpit.com:8443/svn/R0033170001/Program\_Management/KPIT\_Quality\_Process/Templates/Technical\_review\_standard\_guidelines/Unit Testing Guidelines.doc](https://hjph3svni02.kpit.com:8443/svn/R0033170001/Program_Management/KPIT_Quality_Process/Templates/Technical_review_standard_guidelines/Unit%20Testing%20Guidelines.doc)  [https://hjph3svni02.kpit.com:8443/svn/R0033170001/Program\_Management/KPIT\_Quality\_Process/Templates/Technical\_review\_standard\_guidelines/Integration testing guidelines.doc](https://hjph3svni02.kpit.com:8443/svn/R0033170001/Program_Management/KPIT_Quality_Process/Templates/Technical_review_standard_guidelines/Integration%20testing%20guidelines.doc) | KPIT | KPIT |

## Engineering Methods

|  |  |
| --- | --- |
| Method | Source |
| Base Software Development using embedded C | MISRA Guidelines |
| AutoSAR | KSAR |
| Functional Safety Process | ISO 26262, ASIL D. ASIL C  LGM stack : QM |

## Development and Testing environment

GIT Path:

<https://gitlab.kpit.com/epowertrain/inverter_projects/m2_dualcore_r2.5/-/blob/master/Program_Management/KPIT_Commercial_Documents/Tools_Quotation/Tools%20and%20Licenses%20need%20for%20NPe%20Program..xlsx>

| No. | Description of Equipment/Software Tools | Version  (if any) | KPIT/  Customer Licensed Software | Quantity | Period | |
| --- | --- | --- | --- | --- | --- | --- |
| From | To |
| 1 | Future Tools updates, please refer above SVN path |  |  |  |  |  |

## Review & Testing Strategy

### Review Strategy

Prepare a review strategy to enable effective and efficient reviews in the project. Review strategy shall be prepared to ensure all defects are captured internally.

Ensure that all aspects of **W**hy, **W**hat, **W**hen, **H**ow, **W**ho and **W**here are planned in the “Review Strategy”.

Reviews will be performed as per the review strategy document. The review strategy document is placed at below SVN location:

|  |
| --- |
| Review Strategy Document |
| <https://hjph3svn01.kpit.com/svn/R0032999/Program_Management/KPIT_Quality_Process/Review_Strategy/M2_ReviewStrategy.docx> |

2nd level Architecture review by Practice SME is planned.

To ensure review effectiveness, below points are taken care:

* Reasonable Effort & Time is provided and spent for reviews
* Reasonable Review defects are estimated and captured during the reviews
* All reported defects are discussed and tracked to closure
* Right skilled reviewers are assigned to perform reviews
* Reviews are tracked for correctness and completeness
* Completed reviews are analyzed for defects summary, trends, RCA, etc.

### Testing Strategy

Following table describers, the strategy for the verification and validation activities to be conducted for this project

<https://gitlab.kpit.com/epowertrain/inverter_projects/m2_dualcore_r2.5/-/commit/c1f685242564f68cae44c25146f7adfaca19d2ec>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Products | V&V Activity | Method/Strategy | Acceptance Criteria | Regression Strategy |
| Refer to the given path |  |  |  |  |

Testing Strategy document is available at below path:

|  |
| --- |
| Testing Strategy Document |
| <https://gitlab.kpit.com/epowertrain/inverter_projects/m2_dualcore_r2.5/-/commit/c1f685242564f68cae44c25146f7adfaca19d2ec> |

## Decision Analysis and Resolution

DAR was performed for Microcontroller selection for Generic Inverter program

<https://gitlab.kpit.com/epowertrain/inverter_projects/m2_dualcore_r2.5/-/commit/6c0b2dd4447cccaccf7e34ebbf92b35b90e413e7>

DAR was a part of single core delivery and will not be delivered further in multicore. For reference we have added in above svn link.

|  |  |  |  |
| --- | --- | --- | --- |
| Sl. No. | DAR Opportunity | Applicable Phase | Remarks |
| 1. | - | - | Kept NA as per the above comment. |

## Software Reuse Components

M2 project have the reusability of M3 BSW modules, MCAL modules, CDD modules, wrapper and RTE. As per ISO guideline the impact analysis is performed for the reusable modules and the modules delta history is maintained with respect to each artifact as per V cycle.

KSAR BSW from KPIT product team is developed as SEooC, it is reused accordingly.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sl. No. | Product Name | Source Project Name/Uniphase | Configuration ID | Remarks (Full / Partial reuse / Others) |
| 1 | 400 Level – S.w | M3- Multicore Implementation for M3 Inverter | Eaton\_NPe\_M3\_M2\_400L | Partial |
| 2 | SAD (Architecture) - SW | M3- Multicore Implementation for M3 Inverter | Eaton\_NPe\_M3\_M2\_SAD | Partial |
| 3 | SDD - SW | M3- Multicore Implementation for M3 Inverter | Eaton\_NPe\_M3\_M2\_SDD | Partial |
| 4 | LLD-SW | M3- Multicore Implementation for M3 Inverter | Eaton\_NPe\_M3\_M2\_LLD | Partial |
| 5 | DFMEA-Fusa | M3- Multicore Implementation for M3 Inverter | Eaton\_NPe\_M3\_M2\_DFMEA | Partial |
| 6 | DFA/FTA-Fusa | M3- Multicore Implementation for M3 Inverter | Eaton\_NPe\_M3\_M2\_DFA/FTA | Partial |
| 7 | Traceability-Sw | M3- Multicore Implementation for M3 Inverter | Eaton\_NPe\_M3\_M2\_TraceabilityS/w | Partial |

|  |  |  |  |
| --- | --- | --- | --- |
| Sl. No. | Product Name | Uniphase | Remarks (Full / Partial reuse / Others) |
| 1 | RTE: Rte bundle version 6.5.0  OS: 3.1.0  Dem: 5.2.2  Fim: 1.3.1  NvM: 4.0.2  Fee: 5.1.0  BswM: 5.1.0  EcuM: 5.0.0  WdgM: 4.4.1  XCP: 3.0.1  CRC: 2.0.0  CryIf: 1.0.0  csm: 1.0.0  COM: 1.4.0 | BSW Uniphase | These are the modules which are reused and modified (customized) as per project requirement |

# Project Uniphases

## Planning Uniphase

|  |  |
| --- | --- |
| Purpose:  Project management plan, high level project plan, and Milestone plan are prepared in this Uniphase. Also, Communication plan, Configuration management, change management, Release management plan documents are prepared and shared with customer. Project plan at a high level will be prepared during this phase and detailed in this document will be prepared and baselined during Planning Uniphase.  KPIT SDPM and ISO 26262:2018 is used for developing the SDPS, which also contains Configuration Management and Quality Assurance activities.  The objective of this Uniphase is to understand the inputs and arrive at a unique and optimum software process for the successful execution of the project and document the same. The purpose of this plan is to provide an essence of the procedures, practices, standards that all the stakeholders will follow to ensure consistent results.  In this phase of the planning, safety plan is also prepared as per DIA and integrated into the software development project plan.  Safety project plan from KSAR BSW Product team and Integration team shall also be available. | |
| Entry Criteria:  Below Artifacts/components are  necessary as Entry criteria   1. Project Proposal PPT 2. Resource Loading sheet 3. Project Kickoff is conducted 4. Development Documents for ex. Development Interface Agreement (DIA) – From Eaton | Exit Criteria:  Below Artifacts/components are  necessary for Exit criteria   1. Change\_Management\_Plan 2. Communication\_Plan 3. Configuration\_Management\_Plan 4. Intragroup\_Coordination.xls 5. Management\_and\_Control\_Plan 6. NPe M2 Inverter Development Project Plan - Milestone.mpp 7. NPe M2 Milestone Plan 8. Release\_Management\_Plan 9. SDPS\_NPe M2 Inverter Development 10. Review Strategy Document 11. Tools and Licenses need for NPe Program.xls |

* EPMN



### Management and Control

Refer [Section 3.1](file:///D:\Projects\R.0032999\Program_Management\KPIT_Quality_Process\SDPS\SDPS_Temp_VF-1\SDPS_NPe%20M2%20Inverter%20Development_v3.1.docx#_Management_and_Control)

### Process

* + **Study Customer Input Document:** The Proposal/SOW and input documents are studied to create SDPS and high-level project plan. There are some key documents created in this Uniphase
  + **Generate SDPS:** Software Development Process Specification is generated in this activity. That covers all aspects of the project execution
  + **Template and checklist modification:** Process checklist and templates will be modified as per custom project need
  + **Template and checklist review and rework:** Review and rework will be done as per custom prepared checklist and templates.
  + **Review of SDPS:** The SDPS document is reviewed with reference to Proposal/SOW details.
  + **Rework of SDPS:** According to the review comments, rework shall be done.
  + **Define Milestone Plan:** This is a milestone definition document that describes all customer and KPIT milestones with inputs, process, outputs, and acceptance criteria
  + **Define Release Management Plan:** Release management plan is created in this activity. This describes the process and actions for release of any deliverable
  + **Define Communication Plan:** Communication plan describes organization structure for KPIT and Eaton System, the SPOC persons, escalation mechanism
  + **Define Tool and License List**: in SVN. List will be updated based on changes.
  + Review of Communication Plan: Since this document describes the organization structure of KPIT and customer for the project, the single point of contact, escalation matrix, communication flow and important contact details will be assured through effective review.
  + **Rework on Communication Plan:** Rework shall be carried out after receiving review comments.
  + **Define Configuration Management Plan:** This document describes the storage and configuration details
  + **Review of Configuration management Plan:** Configuration management plan is reviewed during this phase. The storage, sharing and exchange mechanism for project artefacts is reviewed
  + **Rework of Configuration management Plan:** The configuration management plan is reworked after review comments are received
  + **Define Change Management Plan:** This document describes the change process and CR classification details
  + **Review of Change management Plan:** Change management plan is reviewed during this phase.
  + **Rework of Change management Plan:** The change management plan is reworked after review comments are received
  + **Prepare Intragroup Coordination sheets:** Travel, Procurements, Hiring and Facilitation details will be mentioned if applicable.
  + **Review Intragroup Coordination sheets:** Review Travel, Procurements, Hiring and Facilitation details
  + **Rework Intragroup Coordination sheets:** Rework Travel, Procurements, Hiring and Facilitation details
  + **Define Management & Control mechanism:** Management and control mechanism is defined during this activity. M&C PPT will be base lined after this is base lined. This document covers SDPS updates, metrics and project closure during Uniphase
  + **Review of Management and Control Plan:** Management and control plan is reviewed during this activity. It is necessary to assure that information is collected from other elements and necessary support is provided to project team, to maintain stability across all the elements of Uniphase
  + **Rework on Management and Control plan:** The Management and control plan is reworked, corrected and baselined after review comments are received
  + **Preparation of Master plan:** Project plan is generated in this activity. That covers all aspects of the project execution
  + **Review of Master Plan:** Project plan is reviewed during this phase. The storage, sharing and exchange mechanism for project artefacts is reviewed
  + **Rework of Master Plan:** The project plan is reworked after review comments are received
  + **Preparation of Estimations:** Project plan is generated in this activity. That covers all aspects of the project execution
  + **Review of Estimations:** Project plan is reviewed during this phase. The storage, sharing and exchange mechanism for project artefacts is reviewed
  + **Rework of Estimations:** The project plan is reworked after review comments are received
  + Perform SDPS screen with SME, PC, QM
  + After fixing defects from review, conduct Screen with external experts (Upstream and Downstream reviewers)
  + Take DM approval
  + Upload baseline version of SDPS on SVN

### Screen

|  |  |
| --- | --- |
| Type of Screen | Readiness Screen |

* Products

|  |  |
| --- | --- |
| List of Upstream Products | 1. Project Proposal PPT 2. Resource Loading sheet |
| List of Downstream Products | 1. Configuration Management Plan 2. Release Management Plan 3. Change Management Plan 4. Management and Control Plan 5. Communication Plan 6. Intragroup\_Coordination 7. NPe M2 Inverter Development Project Plan – Milestone.mpp 8. Npe M2 Daul core Milestone Plan 9. NPe\_M2\_ SDPS Development\_Notes 10. Tools and Licenses need for NPe Program.xls 11. Project Measurement Resource Plan 12. Project Measurement and Resource Planning document (PMRP) |
| List of Horizontal Products | 1. SDPS\_ INVERTER DESIGN 2. Configuration Management Plan 3. Release Management Plan 4. Change Management Plan 5. Management and Control Plan 6. Communication Plan 7. Intragroup\_Coordination 8. PSA Inverter Development Project Plan – Milestone.mpp 9. Tools and Licenses need for NPe Program.xls 10. Milestone Plan 11. SDPS\_ INVERTER DESIGN Notes 12. Estimation 13. ITP\_Template 14. ITR\_Template 15. FuSa - HARA - Technical Review Checklist 16. FuSa - Tool Qualification - Technical Review Checklist |

* Screen Scope

Upstream View:

SDPS document prepared is correct, consistent and complete in accordance with the Proposal/SOW, Customer inputs and Other KPIT Inputs Guidelines, Templates and Checklists are created as appropriate.

Downstream View:

The SDPS, Templates, Guidelines and Review checklists can be followed for the subsequent Uniphase of project execution. The activity notes are detailed enough for each Uniphase to execute the project.

Quality and Configuration Manager view:

The products are as per the KPIT Screen guidelines for Quality and CM View. Ensure SDPS Notes file is completed in all respects.

### Store

I/O/P/I - Input Product / Output Product / Pre-requisite Product / Intermediate Product

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| I/O/P/I | Product Name | CI Identifier | Source | Destination |
| I/-/-/- | Project Proposal PPT | KPIT\_Proposal\_Eaton\_NPe\_M2\_Inverter\_Program\_V2.0 | KPIT | Planning Uniphase |
| -/O/-/- | \*SDPS\_ INVERTER DESIGN | Eaton\_NPe\_M2\_SDPS\_INVERTER DESIGN | Planning Uniphase | All the Uniphases |
| -/O/-/- | \*Configuration Management Plan | Eaton\_Npe\_M2\_Dual\_core \_ \_Config\_Management\_Plan | Planning Uniphase | All the Uniphases |
| -/O/-/- | \*Release Management Plan | Eaton\_Npe\_M2\_Dual\_core \_ Release\_Management\_Plan | Planning Uniphase | All the Uniphases |
| -/O/-/- | \*Change Management Plan | Eaton\_Npe\_M2\_Dual\_core \_ Change\_Management\_Plan | Planning Uniphase | All the Uniphases |
| -/O/-/- | \*Management and Control Plan | Eaton\_Npe\_M2\_Dual\_core \_ Management\_Control\_Plan | Planning Uniphase | All the Uniphases |
| -/O/-/- | \*Communication Plan | Eaton\_Npe\_M2\_Dual\_core \_ Communication\_Plan | Planning Uniphase | All the Uniphases |
| -/O/-/- | Intragroup\_Coordination | Eaton\_Npe\_M2\_Dual\_core \_ Intragroup\_Coordination | Planning Uniphase | All the Uniphases |
| -/O/-/- | \*PSA Inverter Development Project Plan – Milestone.mpp | Eaton\_Npe\_M2\_Dual\_core \_ Inverter Development Project Plan – Milestone.mpp | Planning Uniphase | All the Uniphases |
| -/O/-/- | \*Tools and Licenses need for NPe Program.xls | Eaton\_Npe\_M2\_Dual\_core \_ Tools and Licenses need for NPe Program.xls | Planning Uniphase | All the Uniphases |
| -/O/-/- | \*Milestone Plan | Eaton\_Npe\_M2\_Dual\_core \_ PSA Production Milestone Plan | Planning Uniphase | All the Uniphases |
| -/O/-/- | \*SDPS\_ INVERTER DESIGN Notes | Eaton\_Npe\_M2\_Dual\_core\_SDPS\_ INVERTER DESIGN\_Notes\_001 | Planning Uniphase | All the Uniphases |
| -/O/-/- | Estimation | Eaton\_Npe\_M2\_Dual\_core \_Estimation | Planning Uniphase | All the Uniphases |
| -/O/-/- | ITP\_Template | Eaton\_Inveter\_ITP\_Template | Planning Uniphase | VNV Uniphase. |
| -/O/-/- | ITR\_Template | Eaton\_Inveter\_ITR\_Template | Planning Uniphase | VNV Uniphase. |
| -/O/-/- | FuSa - HARA - Technical Review Checklist | Eaton\_Inveter\_FuSa - HARA - Technical Review Checklist | Planning Uniphase | FuSa Uniphase |
| -/O/-/- | FuSa - Tool Qualification - Technical Review Checklist | Eaton\_Inveter\_FuSa - Tool Qualification - Technical Review Checklist | Planning Uniphase | FuSa Uniphase |
| -/O/-/- | FuSa\_Tool Qualification\_Template | Eaton\_Inveter\_FuSa\_Tool Qualification\_Template | Planning Uniphase | FuSa Uniphase |
| -/O/-/- | PMRP document | Eaton\_Npe\_M2\_Dual\_core \_PMRP\_Document. | Planning Uniphase | All Uniphases |
| -/O/-/- | ^Review Strategy Document | Eaton\_Npe\_M2\_Dual\_core Review\_Strategy\_Document | Planning Uniphase | All Uniphases |
| I/-/-/- | Release Notes Template(Customer Provided) | Eaton\_Npe\_M2\_Dual\_core Release\_Notes Template | Planning Uniphase | BSW Uniphase |

\* Incremental Product

Prefix \* if product is incremental. E.g.: \*Design Document

X indicates next version for the given artefact.

## SYSTEM Uniphase

|  |  |
| --- | --- |
| Purpose:  The purpose of this uniphase is to perform following activities  During Systems Uniphase, the study and understanding of input document, prepare M2 NPe/PSA Requirements Specifications (200 Level), Fault Reaction Plan (200 level). Analyze Hardware-Software Interface Specifications (300 level) and Requirement Validation (Partial, 300 level). All work products shall be reviewed and base lined for other sub-systems to use. Whenever there is any change, these documents shall be changed accordingly and baselined again. | |
| **Entry Criteria:**  Below Artifacts/components are necessary to start with SYSTEM uniphase   1. Project Proposal PPT 2. SDPS\_INVERTER DESIGN 3. Work Package document –(100 Level Requirement Specification) | Exit Criteria:  Below Artifacts/components are necessary to close SYSTEM uniphase   1. System Requirement Specification (200 Level) 2. System Design Architecture (HW and SW architecture) 3. PCB Requirement 4. Power circuit requirement 5. CAN Signal List (PSA/NPe) 6. System Fault reaction Plan 7. HSIS (Eaton) 8. FMEA /System FTA |

* EPMN



### Management and Control

Refer [Section 3.1](file:///D:\Projects\R.0032999\Program_Management\KPIT_Quality_Process\SDPS\SDPS_Temp_VF-1\SDPS_NPe%20M2%20Inverter%20Development_v3.1.docx#_Management_and_Control)

### Process

* Activity Description:

o Previous milestone Feedback and estimation of current milestone: This includes previous milestone delivery feedback and estimation to be carried out for next milestone.

o Update 300 PCB Requirements: Which includes updates on HW PCB Interface/ Functional & Performance/ Integration & Validation Requirement and HW (PCB) Safety Requirements

o System FMEA: Perform System FMEA based on Fault list.

o 200-Level System Requirement to PSA Specific: Update System requirement document to address PSA specific requirements

o Review of previous Milestone: perform the review of current Milestone for all deliverables with Technical Reviewer. Fix the comments from reviewer with Author.

o Organize Technical Screen – Initiate the Screen for all deliverables with external Screener, PC & CM.

o Perform release: Once screen is closed then get SAR Report & Release Checklist from PC. Store all deliverables in SVN with as per Configuration management.

### Screen

|  |  |
| --- | --- |
| Type of Screen | Technical Screen |

* Products

|  |  |
| --- | --- |
| List of Upstream Products | 1. Project Proposal PPT 2. SDPS\_NPe M2 Inverter Development 3. Work package document 4. Hardware Technical Safety Requirement 5. CM Checklist |
| List of Downstream Products | 1. NA |
| List of Horizontal Products | 1. SDPS\_ INVERTER DESIGN 2. Technical review Checklists 3. Project Plan 4. Systems Requirements 5. ^PCB Requirement 6. ^FMEA 7. Fault reaction Plan 8. ^Power circuit requirement 9. High Level SW Requirement |

* Screen Scope

Upstream View:

System Requirement Specification prepared is correct, consistent and complete in accordance with the 100-level requirement specification.

Traceability will be available from System Requirement Specification to 100 level requirement specification in the DNG.

Downstream View:

In this view, product will be verified and validated for correctness and completeness whether it can be used as input for the next phase.

Quality and Configuration Manager view:

The products are as per the KPIT Screen guidelines for Quality and CM view

The product will follow CM guidelines for release process as well as with CM audit process.

### Store

I/O/P/I - Input Product / Output Product / Pre-requisite Product / Intermediate Product

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| I/O/P/I | Product Name | CI Identifier | Source | Destination |
| I/-/-/- | Project Proposal PPT | KPIT\_Proposal\_Eaton\_NPe\_M2\_Inverter\_Program\_V2.0 | KPIT | Systems Uniphase |
| I/-/-/- | \*SDPS\_ INVERTER DESIGN | Eaton\_Inverter\_SDPS\_ INVERTER DESIGN | Planning Uniphase | System Uniphase |
| I/-/-/- | Work package document | Eaton\_Inverter\_100 Level work packages | System Uniphase | System Uniphase |
| I/-/-/- | Hardware Technical Safety Requirement | Eaton\_Inverter\_Updated HW TSR | FuSa Uniphase | System Uniphase |
| -/O/-/- | Systems Requirements | NPe\_M3\_M2\_Dual\_Core \_200\_Level\_System\_Requirement | System Uniphase | Client |
| -/O/-/- | ^PCB Requirement | NPe\_M3\_Dual\_Core \_300\_Level\_PCB\_requirement | System Uniphase | Client |
| -/O/-/- | ^FMEA | Eaton\_Inverter\_FMEA\_Report | System Uniphase | Client |
| -/O/-/- | Fault reaction Plan | NPe\_M3\_M2\_Dual\_Core\_Systems\_Fault \_Reaction\_Plan | System Uniphase | Client |
| -/O/-/- | ^Power circuit requirement | NPe\_M3\_Dual\_Core \_300\_Level\_PowerCircuit\_requirement | System Uniphase | Client |
| -/O/-/- | High Level SW Requirement | NPe\_M3\_M2\_Dual\_Core\_300\_Level\_HighLevel SW\_requirement | System Uniphase | Client |

Note: 300 level requirement was there in KPIT scope till M2 VF milestone.

\* Incremental Product

Prefix \* if product is incremental. E.g.: \*Design Document

Provide the SVN path where the work product stored (SDPS)

<https://hjph3svn01.kpit.com/svn/R0032999/Program_Management/KPIT_Quality_Process/SDPS>

## FUSA Uniphase

|  |  |
| --- | --- |
| **Purpose:**  The purpose of this uniphase is to perform following activities:   * To derive updated technical safety requirements based on PSA piloted system requirements * To derive System FTA as per PSA requirements * Prepare hardware board and control board FMEDA * Hardware software specification updates as per FMEDA and software safety analysis * Prepare software tool qualification report   **Tool Evaluation and Qualification subphase:**  The purpose of this subphase is the following:   * To evaluate tool confidence level of the listed tools in table below (evaluation) * To perform the qualification if applicable * To come up with the usage guidelines for optimum use of the software tools   **Software Requirements subphase:**  The purpose of this subphase is to perform following activities:   * To derive SW Safety Requirements (SSR) and non-Safety Requirements (300 Level as per scope of the project) specifications based on System Requirements including TSR (200 Level), | |
| **Entry Criteria:**  Below Artifacts/components are  necessary as Entry criteria   1. Project Proposal PPT 2. SDPS\_ INVERTER DESIGN 3. 100 & 200 Level Requirement Specification 4. Technical review checklists.(Per ISO26262)   **Tool Evaluation and Qualification subphase:**   1. SDPS\_ INVERTER DESIGN 2. List of tools 3. User manual and/or Safety user manual for the software tool (from an external source) 4. Environment and Constraints of the software tool (from an external source) 5. ISO26262 specifications (mainly Part 8)   **Software Requirements subphase:**   1. Product specifications (100 Level) 2. System Requirements including TSR (200 Level) 3. High Level SW Requirements (300Level) 4. Technical Safety Concept (TSC) 5. System Design Architecture 6. HSIS 7. AUTOSAR Requirements (if applicable) 8. ISO26262 specifications (mainly Part 6) 9. ISO14229 specifications 10. ISO15765 specifications | **Exit Criteria:**  Below Artifacts/components are necessary for Exit criteria   1. FSMP 2. TSR (200 Level) 3. System Design Architecture 4. System FTA (includes SW, HW) 5. HSR (300 Level) 6. SSR (300 Level) 7. SW Architecture (ASW+RTE+BSW) 8. SW FTA 9. HW PCBA Architecture 10. BSW Safety Requirements (KSAR BSW (includes RTE), CDD, MCAL) (400 Level) 11. SW Component/Module Design 12. Software Safety Analysis (DFMEA, DFA, SW- FTA) 13. FMEDA 14. HW FTA 15. Updated HSIS 16. SW Tool document 17. SW Tool Qualification   **Tool Evaluation and Qualification subphase:**   1. Software tool criteria evaluation report 2. Software tool qualification report 3. Software tool usage guidelines   **Software Requirements subphase:**   1. SW Requirements including SSR (300 Level) 2. HSIS (refined) |

* EPMN

Diagram

Description automatically generated

### Management and Control

Refer [Section 3.1](file:///D:\Projects\R.0032999\Program_Management\KPIT_Quality_Process\SDPS\SDPS_Temp_VF-1\SDPS_NPe%20M2%20Inverter%20Development_v3.1.docx#_Management_and_Control)

### Process

* Activity Description

Here provide the milestone wise release activity

• Activity Description

F4.1 Milestone:

o Functional Safety Management Plan: Develop/update Safety Management Plan as per DIA

o Updated Technical Safety Requirements: Derive or update TSRs for NPe FSRs and Technical Safety Concept as per NPe FSC

o Updated System FTA: Update System FTA based on FSR received from NPe

o Review of Milestone F4.1: perform the review of F4.1 Milestone for all deliverables with Technical Reviewer. Fix the comments from reviewer with Author.

o Organize Technical Screen – Initiate the Screen for all deliverables with external Screener, PC & CM.

o Perform release: Once screen is closed then get SAR Report & Release Checklist from PC. Store all deliverables in SVN with as per Configuration management.

F4.2 Milestone:

o Updated Controller Board HW FMEDA: Perform control board HW FMEDA and find SPF an LFM values

o Power Board FMEDA: Perform power board HW FMEDA and find SPF an LFM values

o HW FTA: Perform HW FTA and derive PMHF values

o Updated HSRs and SSRs: Update and Derive HSR and SSR from TSRs

o Review of Milestone F4.2: perform the review of F4.2 Milestone for all deliverables with Technical Reviewer. Fix the comments from reviewer with Author.

o Organize Technical Screen – Initiate the Screen for all deliverables with external Screener, PC & CM.

o Perform release: Once screen is closed then get SAR Report & Release Checklist from PC. Store all deliverables in SVN with as per Configuration management.

F5 Milestone

o SW Safety Analysis: Prepare initial analysis for Safety analysis of SW Architecture Design

o HSIS Updated as per HW FMEDA: Perform Safety updates and review to HSIS

o SW Tools Documentation: Initiate Software Tool Qualification and Collect SW Tools Documentation

o Review of Milestone F5: perform the review of F5 Milestone for all deliverables with Technical Reviewer. Fix the comments from reviewer with Author.

o All Technical Reviews will be performed as per review strategy document (Refer sect. 4.5)

o Organize Technical Screen – Initiate the Screen for all deliverables with external Screener, PC & CM.

o Perform release: Once screen is closed then get SAR Report & Release Checklist from PC. Store all deliverables in SVN with as per Configuration management.

F6 - Milestone

o HSIS- Updated as per SW Safety analysis: Perform SW Safety analysis and Updated HSIS as per SW Safety analysis findings

o SW Tool Qualification Report: Perform SW Tool Qualification and generate report

o Review of Milestone F6: perform the review of F6 Milestone for all deliverables with Technical Reviewer. Fix the comments from reviewer with Author.

o Organize Technical Screen – Initiate the Screen for all deliverables with external Screener, PC & CM.

o Perform release: Once screen is closed then get SAR Report & Release Checklist from PC. Store all deliverables in SVN with as per Configuration management.

All Subsequent milestones post F6:

o All Technical Reviews will be performed as per review strategy document (Refer sect. 4.5)

**Tool Evaluation and Qualification :**

The objective of this subphase is for below listed tools,

to provide criteria to determine the required level of confidence in a software tool (tool evaluation) when applicable,

to provide means for the qualification of the software tool when applicable, in order to create evidence that the software tool is suitable to be used to support the activities or tasks required by the ISO 26262 series of standards (i.e. the user can rely on the correct functioning of a software tool for those activities or tasks required by the ISO 26262 series of standards),

if, applicable come up with the usage guidelines for optimum use of the software tools.

List of tools and languages to be used in project:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl. No.** | **Name** | **Version** | **Tool / Language** | **Uniphase/Subphase Identification** |
| 1 | C | - | Language | - |
| 2 | Assembly | - | Language | - |
| 3 | IBM Rational DOORS | 9.5.2 | Tool | Requirements |
| 4 | Enterprise Architect | 12.0.1208 | Tool | Software Design |
| 5 | Compose for K-SAR | 16.5.2\_alpha8 | Tool | Software Testing, Unit Testing, Integration Testing |
| 6 | Polyspace Code Prover | 2013.5.4 | Tool | Software Unit Design and Implementation |
| 7 | QAC | 8.1.1R | Tool | Software Unit Design and Implementation |
| 8 | IBM Rational Test Real Time | 8.0.1 | Tool | Unit Testing, Integration Testing |
| 9 | SVN (TorstoiseSVN : Subversion for Windows) | 1.13.1 Build 28686 - 64 bit | Tool | All |
| 10 | Redmine | 3.3.8 | Tool | All |
| 11 | Unipro | 2.2.0.0 | Tool | All |
| 12 | Notepad++ | 7.8.7 | Tool | Implementation |
| 13 | Microsoft Office | 2013 | Tool | All |
| 14 | IBM RTC | 5.0.2 | Tool | All |
| 15 | GCC MinGW compiler | 3.4.2 | Tool | Software Testing, Unit Testing, Integration Testing |
| 16 | CYG Gnu Pro GDB | 5.0 | Tool | Software Testing, Unit Testing, Integration Testing |
| 17 | Output Comparator | 1.0.1 | Tool | Unit Testing, Integration Testing |
| 18 | KPIT Agile Planning (KAP) tool | 3.3 | Tool | All |
| 19 | Jira | 8.5.4#05005 | Tool | All |
| 20 | Beyond Compare | 4.3.3 | Tool | All |
| 21 | Tasking compiler | 6.2r2 | Tool | Software Testing, Unit Testing, Integration Testing ( VNV uniphase) |
| 22 | Cantata | 9.0.3 | Tool | Unit Testing, Integration Testing ( VNV uniphase) |
| 23 | Git | 13.1.2 | Tool | All |
| 24 | Active Perl | 5.8.6 | Tool | Software Testing, Unit Testing, Integration Testing( VNV uniphase) |
| 25 | TRACE 32 | vR2016.02.000072893 | Tool | Software Testing, Unit Testing, Integration Testing ( VNV uniphase) |
| 26 | GHS Compiler | 6.1.4 | Tool | Software Testing, Unit Testing, Integration Testing( VNV uniphase) |
| 27 | Green Hills Probe V3 | 5.0.8 | Tool | Software Testing, Unit Testing, Integration Testing( VNV uniphase) |
| 28 | Medini Analyzer | 3.1.2.87457 | Tool | Requirements( Systems Uniphase) |
| 29 | GHS Compiler Multi | 7.1.4 | Tool | Software Testing, Unit Testing, Integration Testing( VNV uniphase) |
| 30 | Timing architect |  | Tool | Timing analysis. , Architeture phase |

* Activity Description

**Perform Software Tool Evaluation:**

During this activity, the software tools will be evaluated to determine the tool confidence level (TCL). To determine the required level of confidence in a software tool used within development the following criteria are evaluated:

1. the possibility that the malfunctioning software tool and its corresponding erroneous output can introduce or fail to detect errors in a safety-related item or element being developed (Tool Impact TI);

2. And the confidence in preventing or detecting such errors in its corresponding output (Tool error Detection TD)

To evaluate the confidence in prevention or detection measures, measures internal to the software tool (e.g. monitoring) as well as measures external to the software tool (e.g. **guidelines, tests, reviews**) implemented in the development process of safety element are considered.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Tool Detection** | **TD1**  (High degree of confidence in prevention or detection measures. Methods: process steps, redundancy or rationality checks in tool itself) | **TD2**  (Medium degree of confidence in prevention or detection measures.) | **TD3**  (detection possible randomly only) |
| **Tool Impact** | **TI1**  (No possibility that Safety element will be impacted by malfunction of Tool) | TCL1 | TCL1 | TCL1 |
| **TI2**  (Safety element will be impacted by malfunction of Tool) | TCL1 | TCL2 | TCL3 |

**Table 1: Determination of TCL**

**(Ref: Table 3 ISO26262:8-11)**

For software tool following shall be considered and available for the evaluation:

1. Version
2. User manual
3. Features
4. Configuration
5. Intended use
6. inputs and expected outputs
7. Environmental and functional constraints
8. Known issues and workarounds

**Perform qualification of Software Tools if required:**

During this activity, if indicated by the determined tool confidence level (TCL2 or TCL3) then appropriate qualification methods are applied to comply with ASIL D. Otherwise (TCL1) there is no need to apply such qualification methods.

| **Sl.No.** | **Methods** | **ASIL D**  **(TCL3/TCL2)** | **Method selection or argumentation** |
| --- | --- | --- | --- |
| 1a | Increased confidence from use in accordance with 11.4.7 | +/+ | NA |
| 1b | Evaluation of the tool development process in accordance with 11.4.8 | +/+ | NA |
| 1c | Validation of the software tool in accordance with 11.4.9 | ++/++ | If TCL comes to be 2 or 3 this method will be applicable and either KPIT or vendor will perform the same. |
| 1d | Development in accordance with a safety standard - a | ++/+ | KPIT will collect a statement from vendor stating that development is as per Safety standard. |
| a No safety standard is fully applicable to the development of software tools. Instead, a relevant subset of requirements of the safety standard can be selected.  EXAMPLE Development of the software tool in accordance with ISO 26262, IEC 61508, EN 50128 or RTCA DO-178C. | | | |

**Table 2: Methods for Tool Qualification**

**(Ref: Table 4/5 ISO26262:8-11)**

**Confirmation Review & Rework of evaluation and qualification report of software Tool:**

During this activity, the Evaluation and qualification reports of software tools will be circulated to the Safety Consultant for review. The aim of the confirmation review will be Confirmation of the correct evaluation of the required level of confidence in the software tool(s) used in the development of a safety element and evaluation of the qualification of the software tool(s), considering the required level of confidence of the respective software tools.

**Prepare Tool Usage Guidelines for software Tools:**

In this activity The Tools which are identified for the project, their usage guidelines will be prepared for the correct and optimum use of the tool.

**Review and Rework Tool Usage Guidelines:**

During this activity, Tool guidelines prepared for software tools will be reviewed by expert team members. The review comments for the Tool Usage Guidelines will be incorporated in Tool Usage Guideline documents.

**Updates to Safety Case**

During this activity Safety Case document will be updated for Tool evaluation/qualification and their usage guideline derived in the above activity.

**Software Requirements subphase:**

* Activity Description

The objective of this activity is to prepare SW Requirement specifications. Following documents as shall be referred:

* Product specifications (100 Level)
* System Requirements including TSR (200 Level)
* Technical Safety Concept (TSC)
* System Design Architecture
* HSIS
* AUTOSAR Requirements (if applicable)
* ISO26262 specifications
* ISO14229 specifications
* ISO15765 specifications

**Study Inputs and Prepare Software Requirement Specification document:**

Derive the SW Requirement document, which contains the requirements applicable to KPIT.

In this activity, Software Requirement Document will be prepared. Software Requirement Documents shall include all the requirements coming from various inputs. In preparation of SRS all the specified input documents will be analysed and requirements will be captured. ASIL allocation from TSR shall be considered and the SW is to be developed as per respective ISO26262 ASIL compliance.

Requirements shall be mentioned in natural language and semi-formal notations.

For each requirement following attributes shall be present:

1. Requirement ID (e.g. generated from DNG)
2. Requirement Category (Requirement Type)
3. Safety Classification/ASIL (Safety/Safety Integrity Level)
4. Requirement status (Requirement Rational/Coverage/Defect/Validation conclusion)
5. Remarks

Appropriate selection of methods as per table below shall be followed for specification of software requirements.

| **Sl. No.** | **Methods** | **ASIL A** | **ASIL B** | **ASIL C** | **ASIL D** | **Argumentation** |
| --- | --- | --- | --- | --- | --- | --- |
| 1a | Natural language | ++ | ++ | ++ | ++ | English Language shall be used. |
| 1b | Informal notations - a, b | ++ | ++ | + | + | English Language shall be used. |
| 1c | Semi-formal notations - a, b, c, d | + | + | ++ | ++ | AUTOSAR standard notations shall be followed. |
| 1d | Formal notations - a | + | + | + | + | Not considered |
| a An appropriate selection of methods for the specification of safety requirement considers their adequacy to achieve the characteristics of safety requirement according to 6.4.2 for a specific issue to be specified, its complexity or the knowledge of the persons specifying or managing safety requirements. Examples include the use of state graphs or diagrams for specifying the complex behavior of software or hardware including many states or/and complex transitions.  b For higher level safety requirements (e.g. safety goals, functional and technical safety requirements) natural language and other types of informal notations are the most appropriate forms, though some requirements may be better handled with semi-formal notations.  c Semi-formal notation formulates requirements using natural language that is supplemented by mathematical or graphical elements such as equations, graphs, diagrams, flow charts, timing diagrams, and many other forms of representation (e.g. UML® and SysML™). Examples include model-based techniques and applying templates and controlled vocabulary for requirement sentences in natural language.  d For lower-level safety requirements where precise hardware and software behaviors and capabilities may be specified, semi-formal notations are more appropriate due to greater clarity. However, even here it may not be possible or necessary to use semi-formal techniques for every requirement. | | | | | | |

**Table 3 Specifying safety requirements**

**(Ref: Table 1 ISO26262:8-6)**

Requirement Management strategy document mentioned in the link below shall be referred for the same and traceability (in DNG) between Requirements shall be provided.

[DNG link for Requirement Management strategy](https://loutcsvrtcp01.napa.ad.etn.com:9443/rm/web#action=com.ibm.rdm.web.pages.showArtifactV2&artifactURI=https%3A%2F%2Floutcsvrtcp01.napa.ad.etn.com%3A9443%2Frm%2Fresources%2F_9Uy4IdumEeiQR8CGNEJSYg&vvc.configuration=https%3A%2F%2Floutcsvrtcp01.napa.ad.etn.com%3A9443%2Frm%2Fcm%2Fstream%2F_DPrZtlSFEeiSR9QaYprQmA&componentURI=https%3A%2F%2Floutcsvrtcp01.napa.ad.etn.com%3A9443%2Frm%2Frm-projects%2F_DKdigVSFEeiSR9QaYprQmA%2Fcomponents%2F_DPiPwlSFEeiSR9QaYprQmA)

**Review & Rework SRS document:**

During this activity, Requirements will be verified by Review and inspection (Screen) as described in “Verification and Validation Strategy” section 4.5 of this document. SRS document will be reviewed against review checklist by expert team members and SRS document will be updated based on review comments. There will be two reviews conducted based on two separate checklist one is technical review checklist and second is Safety checklist which is based on the ISO26262-8:6. SSR shall undergo Safety review.

| **Sl. No.** | **Methods** | **ASIL A** | **ASIL B** | **ASIL C** | **ASIL D** | **Argumentation** |
| --- | --- | --- | --- | --- | --- | --- |
| 1a | Verification by walk-through | ++ | + | o | o | NA |
| 1b | Verification by inspection | + | ++ | ++ | ++ | Review, Inspection (Screen). |
| 1c | Semi-formal verification - a | + | + | ++ | ++ | Review, Inspection (Screen). |
| 1d | Formal verification - a | o | + | + | + | NA |
| a Verification can be supported by executable models. | | | | | | |

**Table 5 Methods for the verification of safety requirements**

**(Ref: Table 2 ISO26262:8-6)**

**Refine/Update HSI Document**

Hardware dependent modules shall refine Hardware software interface specifications (HSIS) in this activity. HSIS will be detailed down to a level allowing the correct control and usage of hardware and shall describe each safety-related dependency between hardware and software.

Inputs from KPIT are provided to customer and HSI document is maintained by customer.

**Review & Rework of HSIS (refined)**

In this activity HSI inputs provided by KPIT will be reviewed by the customer to check the completeness and the adequacy of it for correct control and usage of the hardware.

**Update Safety Case**

During this activity Safety Case document will be updated for SSR and HSIS (refined) derived in the above activity. This includes capturing evidence of safety activities carried out during this subphase.

### Screen

|  |  |
| --- | --- |
| Type of Screen | Technical Screen |

* Products

|  |  |
| --- | --- |
| List of Upstream Products | 1. Project Proposal PPT 2. SDPS\_ INVERTER DESIGN 3. 100 Level Requirement Specification   **Tool Evaluation and Qualification subphase:**   1. SDPS\_ INVERTER DESIGN 2. List of tools 3. User manual and/or Safety user manual for the software tool (from an external source) 4. Environment and Constraints of the software tool (from an external source)   **Software Requirements subphase:**   1. Product specifications (100 Level) 2. System Requirements including TSR (200 Level) 3. Technical Safety Concept (TSC) 4. System Design Architecture 5. HSIS 6. AUTOSAR Requirements (if applicable) 7. ISO26262 specifications 8. ISO14229 specifications 9. ISO15765 specifications |
| List of Downstream Products | NA |
| List of Horizontal Products | 1. ^FSMP 2. ^TSR 3. ^System FTA 4. ^HSR 5. SSR (Software Safety Requirements) 6. ^FMEDA 7. ^HW FTA 8. ^Updated HSIS 9. ^SW Tool document 10. ^SW Tool Qualification 11. SW DFMEA 12. SW FTA 13. SW DFA 14. SW Safety Analysis |

* Screen Scope

Upstream View:

Functional safety artifacts will be created as per system requirements based on PSA.

HSR and SSR will be traced to TSR in the system requirement specifications in the DNG.

Downstream View:

The Functional safety artifacts can be followed for the subsequent Uniphase of project execution

Quality and Configuration Manager View:

The products are as per the KPIT Screen guidelines for Quality and CM View.

The product will follow CM guidelines for release process as well as with CM audit process

Tool Evaluation and Qualification subphase:

Upstream View:

* + Tools which are to be used in the entire project duration are covered
  + User manual, configuration details, known issues are analysed
  + Tool version mentioned and available for the project are matching
  + Tools are evaluated and qualified, if required, correctly
  + Tool Usage Guidelines are updated as appropriate

Downstream View:

* + Tools which are to be used in the entire project duration are evaluated correctly
  + Tool Usage Guidelines are understandable and optimal

Quality and Configuration Manager View:

The products are as per the KPIT Screen guidelines.

**Software Requirements subphase:**

Upstream View:

**SRS:**

The screener shall check if all the inputs required for SRS are correctly, completely identified and the correct/latest version of requirement documents are analyzed.

The screener shall also ensure that all the open points (logged in DNG) references, AUTOSAR SWS references (if applicable) and TSR specification references are correct and complete.

SW Requirements and SSR will be traced to System Requirements and TSR respectively in the system requirement specifications in the DNG.

**HSIS**:

The screener shall check the completeness of the HSIS w.r.t required interaction for software to work correctly.

Downstream View:

**SRS:**

The screener shall check the usability of the SRS for the next Software Architecture subphase preparation of the project.

The screener shall ensure that the SRS items are without any ambiguous statements. Also if there are any open points still requiring clarification from the customer, the screener shall ensure that those open points are raised in DNG for the same.

Screener shall also ensure that the responses from the customer on open points in DNG are updated in the SRS document.

**HSIS:**

The screener shall check if HSIS derived is sufficient for the implementation of software.

Quality and Configuration Manager View:

The products are as per the KPIT Screen guidelines for CM and Quality view.

The product will follow CM guidelines for release process as well as with CM audit process.

### Store

< I/O/P/I - Input Product / Output Product / Pre-requisite Product / Intermediate Product >

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| I/O/P/I | Product Name | CI Identifier | Source | Destination |
| I/-/-/- | Project Proposal PPT | KPIT\_Proposal\_Eaton\_NPe\_M2\_  Inverter\_Program\_V1.2 | KPIT | FUSA Uniphase |
| I/-/-/- | \*SDPS\_ INVERTER DESIGN | Eaton\_NPe\_M2\_Dual\_Core\_SDPS | Planning Uniphase | FUSA Uniphase |
| I/-/-/- | DIA Document | Eaton\_NPe\_M3\_M2\_Dual\_Core\_Eaton\_KPIT\_DIA  \*M2 specific documents will be identified and tagged to M2 project id. Ex : Impact Analysis | FuSa Uniphase | FUSA Uniphase |
| I/-/-/- | Functional Safety Requirement | Eaton\_Inverter\_Functional Safety Concept and Requirement | Customer | FUSA Uniphase |
| I/-/-/- | Fault reaction Plan | NPe\_M3\_M2\_Dual\_Core\_Systems\_Fault \_Reaction\_Plan | Systems Uniphase | FUSA Uniphase |
| I/-/-/- | SW Safety Analysis | Eaton\_NPe\_M3\_M2\_Dual\_Core\_SW Safety Analysis Report | Systems Uniphase | FUSA Uniphase |
| I/-/-/- | Systems Requirements (200L ) | NPe\_M3\_M2\_Dual\_Core \_200\_Level\_System\_Requirement | Systems Uniphase | FUSA Uniphase |
| I/-/-/- | High Level SW Requirement (300L) | NPe\_M3\_M2\_Dual\_Core\_300\_Level\_HighLevel SW\_requirement | Systems Uniphase | FUSA Uniphase |
| I/-/-/- | CAN Signal List | NPe\_M3\_M2\_Dual\_Core \_CAN\_Signal\_List | Systems Uniphase | FUSA Uniphase |
| -/O/-/- | ^FSMP | Eaton\_Inverter\_ Functional\_Safety\_Management\_Plan | FUSA Uniphase | Customer |
| -/O/-/- | ^TSR | Eaton\_Inverter\_Technical\_Safety\_requirement | FUSA Uniphase | Customer |
| -/O/-/- | ^System FTA | Eaton\_Inverter\_System\_FTA | FUSA Uniphase | Customer |
| -/O/-/- | ^HSR | Eaton\_Inverter\_Hardware\_Safety\_requirement | FUSA Uniphase | Customer |
| -/O/-/- | SSR (Software Safety Requirements) | Eaton\_NPe\_M3\_M2\_Dual\_Core\_SW \_Software\_Safety\_Requirements | FUSA Uniphase | Customer |
| -/O/-/- | ^FMEDA | Eaton\_Inverter\_HW\_FMEDA | FUSA Uniphase | Customer |
| -/O/-/- | ^HW FTA | Eaton\_Inverter\_HW\_FTA | FUSA Uniphase | Customer |
| -/O/-/- | ^Updated HSIS | Eaton\_Inverter\_HSIS as per FMEDA | FUSA Uniphase | Customer |
| -/O/-/- | ^SW Tool document | Eaton\_Inverter\_SW\_Tools\_Documentation | FUSA Uniphase | Customer |
| -/O/-/- | ^SW Tool Qualification | Eaton\_Inverter\_SW\_Tools\_Qualification | FUSA Uniphase | Customer |
| -/O/-/- | SW DFMEA | Eaton\_Inverter\_SW\_DFMEA | FUSA Uniphase | Customer |
| -/O/-/- | SW FTA | Eaton\_Inverter\_SW\_FTA | FUSA Uniphase | Customer |
| -/O/-/- | SW DFA | Eaton\_Inverter\_SW\_DFA | FUSA Uniphase | Customer |
| -/O/-/- | SW Safety Analysis | Eaton\_NPe\_M3\_M2\_Dual\_Core\_SW Safety Analysis Report | FUSA Uniphase | Customer |

\* Incremental Product

Prefix \* if product is incremental. E.g.: \*Design Document

<https://hjph3svn01.kpit.com/svn/R0032999/Program_Management/KPIT_Quality_Process/SDPS>

**Tool Evaluation and Qualification subphase:**

I/O/P/I - Input Product / Output Product / Pre-requisite Product / Intermediate Product

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| I/O/P/I | Product Name | CI Identifier | Source | Destination |
| I/-/-/- | \*SDPS\_ INVERTER DESIGN | Eaton\_Inverter\_SDPS\_ INVERTER DESIGN | Planning | Customer |
| -/O/-/- | **^**Tool(s) usage guideline document(s) | Eaton\_Inverter\_TOOL\_QUAL\_ REPORTS | Tool Evaluation and Qualification | Customer |
| -/O/-/- | **^**Tool Evaluation Report | Eaton\_Inverter\_TOOL\_EVAL\_ REPORTS | Tool Evaluation and Qualification | Customer |
| -/O/-/- | **^**Tool Qualification Report | Eaton\_Inverter\_TOOL\_QUAL\_ REPORTS | Tool Evaluation and Qualification | Customer |

\* Incremental Product

Prefix \* if product is incremental. Eg: \*Design Document

^ Product applicable / Delivered earlier in M2 Single Core(Inverter\_Design) and the SDPS in which it is mentioned is stored at : SVN Path – Updated as below

<https://hjph3svn01.kpit.com/svn/R0032999/>

**Software Requirements subphase:**

I/O/P/I - Input Product / Output Product / Pre-requisite Product / Intermediate Product

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| I/O/P/I | Product Name | CI Identifier | Source | Destination |
| I/-/-/- | \*SDPS\_ INVERTER DESIGN | Eaton\_Inverter\_SDPS\_ INVERTER DESIGN | Planning | Customer |
| I/-/-/- | Product specifications | 100L Product specs. | Customer | Customer |
| I/-/-/- | TSR | Eaton\_Inverter\_Technical\_Safety\_requirement | Customer | Customer |
| I/-/-/- | System Requirement specification | Eaton\_Inverter\_200\_Level\_System\_Requirement | Customer | Customer |
| I/-/-/- | TSC | Eaton\_Inverter\_TSC | Customer | Customer |
| I/-/-/- | System Design Architecture | Eaton\_Inverter\_System\_Design\_Architecture | Customer | Customer |
| I/-/-/- | HSIS | Eaton\_Inverter\_HSIS | Customer | Customer |
| -/O/-/- | Software Requirement Specification\* | Eaton\_NPe\_M3\_M2\_Dual\_Core \_ High Level Software Requirement | Software Requirements | Customer |
| -/O/-/- | SSR | Eaton\_NPe\_M3\_M2\_Dual\_Core \_ High Level Software Requirement | Software Requirements | Customer |

\* Incremental Product

Prefix \* if product is incremental. Eg: \*Design Document

## HARDWARE Uniphase

|  |  |
| --- | --- |
| **Purpose:**  All hardware development activities for Eaton Inverter shall be performed during this Uniphase. Based on system requirements and selected microcontroller, the detailed circuit shall be designed. BOM creation, PCB design and prototype testing are also considered during this Uniphase.  **Note: Hardware uniphase was part of single core and no more a uniphase in multicore except 400L CPLD requirements, and HW Testplan and Test report.** | |
| **Entry Criteria:**  Below Artefacts/components are necessary as Entry criteria.   1. Project Proposal PPT 2. SDPS\_ INVERTER DESIGN 3. System requirements 4. Hardware Specifications document 5. Test Software | **Exit Criteria:**  Below Artefacts/components are necessary as Exit criteria.   1. Circuit Schematics 2. Bill of Material 3. Proto Board Testing Report 4. HW FMEDA 5. 400 Level Requirement 6. HW test Plan |

* EPMN



### Management and Control

Refer [Section 3.1](#_Management_and_Control)

### Process

* + Based on system requirements select microcontroller design Architecture and define hardware requirement documents, Organize Technical Screen and Perform Release
  + Prepare, review and rework the hardware test plan, Organize Technical Screen and Perform Release
  + Estimate the components required for schematic design, Organize Technical Screen and Perform Release
  + Design of schematic along with worst case analysis based on hardware architecture, Organize Technical Screen and Perform Release
  + Review schematic design and implementing review and screen comments in schematic, Organize Technical Screen and Perform Release
  + Generate BOM from the schematic
  + Review BOM and update based on review comments, Organize Technical Screen and Perform Release
  + Update the hardware specification in case any update in BOM, Organize Technical Screen and Perform Release
  + Prepare PCB layout based on hardware architecture
  + Review PCB Layout design and implementing review, Organize Technical Screen and Perform Release
  + Prepare hardware test setup for functional testing
  + Review Test report and implementing review and screen comments in test report
  + Board bring up and functional testing of the hardware developed
  + Preparation of test results, review and rework, Organize Technical Screen and Perform Release
  + Prepare Hardware FMEA for each module.
  + All Technical Reviews will be performed as per review strategy document (Refer sect. [4.5](#_Testing_(Verification_and))
  + Review and Rework on Hardware FMEA, Organize Technical Screen and Perform Release
  + Prepare and define 400 level Hardware Requirement for each module
  + Review and Rework 400 level Hardware Requirement, Organize Technical Screen and Perform Release

### Screen

|  |  |
| --- | --- |
| Type of Screen | Technical Screen |

* Products

|  |  |
| --- | --- |
| List of Upstream Products | 1. Project Proposal PPT 2. SDPS\_ INVERTER DESIGN 3. System requirements 4. Hardware Specifications document 5. Test Software 6. CM checklist |
| List of Downstream Products | NA |
| List of Horizontal Products | 1. ^DAR report 2. ^Hardware Architecture 3. ^\* Schematics 4. ^\*Bill of Material (BOM) 5. ^\*Proto Board Testing Report 6. ^Layout 7. ^Hardware FMEA 8. Component level Hardware requirements 9. HW test Plan 10. HW Test report |

* Screen Scope

Upstream View:

Circuit Schematics, HW FMEA ,400 Level Requirement and HW test Plan should comply with system level requirements and hardware specification documents.

Downstream View:

Test Plan and Test Report for hardware proto board testing are complete and results are documented.

Hardware FMEA plan and test report.

Traceability will be available from 400 level hardware Requirement Specification to system level requirement specification in the DNG.

The product will follow CM guidelines for release process.

Quality and Configuration Manager view:

The products are as per the KPIT Screen guidelines for Quality and CM View.

The product will follow CM guidelines for release process

### Store

I/O/P/I - Input Product / Output Product / Pre-requisite Product / Intermediate Product

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| I/O/P/I | Product Name | CI Identifier | Source | Destination |
| I/-/-/- | Project Proposal PPT | Eaton\_Inverter\_KPIT\_Proposal\_Eaton\_NPe\_M3A and M3B\_Inverter\_Program\_V2 | KPIT | Hardware Uniphase |
| I/-/-/- | \*SDPS\_ INVERTER DESIGN | Eaton\_Inverter\_SDPS\_ INVERTER DESIGN | Planning uniphase | Hardware Uniphase |
| I/-/-/- | System requirements (200 Level System Requirement) | NPe\_M3\_M2\_Dual\_Core \_200\_Level\_System\_Requirement | System Uniphase | Hardware Uniphase |
| I/-/-/- | Hardware Requirement Specifications | Eaton\_Inverter\_Hardware Requirement Specification for Inverter Controller Board\_V5.2 | KPIT | Hardware Uniphase |
| I/-/-/- | Test Software | Eaton\_Inverter\_NPE\_Test Software | BSW | Hardware Uniphase |
| I/-/-/- | System FMEA | Eaton\_Inverter\_NPE\_System FMEA | System Uniphase | Hardware Uniphase |
| I/-/-/- | HW Safety Requirements | Eaton\_Inverter\_HW Safety Requirements | FUSA Uniphase | Hardware Uniphase |
| I/-/-/- | 300 Level PCB Requirement | Eaton\_Inverter\_300\_Level\_PCB\_Requirement | System Uniphase | Hardware Uniphase |
| I/-/-/- | CM checklist V1.0 | R.0033170 \_M2 inverter\_CM\_Checklist | Planning Uniphase | All uniphases |
| I/-/-/- | 300 -Level Power Circuit Requirement | Eaton\_Inverter\_300\_Level\_Power\_Circuit\_Requirement | System Uniphase | Hardware Uniphase |
| -/O/-/- | ^DAR report | Eaton\_Inverter\_DAR\_report | Hardware Uniphase | Customer |
| -/O/-/- | ^Hardware Architecture | Eaton\_Inverter\_Hardware\_Architecture | Hardware Uniphase | Customer |
| -/O/-/- | ^\* Schematics | Eaton\_Inverter\_Schematics\_April23rd | Hardware Uniphase | Customer |
| -/O/-/- | ^\*Bill of Material (BOM) | Eaton\_Inverter\_M2 Gate Driver Board\_2019\_03\_14 Controller\_Board\_BOM-Dec12th | Hardware Uniphase | Customer |
| -/O/-/- | ^\*Proto Board Testing Report | ^Eaton\_Inverter\_Proto Board Testing Report | Hardware Uniphase | Customer |
| -/O/-/- | ^Layout | Eaton\_Inverter\_\_PCB | Hardware Uniphase | Customer |
| -/O/-/- | ^Hardware FMEA | Eaton\_Inverter\_\_Hardware FMEA | Hardware Uniphase | Customer |
| -/O/-/- | Component level Hardware requirements | Eaton\_NPe\_M2\_Dual\_Core \_400 level Hardware requirements | Hardware Uniphase | Customer |
| -/O/-/- | HW test Plan | Eaton\_NPe\_M2\_Dual\_Core \_HW test Plan | Hardware Uniphase | Customer |
| -/O/-/- | HW Test report | Eaton\_NPe\_M2\_Dual\_Core \_HW\_modules\_verification\_Test\_Report | Hardware Uniphase | Customer |

\* Incremental Product

Prefix \* if product is incremental. Eg: \*Design Document

^ Product applicable / Delivered earlier in M3 Single Core(Inverter\_Design) and the SDPS in which it is mentioned is stored at :

<https://hjph3svn01.kpit.com/svn/R0032999/Program_Management/KPIT_Quality_Process/SDPS>

## BSW Uniphase

|  |  |
| --- | --- |
| **Purpose:**  The purpose of this uniphase is to perform following activities  All activities related to AUTOSAR and BSW shall be considered during this Uniphase. The final tested and AUTOSAR compliant MCAL with all modules ported and integrated will be ready after this Uniphase. Iterations will be required to port this SW initially on Evaluation board and later prototype HW board designed during project.  **Software Architecture subphase:**  The purpose of this subphase is to perform following activities:   * To derive SW Architecture design based on Software Safety Requirements (SSR – 300 Level).   SW architectural design is able to satisfy both the software safety requirements as well as the other software requirements which are in KPIT scope. Hence, in this subphase, both safety-related and non-safety-related software requirements are handled together   * To perform Safety Analysis on SW Architecture design   **Software Component Requirements subphase:**  The purpose of this subphase is to perform following activities:   * To derive SW Component Requirements (400 Level as per scope of the project) specifications (both non-safety and safety requirements together) based on SW Requirements & SSR (300 Level), SW Architecture design   **Software Component Design subphase:**  The scope of this subphase is to analyze the requirements generated in the Software Component Requirements subphase and prepare the Software Design.  Software Design consists of software architectural diagram, software design decisions, Interface design, low-level functions, and flowcharts for these functions, detailed data structures that will enable the team to develop optimal code.  The Safety analysis (DFMEA) will be performed on this design.  **Software Component Coding subphase:**  The scope of this subphase is to implement code referring to SW Component design.  **Software Component Integration subphase:**  The scope of this subphase is to integrate the units (modules) of BSW which are in KPIT scope.  Safety Project Plan, User Manual and any other related documents from KSAR BSW Product team and Integration team shall be referred for integration of KSAR BSW, RTE, test configurations and test applications.  **Software User Manual subphase:**  The scope of this subphase is to prepare the User Manual and Safety User Manual for respective SW components which are in KPIT scope. This manual can be used by user while integrating SW components/modules. | |
| **Entry Criteria :**  Artifacts/components are  Necessary to start with BSW uniphase   1. Project Proposal PPT 2. SDPS\_ INVERTER DESIGN 3. RH850 E2M MCAL. Compiler setup, Startup File 4. Controller Datasheet 5. Functional Safety Manual 6. CAN matrix 7. Software Requirement Document 8. HSI Document 9. KPIT AUTOSAR stack 10. KPIT Bootloader stack 11. System Requirements (200 Level)   **Software Architecture subphase:**   1. SW Requirements (300 Level) 2. SSR (300 Level) 3. HSIS 4. CAN matrix 5. RTE matrix 6. Technical Safety Concept (TSC) 7. System Design Architecture AUTOSAR Requirements (if applicable) 8. ISO26262 specifications (mainly Part 6)   **Software Component Requirements subphase:**   1. SW Requirements including SSR (300 Level) 2. SW Architecture design 3. AUTOSAR Requirements (if applicable) 4. ISO26262 specifications (mainly Part 6)   **Software Component Design subphase:**   1. SW Component Requirements (400 Level) 2. HSIS (refined) 3. SW Requirements including SSR (300 Level) 4. SW Architecture design 5. AUTOSAR Requirements (if applicable) 6. Configuration and/or Calibration data (if applicable) 7. ISO26262 specifications (mainly Part 6)   **Software Component Coding subphase:**   1. SW Component design 2. SW Component Requirements (400 Level) 3. HSIS (refined) 4. SW Requirements including SSR (300 Level) 5. SW Architecture design 6. AUTOSAR Requirements (if applicable) 7. ISO26262 specifications (mainly Part 6)   **Software Component Integration subphase:**   1. System Requirements including TSR (200 Level) 2. SW Requirements including SSR (300 Level) 3. SW Architecture design 4. Technical Safety Concept (TSC) 5. HSIS 6. AUTOSAR Requirements (if applicable) 7. ISO26262 specifications 8. SW Component Code (in KPIT scope) 9. KSAR BSW + RTE integrated from Integration team 10. SW Component Generators (if applicable) 11. C4K Tool 12. SW Component User Manual, SW Component Safety User Manual (if applicable) 13. SW Component Configuration Requirements   **Software User Manual subphase:**   1. SW Component Requirements (400 Level) 2. HSIS (refined) 3. SW Component design 4. SW Component (UTR) Testing report 5. SW Component testing report (STR) 6. AUTOSAR Requirements (if applicable) 7. ISO26262 specifications 8. SW Requirements including SSR (300 Level) 9. HSIS (refined) 10. SW Component design 11. SW Component Integration Testing report (ITR) 12. AUTOSAR Requirements (if applicable) 13. ISO26262 specifications (mainly Part 7) | **Exit Criteria :**  Below Artifacts/components are necessary for close the BSW uniphase   1. TestReport 2. SW Document 3. Component Requirements Specifications 4. Chronogram PPT 5. Accpetance-Test\_Results   **Software Architecture subphase:**   1. SW Architecture design (RTE+BSW)   **Software Component Requirements subphase:**   1. SW Component Requirements (400 Level) 2. HSIS (refined)   **Software Component Design subphase:**   1. SW Component design 2. SW Component Parameter Definition File (if applicable) 3. SW Component DFMEA   **Software Component Coding subphase:**   1. SW Component code   **Software Component Integration subphase:**   1. Integrated SW   **Software User Manual subphase:**   1. SW Component User Manual 2. SW Component Safety User Manual (if applicable) |

* EPMN

Diagram, calendar

Description automatically generated

### Management and Control

Refer [Section 3.1](#_Management_and_Control)

### Process

**Activity Description**

* + **MCAL shall be provided by Third party** – static code shall be provided by Renesas, we need to the configuration of each MCAL module using DaVinci configurator. Write test application for all MCAL modules, build them using GHS compiler and test using multi IDE debugger
  + Based on HSI document released by Hardware team, MCAL driver shall be configured. Identify delta in HSIS. Identify impacted modules, impacted modules configuration and testing.
  + MCAL testing shall be based on HSI document or any specific requirement by customer
  + CDD shall be developed based on requirement (if any) or as internally agreed based on all the stake holder. CDD design, implementation, unit testing and integration in main code
  + Configuration of Mode manager, RTE, OS – Analysis & understand AUTOSAR s/w requirements for respective drivers, identify module dependencies, configuration of module attributes, module level testing, implementation of test application if required, development of call out stubs if required
  + MCAL and CDDs shall be integrated and testing to be done on test bench -
  + For AUTOSAR, MCAL CDDs and AUTOSAR stack shall be integrated and testing shall be done on test bench -
  + Only Development testing shall be the scope of testing – will do only interface level testing on integrated code
  + Integration Testing shall be performed only if integration test plan is provided by customer
  + Conduct SME Review
  + Perform rework based on Review comments
  + All Technical Reviews will be performed as per review strategy document (Refer sect. [4.5](#_Testing_(Verification_and))
  + Conduct Screen with external experts (Upstream and Downstream reviewers)
  + Perform Release after screen closure. Upload artefacts on SVN. Send release mail to customer.

**Software Architecture subphase:**

* Activity Description

The objective of this activity is to prepare software architecture design and allocation of safety requirements to the components of RTE, BSW (which are in KPIT scope). Following AUTOSAR specifications as shall be referred:

* SW Requirements (300 level)
* SSR (300 level)
* TSR (200 Level)
* CAN matrix
* RTE matrix
* Technical Safety Concept (TSC)
* System Design Architecture
* HSIS
* AUTOSAR Layered Software Architecture
* AUTOSAR Specifications for Modules
* AUTOSAR Safety concept Status Report
* ISO26262 specifications

**Define SW Architecture**

During this activity, software requirements (300 level) and safety requirements (SSR) shall be analyzed and safety architecture shall be prepared. High level design components will be identified to cover the safety requirements and those will be documented in SW Architecture Design and Concept.

Enterprise Architect tool, Visio are used to document the SW Architecture design.

The traceability between the SSR specification and SW Architecture shall be provided.

Requirement Management strategy document mentioned in the link below shall be referred for the same.

[DNG link for Requirement Management strategy](https://loutcsvrtcp01.napa.ad.etn.com:9443/rm/web#action=com.ibm.rdm.web.pages.showArtifactV2&artifactURI=https%3A%2F%2Floutcsvrtcp01.napa.ad.etn.com%3A9443%2Frm%2Fresources%2F_9Uy4IdumEeiQR8CGNEJSYg&vvc.configuration=https%3A%2F%2Floutcsvrtcp01.napa.ad.etn.com%3A9443%2Frm%2Fcm%2Fstream%2F_DPrZtlSFEeiSR9QaYprQmA&componentURI=https%3A%2F%2Floutcsvrtcp01.napa.ad.etn.com%3A9443%2Frm%2Frm-projects%2F_DKdigVSFEeiSR9QaYprQmA%2Fcomponents%2F_DPiPwlSFEeiSR9QaYprQmA)

To ensure that the software architectural design captures the information necessary to allow the subsequent development activities to be performed correctly and effectively, the software architectural design will be described using Natural language and the Semi-formal notations. This will be achieved by using UML standard with Enterprise Architect tool, Visio.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sl. No.** | **Methods** | **ASIL A** | **ASIL B** | **ASIL C** | **ASIL D** | **Method Selection/Argumentation** |
| 1a | Natural Language - a | ++ | ++ | ++ | ++ | English Language shall be used. |
| 1b | Informal notations | ++ | ++ | + | + | - |
| 1c | Semi-formal notations - b | + | + | ++ | ++ | UML 2.0 notation (Enterprise Architect tool) |
| 1d | Formal notations | + | + | + | + | Not considered |
| a Natural language can complement the use of notations for example where some topics are more readily expressed in natural language or providing explanation and rationale for decisions captured in the notation.  b Semi-formal notations can include pseudocode or modelling with UML®, SysML®, Simulink® or Stateflow®. | | | | | | |

**Table 5 Notations for software architectural design**

**(Ref: Table 2 ISO26262:6-7)**

To avoid failures due to high complexity following principles will be followed for Software architecture design.

Principles for software architectural design:

| **Sl. No.** | **Methods** | **ASIL A** | **ASIL B** | **ASIL C** | **ASIL D** | **Argumentation** |
| --- | --- | --- | --- | --- | --- | --- |
| 1a | Appropriate hierarchical structure of the software components | ++ | ++ | ++ | ++ | Standard layered architecture of AUTOSAR is followed wherever applicable. |
| 1b | Restricted size and complexity of software components - a | ++ | ++ | ++ | ++ | Standard layered architecture of AUTOSAR is followed wherever applicable. |
| 1c | Restricted size of interfaces - a | + | + | + | + | Standard layered architecture of AUTOSAR is followed wherever applicable. |
| 1d | Strong cohesion within each software component - b | + | ++ | ++ | ++ | Standard layered architecture of AUTOSAR is followed wherever applicable. |
| 1e | Loose coupling between software components – b, c | + | ++ | ++ | ++ | Standard layered architecture of AUTOSAR is followed wherever applicable. |
| 1f | Appropriate scheduling properties | ++ | ++ | ++ | ++ | Standard layered architecture of AUTOSAR is followed wherever applicable. |
| 1g | Restricted use of interrupts – a, d | + | + | + | ++ | Use of interrupts is application/integrator’s responsibility. |
| 1h | Appropriate spatial isolation of the software components | + | + | + | ++ | OS of KSAR BSW provides the memory partitioning to the Software components. |
| 1i | Appropriate management of shared resources - e | ++ | ++ | ++ | ++ | OS of KSAR BSW provides to handle shared resources. |
| a In principles 1b, 1c, and 1g “restricted” means to minimize in balance with other design considerations.  b Principles 1d and 1e can, for example, be achieved by separation of concerns which refers to the ability to identify, encapsulate, and manipulate those parts of software that are relevant to a particular concept, goal, task, or purpose.  c Principle 1e addresses the management of dependencies between software components.  d Principle 1g can include minimizing the number, or using interrupts with a clear priority, in order to achieve determinism.  e Principle 1i applies for shared hardware resources as well as shared software resources in the case of coexistence. Such resource management can be implemented in software or hardware and includes safety mechanisms and/or process measures that prevent conflicting access to shared resources as well as mechanisms that detect and handle conflicting access to shared resources. | | | | | | |

**Table 6 Principles for software architectural design**

**(Ref: Table 3 ISO26262:6-7)**

Safety Analysis will be carried out on software architecture as described below under Safety Analysis activity. Depending on the results of the safety analyses, safety mechanisms for error detection and error handling shall be applied.

Safety mechanisms for error detection can include (Ref: ISO26262:6-7.4.12):

| **Sl. No.** | **Method/Technique** |
| --- | --- |
| 1a | Range checks of input and output data |
| 1b | Plausibility check (e.g. using a reference model of the desired behaviour, assertion checks, or comparing signals from different sources) |
| 1c | Detection of data errors (e.g. error detecting codes and multiple data storage) |
| 1d | Monitoring of program execution by an external element such as an ASIC or another software element performing a watchdog function. Monitoring can be logical or temporal monitoring or both |
| 1e | Temporal monitoring of program execution |
| 1f | Diverse redundancy in the design |
| 1g | Access violation control mechanisms implemented in software or hardware concerned with granting or denying access to safety-related shared resources |

Safety mechanisms for error handling can include (Ref: ISO26262:6-7.4.12):

|  |  |
| --- | --- |
| **Sl. No.** | **Method/Technique** |
| 1a | Deactivation in order to achieve and maintain a safe state |
| 1b | Static recovery mechanism (e.g. recovery blocks, backward recovery, forward recovery and recovery through repetition) |
| 1c | Graceful degradation by prioritizing functions to minimize the adverse effects of potential failures on functional safety |
| 1d | Homogenous redundancy in the design, which focuses primarily on controlling the effects of transient faults or random faults in the hardware on which a similar software is executed (e.g. temporal redundant execution of software) |
| 1e | Diverse redundancy in the design which implies dissimilar software in each parallel path and focuses primarily on the prevention or control of systematic faults in the software |
| 1f | Correcting codes for data |
| 1g | Access permission management implemented in software or hardware concerned with granting or denying access to safety-related shared resources |

Upper estimation of the required resources for embedded software will be mentioned in terms of execution time, RAM-ROM, hardware resources.

SW architecture design will also document the design decisions.

**Verification of SW Architecture Design**

The software architectural design will be verified as described in “Verification and Validation Strategy” Section 4.5 of this document.

1. Review

2. Inspection (Screen)

3. Safety Analyses

4. Integration Testing

**Review & Rework of SW Architecture Design**

Though the review is not suggested by ISO26262 for SW architecture, it is KPIT standard and hence review will be carried out for SW architecture design against review checklist by expert team members and review comments will be incorporated in SW architecture design documents. There will be two reviews planned: technical with Technical review checklist based on KPIT standard and Safety review-based ISO 26262 SW architecture Design recommendations.

Review is to ensure that:

* Design modules are developed for the software safety requirements and ensure the correctness of the same
* Interfaces between different design modules are correct
* Optimization, reusability, portability, scalability and efficiency have been considered
* Traceability to Software Safety Requirements

**Safety Analysis**

KPIT shall perform FTA and software DFMEA of SW components which are in KPIT scope.

**Perform FTA (if applicable)**

Fault Tree Analysis (FTA): Deductive analysis. In this activity, FTA will be started at SW components which are in KPIT scope level by identifying the Top-level event violating the respective SW component requirement.

Software component safety requirements and SW architecture design will be refined for the recommendations of the failures.

**Verification**

Verification will be planned for FTA using verification methods: review, screen, Confirmation Review.

**Review and rework of FTA**

During this activity, FTA report which is prepared in previous activity will be reviewed by expert team member for correctness and completeness. Reviewer will also check whether all the safety mechanisms identified is really needed. Review comments will be incorporated in the FTA reports.

**Confirmation review of FTA(screen)**

Aim of this activity isevaluation of the FTA Report to confirm the adherence to safety standard. During this activity, the FTA Report shall circulate to the Safety confirmation reviews. The review comments will be incorporated in the above Reports and its associated documents.

**Update Requirements & SW Architecture**

The safety analysis may contribute to the identification of new functional or non-functional software requirements that are not previously identified. Also, there could be scenarios impacting only the Architecture design which may trigger modifications in Architecture design. A CR shall be raised to update the required changes in software safety requirements and SW Architecture.

**Update Safety Case**

During this activity Safety Case document will be updated for SW architecture design and concept derived in the above activity. This includes capturing evidence of safety activities carried out during this subphase.

**Software Component Requirements subphase:**

* Activity Description

The objective of this activity is to prepare SW Component Requirements specifications. Following documents as shall be referred:

* SW Requirements including SSR (300 Level)
* SW Architecture design
* AUTOSAR Requirements (if applicable)
* Common SRS (if applicable)
* ISO26262 specifications

**Study Inputs and Prepare Software Requirement Specification document:**

In this activity, Software Component Requirement Document will be prepared. Software Component Requirement Documents shall include all the requirements coming from various inputs. In preparation of Software Component Requirement all the specified input documents will be analysed and requirements will be captured. ASIL allocation and decomposition from SW Requirements shall be considered and the SW Components are developed as per respective ISO26262 ASIL methods.

All SW Components in KPIT scope is to be developed as per ISO26262 process for applicable methods. Those shall contain SW Component requirements derived from SSR and Safety Mechanisms. Those shall also contains Safety Mechanisms to detect interference from memory or/and timing related arising due to interaction with third party provided SW Components which are not developed as per required ISO26262 ASIL methods.

Requirements shall be mentioned in natural language and semi-formal notations.

For each requirement following attributes shall be present

1. Requirement ID (e.g. generated from DNG)
2. Requirement Category (Requirement Type)
3. Safety Classification/ASIL (Safety/Safety Integrity Level)
4. Requirement status (Requirement Rational/Coverage/Defect/Validation conclusion)
5. Remarks

Appropriate selection of methods as per table below shall be followed for specification of software requirements.

| **Sl. No.** | **Methods** | **ASIL A** | **ASIL B** | **ASIL C** | **ASIL D** | **Argumentation** |
| --- | --- | --- | --- | --- | --- | --- |
| 1a | Natural language | ++ | ++ | ++ | ++ | English Language shall be used. |
| 1b | Informal notations - a, b | ++ | ++ | + | + | English Language shall be used. |
| 1c | Semi-formal notations - a, b, c, d | + | + | ++ | ++ | AUTOSAR standard notations shall be followed. |
| 1d | Formal notations - a | + | + | + | + | Not considered |
| a An appropriate selection of methods for the specification of safety requirement considers their adequacy to achieve the characteristics of safety requirement according to 6.4.2 for a specific issue to be specified, its complexity or the knowledge of the persons specifying or managing safety requirements. Examples include the use of state graphs or diagrams for specifying the complex behavior of software or hardware including many states or/and complex transitions.  b For higher level safety requirements (e.g. safety goals, functional and technical safety requirements) natural language and other types of informal notations are the most appropriate forms, though some requirements may be better handled with semi-formal notations.  c Semi-formal notation formulates requirements using natural language that is supplemented by mathematical or graphical elements such as equations, graphs, diagrams, flow charts, timing diagrams, and many other forms of representation (e.g. UML® and SysML™). Examples include model-based techniques and applying templates and controlled vocabulary for requirement sentences in natural language.  d For lower-level safety requirements where precise hardware and software behaviors and capabilities may be specified, semi-formal notations are more appropriate due to greater clarity. However, even here it may not be possible or necessary to use semi-formal techniques for every requirement. | | | | | | |

**Table 7 Specifying safety requirements**

**(Ref: Table 1 ISO26262:8-6)**

Requirement Management strategy document mentioned in the link below shall be referred for the same and traceability (in DNG) between Requirements shall be provided.

[DNG link for Requirement Management strategy](https://loutcsvrtcp01.napa.ad.etn.com:9443/rm/web#action=com.ibm.rdm.web.pages.showArtifactV2&artifactURI=https%3A%2F%2Floutcsvrtcp01.napa.ad.etn.com%3A9443%2Frm%2Fresources%2F_9Uy4IdumEeiQR8CGNEJSYg&vvc.configuration=https%3A%2F%2Floutcsvrtcp01.napa.ad.etn.com%3A9443%2Frm%2Fcm%2Fstream%2F_DPrZtlSFEeiSR9QaYprQmA&componentURI=https%3A%2F%2Floutcsvrtcp01.napa.ad.etn.com%3A9443%2Frm%2Frm-projects%2F_DKdigVSFEeiSR9QaYprQmA%2Fcomponents%2F_DPiPwlSFEeiSR9QaYprQmA)

**Review & Rework SW Component Requirement document:**

During this activity, Requirements will be verified by Review and inspection (Screen) as described in “Verification and Validation Strategy” section 4.5 of this document. SW Component Requirement document will be reviewed against review checklist by expert team members and review comments will be incorporated in SW Component Requirement document. There will be two reviews conducted based on two separate checklist one is technical review checklist and second is Safety checklist which is based on the ISO26262-8:6. SW Component Safety Requirement shall undergo Safety review.

| **Sl. No.** | **Methods** | **ASIL A** | **ASIL B** | **ASIL C** | **ASIL D** | **Argumentation** |
| --- | --- | --- | --- | --- | --- | --- |
| 1a | Verification by walk-through | ++ | + | o | o | NA |
| 1b | Verification by inspection | + | ++ | ++ | ++ | Review, Inspection (Screen). |
| 1c | Semi-formal verification - a | + | + | ++ | ++ | Review, Inspection (Screen). |
| 1d | Formal verification - a | o | + | + | + | NA |
| a Verification can be supported by executable models. | | | | | | |

**Table 8 Methods for the verification of safety requirements**

**(Ref: Table 2 ISO26262:8-6)**

**Refine/Update HSI Document**

Hardware dependent modules shall refine Hardware software interface specifications (HSIS) in this activity. HSIS will be detailed down to a level allowing the correct control and usage of hardware and shall describe each safety-related dependency between hardware and software.

Inputs from KPIT are provided to customer and HSI document is maintained by customer.

**Review & Rework of HSIS (refined)**

In this activity HSI inputs provided by KPIT will be reviewed by the customer to check the completeness and the adequacy of it for correct control and usage of the hardware.

**Update Safety Case**

During this activity Safety Case document will be updated for SW Component Requirement and HSIS (refined) derived in the above activity. This includes capturing evidence of safety activities carried out during this subphase.

**Software Component Design subphase:**

* Activity Description:

The goal of the Design activity is to analyze the requirements generated in the Software Component Requirements subphase and prepare the SW design document, Low level design, Safety architecture design for module.

SW design document (SDD) contains static and dynamic aspects like component overview, block diagram, static and dynamic design aspects, interface specifications, configurations and dependencies.

Safety architecture design contains the safety mechanisms to be part of respective components.

Low level design (LLD) contains detailed information/description on data structure design, macros, enums, external variables, calibration parameters, each function details and call tree.

SW design document, Safety architecture design are referred as SW Component architecture design and Low level design is referred as SW Component detail design. SW design document, Safety architecture design and Low level design together are prepared in EA and forms SW Component Design.

**Prepare Architecture design (SW design document, Safety architecture design) for SW Component**

During this activity, design components as specified above will be identified to cover the requirements by referring the SRS and those will be documented in SW design document, Safety architecture design of the module.

SW design document (SDD) contains static and dynamic aspects like component overview, block diagram, static and dynamic design aspects, interface specifications, configurations and dependencies.

Safety architecture design contains the safety mechanisms to be part of respective components.

SW design document, Safety architecture design also contains details like KPIT design decisions, High Level Design, design for Common Requirements, concept of execution, interface design to cover the requirements. The design will be prepared considering various aspects such as memory optimization, performance, reusability, portability, and scalability etc.

Enterprise Architect tool shall be used to document the SW design document, Safety architecture design.

The traceability between the module SW Component requirements and SW design document, Safety architecture design shall be provided. Following steps to be followed for establishing traceability between Design and SW Component requirements:

1. Design Ids would be given manually for each element
2. For any diagrams like Activity Diagram, Sequence Diagram, Dataflow Diagrams, User Diagrams, etc.; TAG should be provided only to packages
3. Traceability tool extracts requirements and design ids and generate Traceability Report

To ensure that the SW Component Architecture design captures the information necessary to allow the subsequent development activities to be performed correctly and effectively, the SW Component Architecture design will be described using the Semi-formal notations which is required as per respective ASIL requirements of ISO26262. This will be achieved by using UML standard with Enterprise Architect tool.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sl. No.** | **Methods** | **ASIL A** | **ASIL B** | **ASIL C** | **ASIL D** | **Argumentation** |
| 1a | Natural language - a | ++ | ++ | ++ | ++ | English Language shall be used. |
| 1b | Informal notations | ++ | ++ | + | + | Not considered. |
| 1c | Semi-formal notations - b | + | + | ++ | ++ | UML 2.0 notation (Enterprise Architect tool). |
| 1d | Formal notations | + | + | + | + | Not considered. |
| a Natural language can complement the use of notations for example where some topics are more readily expressed in natural language or provide an explanation and rationale for decisions captured in the notations.  EXAMPLE To avoid possible ambiguity of natural language when designing complex elements, a combination of an activity diagram with natural language can be used.  b Semi-formal notations can include pseudocode or modelling with UML®, SysML®, Simulink® or Stateflow®.  NOTE UML®, SysML®, Simulink® and Stateflow® are examples of suitable products available commercially. This information is given for the convenience of users of this document and does not constitute an endorsement by ISO of these products. | | | | | | |

**Table 9 Notations for software unit design**

**(Ref: Table 5 ISO26262:6-8)**

To avoid failures due to high complexity following principles will be followed for SW Component Architecture design:

| **Sl. No.** | **Methods** | **ASIL A** | **ASIL B** | **ASIL C** | **ASIL D** | **Argumentation** |
| --- | --- | --- | --- | --- | --- | --- |
| 1a | Appropriate hierarchical structure of the software components | ++ | ++ | ++ | ++ | Standard layered architecture of AUTOSAR is followed wherever applicable. |
| 1b | Restricted size and complexity of software components - a | ++ | ++ | ++ | ++ | Standard layered architecture of AUTOSAR is followed wherever applicable. |
| 1c | Restricted size of interfaces - a | + | + | + | + | Standard layered architecture of AUTOSAR is followed wherever applicable. |
| 1d | Strong cohesion within each software component - b | + | ++ | ++ | ++ | Standard layered architecture of AUTOSAR is followed wherever applicable. |
| 1e | Loose coupling between software components – b, c | + | ++ | ++ | ++ | Standard layered architecture of AUTOSAR is followed wherever applicable. |
| 1f | Appropriate scheduling properties | ++ | ++ | ++ | ++ | Standard layered architecture of AUTOSAR is followed wherever applicable. |
| 1g | Restricted use of interrupts – a, d | + | + | + | ++ | Use of interrupts is application/integrator’s responsibility. |
| 1h | Appropriate spatial isolation of the software components | + | + | + | ++ | OS of KSAR BSW provides the memory partitioning to the Software components. |
| 1i | Appropriate management of shared resources - e | ++ | ++ | ++ | ++ | OS of KSAR BSW provides to handle shared resources. |
| a In principles 1b, 1c, and 1g “restricted” means to minimize in balance with other design considerations.  b Principles 1d and 1e can, for example, be achieved by separation of concerns which refers to the ability to identify, encapsulate, and manipulate those parts of software that are relevant to a particular concept, goal, task, or purpose.  c Principle 1e addresses the management of dependencies between software components.  d Principle 1g can include minimizing the number, or using interrupts with a clear priority, in order to achieve determinism.  e Principle 1i applies for shared hardware resources as well as shared software resources in the case of coexistence. Such resource management can be implemented in software or hardware and includes safety mechanisms and/or process measures that prevent conflicting access to shared resources as well as mechanisms that detect and handle conflicting access to shared resources. | | | | | | |

Table 10: Principles for software architectural design

**(Ref: Table 3 ISO26262:6-7)**

Allocation of Safety/non-safety requirements to the SW Components will be shown through the traceability in DNG.

Upper estimation of the required resources for module will be mentioned in terms of execution time, RAM-ROM, hardware resources.

**Verification of SW Component Architecture design (SDD)**

The software architectural design of module will be verified by methods described in “Verification and Validation Strategy” section 4.5 of this document.

**Review & Rework of SW Component Architecture design (SDD)**

Review will be carried out for SW Component architecture design against review checklist by expert team members and review comments will be incorporated in SW Component Architecture design documents. There will be two reviews planned technical with Technical review checklist based on KPIT standard and Safety review-based ISO 26262 SW architecture Design recommendations.

Review is to ensure that:

* Design modules are developed for the SW Component requirements and ensure the correctness of the same.
* Interfaces between different design modules are correct.
* Optimization, reusability, portability, scalability and efficiency have been considered.
* Traceability to Requirements.

**Prepare PDF (if applicable)**

During this activity, Parameter Definition File (PDF) provided as part of AUTOSAR specification will be updated. These updates shall be made referring to AUTOSAR SWS and SRS Configuration Parameter Definition File will be verified with the KPIT ECU Configuration Editor.

**Review and rework of PDF (if applicable)**

During this activity, the PDF that are developed in the previous activity will be reviewed against the Technical Review Checklist. Review is to ensure that:

* The parameters that are listed out in the Parameter Definition File are complete and correct.
* The content provided in the Parameter Definition File is as per the ECU Parameter Configuration Specification and SRS.

**Prepare LLD**

During this activity, design modules will be identified to cover the requirements by referring to SW Component requirements and SW Component Architecture design (SDD). Low level design (LLD) contains detailed information/description on data structure design, macros, enums, external variables, calibration parameters, each function details and call tree, flowcharts.

The Low-Level Design contain low-level functions to cover the requirements. The design will be prepared considering various aspects such as memory optimization, performance, reusability, portability and scalability etc. The LLD will contain detailed flowcharts for the identified functions including hardware accesses (for applicable modules), detailed data structures and macros.

To ensure that the software component unit design captures the information necessary to allow the subsequent development activities to be performed correctly and effectively, the software component unit design will be described using the notations given below:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sl. No.** | **Methods** | **ASIL A** | **ASIL B** | **ASIL C** | **ASIL D** | **Argumentation** |
| 1a | Natural language | ++ | ++ | ++ | ++ | English Language shall be used. |
| 1b | Informal notations | ++ | ++ | + | + | Not considered. |
| 1c | Semi-formal notations | + | + | ++ | ++ | UML 2.0 notation (Enterprise Architect tool). |
| 1d | Formal notations | + | + | + | + | C language to define header files. |
| a Natural language can complement the use of notations for example where some topics are more readily expressed in natural language or providing explanation and rationale for decisions captured in the notation.  b Semi-formal notations can include pseudocode or modelling with UML®, SysML®, Simulink® or Stateflow®. | | | | | | |

**Table 11 Notations for software unit design**

**(Ref: Table 5 ISO26262:6-8)**

To avoid failures and to achieve correctness of data flow and control flow within and between software units, consistency of interfaces between software units, correct order of execution of functions within software units design principles for software units and implementation at code level given below shall be followed:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sl. No.** | **Methods** | **ASIL A** | **ASIL B** | **ASIL C** | **ASIL D** | **Argumentation** |
| 1a | One entry and one exit point in subprograms and functions - a | ++ | ++ | ++ | ++ | MISRA C 2012 shall be followed. |
| 1b | No dynamic objects or variables, or else online test during their creation - a | + | ++ | ++ | ++ | MISRA C 2012 shall be followed. |
| 1c | Initialization of variables | ++ | ++ | ++ | ++ | MISRA C 2012 shall be followed. |
| 1d | No multiple use of variable names - a | ++ | ++ | ++ | ++ | MISRA C 2012 shall be followed. |
| 1e | Avoid global variables or else justify their usage - a | + | + | ++ | ++ | If any global variable usage shall be justified in unit design. |
| 1f | Restricted use of pointers - a | + | ++ | ++ | ++ | MISRA C 2012 shall be followed. |
| 1g | No implicit type conversions - a | + | ++ | ++ | ++ | MISRA C 2012 shall be followed. |
| 1h | No hidden data flow or control flow | + | ++ | ++ | ++ | MISRA C 2012 shall be followed. |
| 1i | No unconditional jumps - a | ++ | ++ | ++ | ++ | MISRA C 2012 shall be followed. |
| 1j | No recursions | + | + | ++ | ++ | MISRA C 2012 shall be followed. |
| a Principles 1a, 1b, 1d, 1e, 1f, 1g and 1i may not be applicable for graphical modelling notations used in model-based development. | | | | | | |

**Table 12 Design principles for software unit design and implementation**

**(Ref: Table 6 ISO26262:6-8)**

**Verification**

Verification will be planned for Low level Design document using verification methods described in “Verification and Validation Strategy” Section 4.5 of this document.

**Review and rework of LLD**

During this activity, the LLD that is developed in the previous activity will be reviewed. Review is to ensure that:

* Low-level functions have been identified to cover all the requirements in the SW Component requirements.
* Whether flowcharts generated for the low-level functions are correct.
* Correctness of detailed data structures.
* Optimization, reusability, portability, scalability and efficiency have been considered.

**Perform DFMEA, DFA**

DFMEA is performed for individual safety software components (e.g. CPLD, SBC, Gate driver, GTM, Wdg Manager, Reset Manager) to analyze their failures and impact of this failures on the system. Qualitative software DFMEA shall be performed for software components which are in KPIT scope.

During this activity, Design Failure Mode Analysis will be performed to analyze the design efficiency towards achieving the software safety. This is a group activity where experts analyze the possible failure modes together and come up with detection/prevention mechanism. This activity checks the efficiency of safety mechanism. If any new safety requirements are derived from the analysis it will be considered in the software requirement. Safety analysis report will be prepared to provide the details of analysis, findings and possible improvements in the design. For any fault if no prevention or detection mechanism is available, recommended action shall be derived.

Each API of all SW Component units will be analyzed in detail for complete functionality to ensure the efficiency of the design and the safety mechanisms.

DFA shall be performed after the completion of above qualitative software DFMEA to verify the usage of memory, timing protection, data exchange features provided by other BSW modules (e.g. OS) for achieving Freedom From Interference (FFI) between modules developed for mixed ASIL rating (e.g. WdgM and other SW components in same partition which are of different ASIL).

And also to identify any more safety mechanisms to achieve FFI.

**Verification**

Verification will be planned for DFMEA, DFA using verification methods: review, screen

**Review and rework of DFMEA, DFA /FTA**

During this activity, Safety analysis report which is prepared in previous activity will be reviewed by expert team member for correctness and completeness. Reviewer will also check whether all the safety mechanisms identified are really needed. Review comments will be incorporated in the DFMEA, DFA reports.

**Confirmation review of DFMEA, DFA /FTA(screen)**

Aim of this activity isevaluation of the safety analysis Report to confirm the adherence to safety standard. During this activity, the DFMEA, DFA Reports are circulated to the Safety Consultant and team for their reviews. The review comments will be incorporated in the above Reports and its associated documents.

**Update SW Component requirements, SDD, LLD**

The safety analysis may contribute to the identification of new functional or non-functional software requirements that are not previously identified. Also, there could be scenarios impacting only the HLD or LLD which may trigger modifications in HLD or LLD. A Change Request (CR) shall be raised to update the required changes in SW Component requirements.

**Update Safety Case**

During this activity Safety Case document will be updated for SDD and LLD derived in the above activity. This includes capturing evidence of safety activities carried out during this subphase.

**Software Component Coding subphase:**

**Develop Code**

In this activity common code will be developed referring to SW Component Design. Team will develop ‘.c’ and ‘.h’ files considering the file structure and function header, commenting style and revision history block as per Coding Standards for all the functions identified in SW Component Design.

Code optimization factors shall be considered during coding. Code will be developed considering portability and reusability.

Coding will be carried out considering appropriate methods to cover the reentrancy aspects for the functions that are identified as re-entrant.

Big Endian and Little-Endian compile options will be used for bit/byte ordering for defining the bus access/device related data structures.

The team will identify all constants and variables to group them into a single module for easy location in ROM and RAM, respectively.

The team will identify all configurable data items including Enums, Typedefs and #define to group them into a respective header (‘.h’) files.

All the functions identified in the SW Component design should be implemented in the code.

The team will identify all global and common data types including Enums, Typedefs and “#defines” to form the data module to be used / included in all other SW Components.

Following documents will be referred for developing code

* Specification of Compiler Abstraction
* Specification of Memory Mapping
* Specification of Platform Types
* Specification of Standard Types
* Common SRS (if applicable)

Based on the SW Component architectural design and LLD, the detailed design of the software units is developed. The detailed design will be implemented directly as source code, in accordance with the coding guidelines. Following design principles to be followed at implementation level:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sl. No.** | **Methods** | **ASIL A** | **ASIL B** | **ASIL C** | **ASIL D** | **Argumentation** |
| 1a | One entry and one exit point in subprograms and functions - a | ++ | ++ | ++ | ++ | MISRA C 2012 shall be followed. |
| 1b | No dynamic objects or variables, or else online test during their creation - a | + | ++ | ++ | ++ | MISRA C 2012 shall be followed. |
| 1c | Initialization of variables | ++ | ++ | ++ | ++ | MISRA C 2012 shall be followed. |
| 1d | No multiple use of variable names - a | ++ | ++ | ++ | ++ | MISRA C 2012 shall be followed. |
| 1e | Avoid global variables or else justify their usage - a | + | + | ++ | ++ | If any global variable usage shall be justified in unit design. |
| 1f | Restricted use of pointers - a | + | ++ | ++ | ++ | MISRA C 2012 shall be followed. |
| 1g | No implicit type conversions - a | + | ++ | ++ | ++ | MISRA C 2012 shall be followed. |
| 1h | No hidden data flow or control flow | + | ++ | ++ | ++ | MISRA C 2012 shall be followed. |
| 1i | No unconditional jumps - a | ++ | ++ | ++ | ++ | MISRA C 2012 shall be followed. |
| 1j | No recursions | + | + | ++ | ++ | MISRA C 2012 shall be followed. |
| a Principles 1a, 1b, 1d, 1e, 1f, 1g and 1i may not be applicable for graphical modelling notations used in model-based development. | | | | | | |

**Table 13 Design principles for software unit design and implementation**

**(Ref: Table 6 ISO26262:6-8)**

Traceability should be maintained between Low-level design and code.

1. Design Ids along with element information will be exported from EA to excel using tool for easy reference.
2. Design Ids will be updated in code header files for structures and group of enums as comment
3. Design Ids will be updated in code source files in API header as comment for APIs and for others will be provided as comments
4. Traceability tool extracts design ids and filenames with line numbers /API names and generate Traceability Report.

**Perform Static Analysis and Code Sanity Checks:**

Developed code shall be verified against the coding guidelines using rule checker (Katapult) tools. The reports generated Katapult tools shall be analyzed for zero violations. In case of any violations that cannot be fixed, approval for violations along with proper justification/rationale shall be taken from approving authority.

The code along with generated (if applicable) C source and header files will be verified for compliance with MISRA standard using Katapult tool. Also, dynamic/runtime behavior analysis of code including generated C Source and header files will be performed using Katapult. The code shall be updated based on the warnings reported by the tool. The updated code will be reviewed for correctness. If any of the warning could not be adhered, then that part of the code will be analyzed manually to ensure that there is no functional and safety impact. Rationale for non-adherence to any rule shall be justified through comments in code. Consolidated reports shall be prepared.

**Technical and Safety Review of Code:**

During this activity, the Code developed in the previous activity will be reviewed. Review is performed to ensure that:

* Adherence to Coding Standards as per Coding guidelines
* The correctness of the functions to meet the proposed functionality
* All data for their correct data type definition, initialization, usage as per data type definition throughout the code and placement of the same in appropriate module
* Error and warning free compilation of each modules
* The code optimization, reusability, and portability to various platform/compiler variants
* Reentrancy aspects have been correctly covered in the code

**Software Component Integration subphase:**

Prepare Integrated Stack:

* This activity deals with configuration and generation of MCAL, BSW, RTE, CDD SW Components (in KPIT scope).
* Generated files (if applicable) are integrated with test application. Memory map and linker file shall be prepared.
* Integrated SW shall be built with controller specific compiler options. After the stack is built successfully, executable is flashed into the actual target hardware.

Review and Rework of Integrated stack:

In this phase, integrated stack shall be reviewed. Technical review is to ensure

* Integrated SW shall contain all the relevant SW components (in KPIT scope)
* Integrated SW shall not contain duplicate copies of the files
* Check for temporary variables created during testing present in the Integrated SW
* Unnecessary files/stubs shall be removed from the Integrated SW

Technical review checklist will be used to perform the review and report will be prepared.

* Run Polyspace tool for static code analysis.

**Software User Manual subphase:**

* Activity Description:

The goal of this activity is to analyze all the products of all the subphases and to come up with the SW Component User Manual and SW Component Safety User Manual document.

**Prepare User Manual:**

During this activity, the team shall document the information related to interface of the module, precautions to be taken, file structure, Application Interface files, APIs and the usage details. The team will also document deviations from the AUTOSAR Specifications (if applicable), in the User Manual.

**Verification**

Verification will be planned for SW Component User manual document using verification methods: Review, Screen.

**Review & Rework of User Manual:**

During this activity, the SW Component User Manual prepared in the previous activity will be reviewed. Review is to ensure that:

* The information provided in User Manual is enough and understandable.
* The clear presentation of the information related to the software modules is available.

**Prepare Safety User Manual (if applicable):**

During this activity, the module team/owner will document the safety features implemented for the specific KPIT BSW components, error and protection related information, interface of the module, precautions to be taken, usage details and links to respective User and Safety User Manuals. It will have a brief overview of the implemented technical safety features or concepts. It also includes configuration details of these technical safety features (including all protection features) for the ASIL application. This will describe the customer actions required to be taken to ensure controllability in case of a failure. This will also describe the relevant functions (i.e. the intended usage, the status information or user interaction) and their operating modes. The assumptions of use, external dependencies, constraints, precautions, failure modes and safe state, checklist to ensure integrity, expected actions from the user will also be mentioned in safety user manual. This document shall describe actions required by user of the SW Component to ensure functional safety of KPIT BSW and SW Components in safety item.

**Review & Rework of Safety User Manual:**

During this activity, the Safety User Manual in the previous activity will be reviewed. Review is to ensure that:

* The information provided in Safety User Manual is enough and understandable.
* The clear presentation of the information related to the software modules is available.

**Safety Case update:**

During this activity, Safety Case will be updated for the KPIT BSW components.

This activity will be done in parallel with preparation of User Manual. This includes capturing evidence of safety activities carried out during this subphase.

### Screen

|  |  |
| --- | --- |
| Type of Screen | Technical Screen |

* Products

|  |  |
| --- | --- |
| List of Upstream Products | 1. Project Proposal PPT 2. SDPS\_ INVERTER DESIGN 3. RH850 E2M MCAL. Compiler setup, Startup File 4. Controller Datasheet 5. Functional Safety Manual 6. CAN matrix 7. Software Requirement Document 8. HSI Document 9. KPIT AutoSAR stack 10. KPIT Bootloader stack 11. CM checklist   **Software Architecture subphase:**   1. SW Requirements (300 Level) 2. SSR (300 Level) 3. HSIS 4. CAN matrix 5. RTE matrix 6. Technical Safety Concept (TSC) 7. System Design Architecture AUTOSAR Requirements (if applicable) 8. ISO26262 specifications   **Software Component Requirements subphase:**   1. SW Requirements including SSR (300 Level) 2. SW Architecture design 3. AUTOSAR Requirements (if applicable) 4. ISO26262 specifications   **Software Component Design subphase:**   1. SW Component Requirements (400 Level) 2. HSIS (refined) 3. SW Requirements including SSR (300 Level) 4. SW Architecture design 5. AUTOSAR Requirements (if applicable) 6. Configuration and/or Calibration data (if applicable) 7. ISO26262 specifications   **Software Component Coding subphase:**   1. SW Component design 2. SW Component Requirements (400 Level) 3. HSIS (refined) 4. SW Requirements including SSR (300 Level) 5. SW Architecture design 6. AUTOSAR Requirements (if applicable) 7. ISO26262 specifications   **Software Component Integration subphase:**   1. System Requirements including TSR (200 Level) 2. SW Requirements including SSR (300 Level) 3. SW Architecture design 4. Technical Safety Concept (TSC) 5. HSIS 6. AUTOSAR Requirements (if applicable) 7. ISO26262 specifications 8. SW Component Code (in KPIT scope) 9. KSAR BSW + RTE integrated from Integration team 10. SW Component Generators (if applicable) 11. C4K Tool 12. KSAR BSW modules, SW Component User Manual (if applicable) 13. SW Component Configuration Requirements   **Software User Manual subphase:**   1. SW Component Requirements (400 Level) 2. HSIS (refined) 3. SW Component design 4. SW Component (UTR) Testing report 5. SW Component testing report (STR) 6. AUTOSAR Requirements (if applicable) 7. ISO26262 specifications 8. SW Requirements including SSR (300 Level) 9. HSIS (refined) 10. SW Component design 11. SW Component Integration Testing report (ITR) 12. AUTOSAR Requirements (if applicable) 13. ISO26262 specifications |
| List of Downstream Products | NA |
| List of Horizontal Products | 1. TestReport (ITP, ITR, Testlogs – if applicable. ) (Optional) 2. Tools(C4K, Altilium Setup) (Optional) 3. Architecture Document update (Delta Changes for M2 Dual core) (Optional) 4. Chronogram PPT (Optional) 5. Accpetance-Test\_Results (Optional) 6. Supporting\_Document\_Structure.txt (Optional) 7. CDD Requirement Specification 8. CDD Safety design (SAD) 9. CDD Design Document 10. CDD Low Level Design Document 11. Impact Analysis 12. Traceability-Sw |

* Screen Scope

Upstream View:

Autosar421\_compliant\_ported\_MCAL\_stack\_and\_Test Code is delivered in accordance with Upstream Products.

Traceability is not applicable for MCAL modules however CDD modules will be traced to 400 level CRS requirements which will trace to software requirement document

Downstream View:

The Autosar421\_compliant\_ported\_MCAL\_stack\_and\_Test Code, CRS and Design documents can be followed for the subsequent Uniphase of project execution.

Quality and Configuration Manager View:

The products are as per the KPIT Screen guidelines for Quality and CM View.

The product will follow CM guidelines for release process as well as with CM audit process.

Software Architecture subphase:

Upstream View:

**SW Architecture and Concept:** The screener will check compliance of the SW architecture with software requirements (300 level) and software safety requirements. The Traceability section will be verified for the correct mapping of design elements to the requirements. The Screener shall also check the adherence to the design guidelines.

SW architecture design elements will be traced to SSR in the system requirement specifications in the DNG.

**FTA:** The screener will check for completeness of the analysis with respect to SW Architecture. For FTA report, all the safety related architectural components are covered in the analysis. All the possible failure modes are analyzed for the architectural block.

Downstream View:

**SW Architecture:** The screen scope is to check for the identification, adequacy and correctness of reusable and configurable units. The screener will check for the understandability and correctness of the design elements. The screener will also check whether the design elements are sufficient to develop Low level (Component level) design, code and are testable. The screener will check for correctness of information provided by Traceability in the DNG. The screener shall check the design compatibility with the hardware.

**FTA:** The screener will check that reports are complete, recommendations are feasible.

Quality and Configuration Manager View:

The products are as per the KPIT Screen guidelines for CM and Quality view.

The product will follow CM guidelines for release process as well as with CM audit process.

Software Component Requirements subphase:

Upstream View:

**SW Component Requirements:**

The screener shall check if all the inputs required for SW Component Requirements are correctly, completely identified and the correct/latest version of requirement documents are analyzed.

The screener shall also ensure that all the open points (logged in DNG) references, AUTOSAR SWS references (if applicable) and SSR specification references are correct and complete

SW Component Requirements will be traced to SW Requirements and SW Component Safety Requirements to SSR in the system requirement specifications in the DNG.

**HSIS**:

The screener shall check the completeness of the HSIS w.r.t required interaction for software to work correctly.

Downstream View:

**SW Component Requirements:**

The screener shall check the usability of the SW Component Requirements for the next Software Architecture subphase preparation of the project.

The screener shall ensure that the SW Component Requirements items are without any ambiguous statements. Also if there are any open points still requiring clarification from the customer, the screener shall ensure that those open points are raised in DNG for the same.

Screener shall also ensure that the responses from the customer on open points in DNG are updated in the SRS document

Screener shall also ensure that the responses from the customer on open points in DNG are updated in the SW Component Requirements document.

**HSIS:**

The screener shall check if HSIS derived is sufficient for the implementation of Software.

Quality and Configuration Manager View:

The products are as per the KPIT Screen guidelines for CM and Quality view.

The product will follow CM guidelines for release process as well as with CM audit process.

Software Component Design subphase:

**Upstream View:**

**SW Component Architecture design (SDD), PDF**: The screener will check compliance of the SW architecture with requirements. The Traceability section will be verified for the correct mapping of design elements to the requirements. The Screener shall also check the adherence to the design guidelines.

**LLD:** The Screener will check whether the design decisions are covered in LLD. As SDD and LLD are present in same EA project, components from SDD are reused in LLD. Hence no explicit links are required.

**DFMEA, and FTA:** The screener will check for completeness of the analysis with respect to SW Component Architecture design (SDD) and LLD. For DFMEA, FTA report, all the safety related components and FFI are covered in the analysis. All the possible failure modes are analyzed for the component architecture design.

**Downstream View:**

**SW Component Architecture design (SDD), PDF:** The screen scope is to check for the identification, adequacy and correctness of reusable and configurable units. The screener will check for the understand ability and correctness of the design elements. The screener will also check whether the design elements are sufficient to develop Low level design, code and are testable. The screener will check for correctness of information provided in the Traceability matrix. The screener shall check the design compatibility with the hardware

**LLD:** The screen scope is to check for the identification, adequacy, and correctness of reusable and configurable items. The screener will check for the understandability and correctness of the design elements. The screener will also check whether the design elements are enough to develop code and are testable. The screener will check for correctness about functionality.

**DFMEA, FTA:** The screener will check that reports are complete, recommendations are feasible.

**Quality & Configuration Management View:**

The screener shall check the products as per KPIT screen guidelines for CM and Quality view.

Software Component Coding subphase:

**Upstream View:**

**Prepared SW Component code**:

SW Component code developed is correct, consistent, and complete in accordance with SW Component Design and SW Component Requirements (400 Level).

The screener shall verify that the code is developed in accordance with the SW Component Design and SW Component Requirements (400 Level). It shall also be verified that code is developed for each element of the SW Component Design and SW Component Requirements (400 Level).

**Downstream View:**

**Prepared SW Component code**:

SW Component code can be compiled and build without Error and warning.

The screener shall check code is clear such that a UTP can be developed, and developed code can be tested.

**Quality & Configuration Management View:**

The screener shall check the products as per KPIT screen guidelines for CM and Quality view.

Software Component Integration subphase:

Upstream View:

Prepared Integrated SW:

Integrated SW is correct, consistent and complete in accordance with SW Requirements including SSR (300 Level) and SW Architecture design.

The screener shall verify that the code is developed in accordance with the SW Requirements including SSR (300 Level). It shall also be verified that code is developed for each element of the SW Requirements including SSR (300 Level) and SW Architecture design.

**Downstream View:**

Prepared Integrated SW:

SW Component code can be compiled and build without Error and warning.

The screener shall check code is clear such that a ITP can be developed and integrated SW can be tested.

**Quality & Configuration Management View:**

The screener shall check the products as per KPIT screen guidelines for CM and Quality view. Screener shall also check the consistency across modules.

Software User Manual subphase:

**Upstream View:**

SW Component User Manual & Safety User Manual:

The screener shall check the mapping of the details provided in module User Manual to SW Component Requirements (400 Level), AUTOSAR SWS (if applicable), Common SRS (if applicable) and SW Component Design.

The screener shall check the mapping of the details provided in KPIT SW User Manual to SW Requirements including SSR (300 Level), Common SRS (if applicable) and SW Architecture design.

Safety Case:

The screener shall check mapping of Safety Case Report to ISO26262 recommended safety processes and artifacts.

**Downstream View:**

SW Component User Manual & Safety User Manual:

The screener shall check for the clarity and understandability of the information documented in User Manuals. The screener shall also check whether the user will be able to integrate the SW components with the application referring the information provided in the respective user manuals.

Safety Case:

The screener shall check completeness of Safety Case Report & ensures it compiles all the evidence of the safety activities followed and the safety features implemented in the project.

**Quality & Configuration Management View:**

The screener shall check the products as per KPIT screen guidelines for CM and Quality view. Screener shall also check the consistency across modules.

### Store

**Products table**

I/O/P/I - Input Product / Output Product / Pre-requisite Product / Intermediate Product

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| I/O/P/I | Product Name | CI Identifier | Source | Destination |
| I/-/-/- | Project Proposal PPT | Eaton\_NPe\_M2\_Dual\_core\_Inverter\_Program\_V2.0 | KPIT | BSW Uniphase |
| I/-/-/- | \*SDPS\_ INVERTER DESIGN | Eaton\_NPe\_M2\_SDPS\_Multicore\_Inverter | Planning | BSW Uniphase |
| I/-/-/- | RH850 E2M MCAL. Compiler setup, Startup File (Requirement) | Eaton\_NPe\_M2\_RH850 E2M MCAL. Compiler setup, Startup File | Customer | BSW Uniphase |
| I/-/-/- | Controller Datasheet (Data Sheet , Sample code) | Eaton\_NPe\_M2\_Controller Datasheet | Customer | BSW Uniphase |
| I/-/-/- | Functional Safety Manual | Eaton\_NPe\_M3\_M2\_Functional Safety Manual | Customer | BSW Uniphase |
| I/-/-/- | CAN matrix (CAN DBC) | Eaton\_NPe\_M3\_M2\_CAN\_Signal\_List | Customer | BSW Uniphase |
| I/-/-/- | Software Requirement Document (Requirement Customer Specs. Systems, High Level SW Req.) | Eaton\_NPe\_M3\_M2\_Software\_Requirement\_Document | Systems | BSW Uniphase |
| I/-/-/- | HSI Document (HSIS Document ) | Eaton\_NPe\_M2\_HSI Document | Hardware | BSW Uniphase |
| I/-/-/- | KPIT AUTOSAR stack (AUTOSAR Stack) | Eaton\_NPe\_M2\_KPIT\_AutoSAR stack | KPIT AUTOSAR team | BSW Uniphase |
| I/-/-/- | KPIT/Eaton Bootloader stack (Requirement) (Optional) | Eaton\_NPe\_M2\_KPIT\_Bootloader stack | KPIT Bootloader team | BSW Uniphase |
| I/-/-/- | Fault reaction Plan | NPe\_M3\_M2\_Dual\_Core\_Systems\_Fault \_Reaction\_Plan | System Uniphase | BSW Uniphase |
| I/-/-/- | SW Safety Analysis | Eaton\_NPe\_M3\_M2\_Dual\_Core\_SW Safety Analysis Report | FUSA Uniphase | BSW Uniphase |
| I/-/-/- | Systems Requirements (200L ) | NPe\_M3\_M2\_Dual\_Core \_200\_Level\_System\_Requirement | System Uniphase | BSW Uniphase |
| I/-/-/- | High Level SW Requirement (300L) | NPe\_M3\_M2\_Dual\_Core\_300\_Level\_HighLevel SW\_requirement | System Uniphase | BSW Uniphase |
| I/-/-/- | CAN Signal List | NPe\_M2\_Dual\_Core \_CAN\_Signal\_List | System Uniphase | BSW Uniphase |
| I/-/-/- | BSW\_and\_TestCode (Tested MCAL Package, CDD Code , Integrated Stack , Code) | Eaton\_NPE\_M2\_DualCore\_BSW\_and\_TestCode | Software Uniphase | BSW Uniphase |
| I/-/-/- | CM checklist V1.0 | M2 inverter\_CM\_Checklist | Planning Uniphase | All uniphases |
| I/-/-/- | Traceability | Eaton\_NPe\_M3\_M2\_TraceabilityS/w | Multicore Implementation for M2 Inverter | Customer, Multicore Implementation for M2 Inverter |
| I/-/-/- | CDD Requirement Specification | Eaton\_NPe\_M3\_M2\_400L | Multicore Implementation for M2 Inverter | Customer, Multicore Implementation for M2 Inverter |
| I/-/-/- | CDD Safety design (SAD) | Eaton\_NPe\_M3\_M2\_SAD | Multicore Implementation for M2 Inverter | Customer, Multicore Implementation for M2 Inverter |
| I/-/-/- | CDD Design Document | Eaton\_NPe\_M3\_M2\_SDD | Multicore Implementation for M2 Inverter | Customer, Multicore Implementation for M2 Inverter |
| I/-/-/- | CDD Low Level Design Document | Eaton\_NPe\_M3\_M2\_LLD | Multicore Implementation for M2 Inverter | Customer, Multicore Implementation for M2 Inverter |
| I/-/-/- | Architecture Document | Eaton\_NPe\_M3\_M2\_Dualcore\_Architecture | Software Uniphase | Customer, VnV |
| -/O/-/- | TestReport (ITP, ITR, Testlogs – if applicable. ) (Optional) | Eaton\_NPE\_M3\_M2\_DualCore\_TestReport | Software Uniphase | Customer |
| -/O/-/- | Tools(C4K, Altilium Setup) (Optional) | Eaton\_NPe\_M2\_Dualcore\_Tools\_Setup | Software Uniphase | Customer, VnV |
| -/O/-/- | Architecture Document update (Delta Changes for M2 Dual core) (Optional) | Eaton\_NPe\_M3\_M2\_Dualcore\_Architecture | Software Uniphase | Customer, VnV |
| -/O/-/- | Chronogram PPT (Optional) | Eaton\_NPe\_M2\_Chronogram | BSW Uniphase | Customer |
| -/O/-/- | Accpetance-Test\_Results (Optional) | Eaton\_NPe\_M2\_Accpetance\_Test\_Results | BSW Uniphase | Customer |
| -/O/-/- | Supporting\_Document\_Structure.txt (Optional) | Eaton\_NPe\_M2\_Supporting\_Document\_Structure.txt | BSW Uniphase | Customer |
| -/O/-/- | CDD Requirement Specification | Eaton\_NPe\_M3\_M2\_400L | Multicore Implementation for M2 Inverter | Customer, Multicore Implementation for M2 Inverter |
| -/O/-/- | CDD Safety design (SAD) | Eaton\_NPe\_M3\_M2\_SAD | Multicore Implementation for M2 Inverter | Customer, Multicore Implementation for M2 Inverter |
| -/O/-/- | CDD Design Document | Eaton\_NPe\_M3\_M2\_SDD | Multicore Implementation for M2 Inverter | Customer, Multicore Implementation for M2 Inverter |
| -/O/-/- | CDD Low Level Design Document | Eaton\_NPe\_M3\_M2\_LLD | Multicore Implementation for M2 Inverter | Customer, Multicore Implementation for M2 Inverter |
| -/O/-/- | Impact Analysis | Eaton\_NPe\_M2\_ImpactAnalysis | Multicore Implementation for M2 Inverter | Customer, Multicore Implementation for M2 Inverter |
| -/O/-/- | Traceability-Sw | Eaton\_NPe\_M3\_M2\_TraceabilityS/w | Multicore Implementation for M2 Inverter | Customer, Multicore Implementation for M2 Inverter |

\* Incremental Product

Prefix \* if product is incremental. Eg: \*Design Document

^ Product applicable / Delivered earlier in M2 Single Core(Inverter\_Design) and the SDPS in which it is mentioned is stored at : SVN Path as below :

<https://hjph3svn01.kpit.com/svn/R0032999/Program_Management/KPIT_Quality_Process/SDPS>

**Software Architecture subphase:**

I/O/P/I - Input Product / Output Product / Pre-requisite Product / Intermediate Product

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| I/O/P/I | Product Name | CI Identifier | Source | Destination |
| I/-/-/- | \*SDPS\_ INVERTER DESIGN | Eaton\_NPe\_M2\_SDPS\_Multicore\_Inverter | Planning | All Subsequent Uniphases |
| I/-/-/- | SSR | Eaton\_NPe\_M2\_Software\_Safety\_requirement | Software Safety Requirements Uniphase | All Subsequent Uniphases |
| I/-/-/- | TSC | Eaton\_NPe\_M2\_TSC | Customer | All Subsequent Uniphases |
| I/-/-/- | HSIS | Eaton\_NPe\_M2\_HSIS | Customer | All Subsequent Uniphases |
| -/O/-/- | Software Architecture | Eaton\_NPe\_M3\_M2\_Dualcore\_Architecture | Software Architecture | All Subsequent Uniphases |

\* Incremental Product

Prefix \* if product is incremental. Eg: \*Design Document

**Software Component Requirements subphase:**

I/O/P/I - Input Product / Output Product / Pre-requisite Product / Intermediate Product

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| I/O/P/I | Product Name | CI Identifier | Source | Destination |
| I/-/-/- | \*SDPS\_ INVERTER DESIGN | Eaton\_NPe\_M2\_SDPS\_Multicore\_Inverter | Planning | All Subsequent Uniphases |
| I/-/-/- | TSR | Eaton\_NPe\_M2\_Technical\_Safety\_requirement | Customer | All Subsequent Uniphases |
| I/-/-/- | Software Requirement Specification\* | Eaton\_NPe\_M3\_M2\_Software\_Requirement Specification\* | Software Requirements | All Subsequent Uniphases |
| I/-/-/- | SSR | Eaton\_NPe\_M3\_M2\_Software\_Safety\_requirement | Software Requirements | All Subsequent Uniphases |
| I/-/-/- | Software Architecture | Eaton\_NPe\_M2\_M2\_Dualcore\_Architecture | Software Architecture | All Subsequent Uniphases |
| I/-/-/- | HSIS | Eaton\_NPe\_M2\_HSIS | Customer | All Subsequent Uniphases |
| -/O/-/- | Software Component Requirement Specification (Optional)\* | Eaton\_NPe\_M3\_M2\_400L | Software Component Requirements | All Subsequent Uniphases |

\* Incremental Product

Prefix \* if product is incremental. Eg: \*Design Document

**Software Component Design subphase:**

I/O/P/I - Input Product / Output Product / Pre-requisite Product / Intermediate Product

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| I/O/P/I | Product Name | CI Identifier | Source | Destination |
| I/-/-/- | \*SDPS\_ INVERTER DESIGN | Eaton\_NPe\_M2\_SDPS\_ INVERTER DESIGN | Planning | All Subsequent Uniphases |
| I/-/-/- | SSR | Eaton\_NPe\_M3\_M2\_Software\_Safety\_requirement | Software Safety Requirements | All Subsequent Uniphases |
| I/-/-/- | Software Component Requirement Specification\* | Eaton\_NPe\_M2\_Software\_Component\_requirement | Software Component Requirements | All Subsequent Uniphases |
| I/-/-/- | HSIS | Eaton\_NPe\_M2\_HSIS | Customer | All Subsequent Uniphases |
| I/-/-/- | Software Architecture | Eaton\_NPe\_M3\_M2\_Dualcore\_Architecture | Software Architecture | All Subsequent Uniphases |
| -/O/-/- | SW Component Architecture design (Optional) | Eaton\_NPe\_M3\_M2\_SAD  Eaton\_Npe\_M3\_M2\_SDD | Software Component Design | All Subsequent Uniphases |
| -/O/-/- | SW Low Level Design (Optional) | Eaton\_NPe\_M3\_M2\_LLD | Software Component Design | All Subsequent Uniphases |

\* Incremental Product

Prefix \* if product is incremental. Eg: \*Design Document

**Software Component Coding subphase:**

I/O/P/I - Input Product / Output Product / Pre-requisite Product / Intermediate Product

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| I/O/P/I | Product Name | CI Identifier | Source | Destination |
| I/-/-/- | \*SDPS\_ INVERTER DESIGN | Eaton\_NPe\_M2\_SDPS\_ INVERTER DESIGN | Planning | All Subsequent Uniphases |
| I/-/-/- | SW Component Architecture design | Eaton\_NPe\_M3\_M2\_SAD  Eaton\_NPe\_M3\_M2\_SDD | Software Component Design | All Subsequent Uniphases |
| I/-/-/- | LLD | Eaton\_NPe\_M3\_M2\_LLD | Software Component Design | All Subsequent Uniphases |
| I/-/-/- | SSR | Eaton\_NPe\_M3\_M2\_Software\_Safety\_requirement | Software Safety Requirements | All Subsequent Uniphases |
| I/-/-/- | Software Component Requirement Specification\* | Eaton\_NPe\_M3\_M2\_400L | Software Component Requirements | All Subsequent Uniphases |
| I/-/-/- | HSIS | Eaton\_NPe\_M2\_HSIS | Customer | All Subsequent Uniphases |
| I/-/-/- | Software Architecture | Eaton\_NPe\_M3\_M2\_Dualcore\_Architecture | Software Architecture | All Subsequent Uniphases |
| -/O/-/- | SW Component code | Eaton\_NPE\_M2\_DualCore\_BSW | Software Coding and Testing | All Subsequent Uniphases |

\* Incremental Product

Prefix \* if product is incremental. Eg: \*Design Document

**Software Component Integration subphase:**

I/O/P/I - Input Product / Output Product / Pre-requisite Product / Intermediate Product

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| I/O/P/I | Product Name | CI Identifier | Source | Destination |
| I/-/-/- | \*SDPS\_ INVERTER DESIGN | Eaton\_NPe\_M2\_SDPS\_ INVERTER DESIGN | Planning | All Subsequent Uniphases |
| I/-/-/- | TSR | Eaton\_NPe\_M2\_Technical\_Safety\_requirement | Customer | All Subsequent Uniphases |
| I/-/-/- | Software Requirement Specification\* | Eaton\_NPe\_M3\_M2\_Software\_Requirement Specification\* | Software Requirements | All Subsequent Uniphases |
| I/-/-/- | SSR | Eaton\_NPe\_M3\_M2\_Software\_Safety\_requirement | Software Requirements | All Subsequent Uniphases |
| I/-/-/- | Software Architecture | Eaton\_NPe\_M3\_M2\_Dualcore\_Architecture | Software Architecture | All Subsequent Uniphases |
| I/-/-/- | HSIS | Eaton\_NPe\_M2\_HSIS | Customer | All Subsequent Uniphases |
| I/-/-/- | SW Component User Manual | UM | Software Component User Manual | All Subsequent Uniphases |
| I/-/-/- | SW Component Safety User Manual | SUM | Software Component User Manual | All Subsequent Uniphases |
| -/O/-/- | Integrated SW | Eaton\_NPE\_M2\_DualCore\_BSW\_and\_TestCode | Software Component Integration | All Subsequent Uniphases |

\* Incremental Product

Prefix \* if product is incremental. Eg: \*Design Document

**Software User Manual subphase:**

I/O/P/I - Input Product / Output Product / Pre-requisite Product / Intermediate Product

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| I/O/P/I | Product Name | CI Identifier | Source | Destination |
| I/-/-/- | \*SDPS\_ INVERTER DESIGN | Eaton\_NPe\_M2\_SDPS\_ INVERTER DESIGN | Planning | All Subsequent Uniphases |
| I/-/-/- | SW Component Architecture design | Eaton\_NPe\_M3\_M2\_SAD  Eaton\_NPe\_M3\_M2\_SDD | Software Component Design | All Subsequent Uniphases |
| I/-/-/- | SW Low Level Design | Eaton\_NPe\_M3\_M2\_LLD | Software Component Design | All Subsequent Uniphases |
| I/-/-/- | Software Component Requirement Specification\* | Eaton\_NPe\_M3\_M2\_400L | Software Component Requirements | All Subsequent Uniphases |
| I/-/-/- | HSIS | Eaton\_NPe\_M2\_HSIS | Customer | All Subsequent Uniphases |
| I/-/-/- | SW Architecture | Eaton\_NPe\_M3\_M2\_Dualcore\_Architecture | Software Architecture | All Subsequent Uniphases |
| I/-/-/- | SW Component (UTR) Testing report | Eaton\_NPe\_M2\_UTR | Software Coding and Testing | All Subsequent Uniphases |
| I/-/-/- | Integrated SW Test Report | Eaton\_NPe\_M2\_ITR | Software Component Integration and Verification | All Subsequent Uniphases |
| -/O/-/- | SW Component User Manual  (Optional) | Eaton\_NPe\_M2\_UM | Software Component User Manual | All Subsequent Uniphases |
| -/O/-/- | SW Component Safety User Manual  Optional) | Eaton\_NPe\_M2\_SUM | Software Component User Manual | All Subsequent Uniphases |
| -/O/-/- | KPIT SW User Manual  Optional) | Eaton\_NPe\_M2\_UM | Software User Manual | All Subsequent Uniphases |
| -/O/-/- | KPIT SW Safety User Manual (Optional) | Eaton\_NPe\_M2\_SUM | Software Component User Manual | All Subsequent Uniphases |

\* Incremental Product

Prefix \* if product is incremental. Eg: \*Design Document

## ASW Uniphase

|  |  |
| --- | --- |
| **Purpose:**  **The purpose of this uniphase is to perform following activities**   1. Software Requirements Specification 2. RTE Integration   **Note: ASW uniphase was part of single core and no more a uniphase in multicore** | |
| **Entry Criteria for Software**  Below Artifacts/components are  necessary to start with ASW Uniphase   1. Project Proposal PPT 2. SDPS\_ INVERTER DESIGN 3. Eaton Skeleton model and SW components code 4. Arxml files from NPe 5. Arxml files from Eaton 6. Software Components Source Code from NPe 7. Interface sheet from Eaton & NPe 8. Interface sheet (CARASI) from NPe 9. AUTOSAR Modeling Guidelines 10. High level architecture from NPe 11. High level application SW architecture Diagram from Eaton 12. HSI Document 13. CAN Signal List 14. OS Tasks Sheet 15. AUTOSAR Service Software Component Arxml File 16. System Requirement Specifications 17. Test Plan from Eaton/NPe | **Exit Criteria for Software**  Below Artifacts/components are necessary to close ASW Uniphase   1. Software Requirement Specification (300 or 400 Level) 2. RTE Integrated Code 3. Test Report 4. Software\_Requirement (300 or 400 Level?) 5. Documents (If applicable) |

* 300EPMN

Diagram

Description automatically generated

### Management and Control

Refer [Section 3.1](#_Management_and_Control)

### Process

* **Activity Description** 
  + Software Requirement Specification:

1. Extract high level (200 level) System Requirement Specifications from DNG
2. Analyse 200 level System Requirement Specifications
3. Prepare functional block diagram, Identify input, processing and output
4. Write 300 Level Software Requirements from System Requirements
5. Review will be performed by SME for the Software Requirement Specs document
6. Rework will be performed on review comments
7. Conduct Screen with external experts (Upstream and Downstream reviewers)
8. Perform Release after screen closure. Upload artefacts on SVN. Send release mail to customer.
9. Upload Software Requirement Specifications on DNG tool with traceability to System Requirements
   * RTE integration:

RTE Integration involves interface testing 3 scenarios - NPe Micro models, Npe and Eaton models and ASW-BSW integration.

* 1. NPe and Eaton models RTE Integration -

1. Analyse Signal provided by Eaton & NPe
2. Prepare Test Plan
3. Analyse Models and Code provided by Eaton
4. Map signals between Eaton and Npe Models
5. Write Wrapper for ASW and Npe Software Components Code
6. Test the build at code level (ASW and Npe S/W Components)
   1. Mircro Model RTE Integration (NPe Models)
7. Analyse Signal provided by NPe
8. Analyse and Validate ARXML provided by NPe
9. Prepare Test Plan
10. RTE Integration using C4K
11. Write Wrapper for FID
12. Test the FID build
13. Conduct Review
14. Perform rework based on Review comments
    1. ASW BSW (RTE) Integration -
15. Analyse Signal provided by Eaton & NPe
16. Map signals between ASW and BSW
17. Prepare Test Report
18. Build and Test final build
19. Perform ASW and BSW integration testing
20. Conduct SME Review
21. Perform rework based on Review comments
22. All Technical Reviews will be performed as per review strategy document (Refer sect. [4.5](#_Testing_(Verification_and))
23. Conduct Screen with external experts (Upstream and Downstream reviewers)
24. Perform Release after screen closure. Upload artefacts on SVN. Send release mail to customer

### Screen

|  |  |
| --- | --- |
| Type of Screen | Technical Screen |

* Products

|  |  |
| --- | --- |
| List of Upstream Products | 1. Project Proposal PPT 2. SDPS\_ INVERTER DESIGN 3. Eaton Skeleton model and SW components code 4. Arxml files from NPe 5. Arxml files from Eaton 6. Software Components Source Code from NPe 7. Interface sheet from Eaton & NPE 8. Interface sheet (CARASI) from NPe 9. AUTOSAR Modeling Guidelines 10. High level architecture from NPe 11. High level application SW architecture Diagram from Eaton 12. HSI Document 13. CAN Signal List 14. OS Tasks Sheet 15. AUTOSAR Service Software Component Arxml File 16. System Requirement Specifications 17. Test Plan from Eaton/Npe 18. CM checklist |
| List of Downstream Products | None (As no output products will be generated from ASW further.) |
| List of Horizontal Products | 1. ^Software Requirement Specification\* 2. ^RTE Integrated Code (ASW-BSW Final Build) \* 3. ^Test Report 4. ^Software Requirement 5. ^Source Code 6. ^Documents (If applicable) |

* Screen Scope

Upstream View:

Software Requirement Specification prepared is correct, consistent and complete in accordance with the System Requirement. Completeness of traceability from System Requirements to Software Requirements.

Required artefacts for RTE Integration are available and checked-in in the store.

Downstream View:

The Software Requirement Specification can be followed for the subsequent Uniphase of project execution. Completeness of bi-directional traceability for System and Software requirements.

RTE Integrated build is successful. ASW and BSW single interface verification. Test Report integration testing and test cases are complete, and results are documented.

Quality and Configuration Manager view:

The products are as per the KPIT Screen guidelines for Quality and CM View.

The product will follow CM guidelines for release process as well as with CM audit processes

### Store

I/O/P/I - Input Product / Output Product / Pre-requisite Product / Intermediate Product

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| I/O/P/I | Product Name | CI Identifier | Source | Destination |
| I/-/-/- | Project Proposal PPT and  M3 Inverter Proposal for 2020\_V3 | Eaton\_NPe\_M2\_Dual\_core\_Inverter\_Program\_V2.0 | KPIT | ASW Uniphase |
| I/-/-/- | \*SDPS\_ INVERTER DESIGN | Eaton\_NPe\_M2\_SDPS\_ INVERTER DESIGN | Planning | ASW Uniphase |
| I/-/-/- | Eaton Skeleton model and SW components code | Eaton\_NPe\_M2\_NPE\_Eaton Skeleton model and SW components code | Customer | ASW Uniphase |
| I/-/-/- | Arxml files from Eaton | Eaton\_NPe\_M2\_NPE\_Arxml files from Eaton | Customer | ASW Uniphase |
| I/-/-/- | Software Components Source Code from NPe | Eaton\_NPe\_M2\_Software Components Source Code from NPe | Customer | ASW Uniphase |
| I/-/-/- | Interface sheet from Eaton & NPE | Eaton\_NPe\_M2\_ Interface sheet from Eaton & NPE | Customer | ASW Uniphase |
| I/-/-/- | Interface sheet (CARASI) from NPe | Eaton\_NPe\_M2\_ Interface sheet (CARASI) from NPe | Customer | ASW Uniphase |
| I/-/-/- | AUTOSAR Modeling Guidelines | Eaton\_NPe\_M2\_ NPE\_AutoSAR Modeling Guidelines | KPIT | ASW Uniphase |
| I/-/-/- | High level architecture from NPe | Eaton\_NPe\_M2\_ High level architecture from NPe | Customer | ASW Uniphase |
| I/-/-/- | High level application SW architecture Diagram from Eaton | Eaton\_NPe\_M2\_ High level application SW architecture Diagram from Eaton | Customer | ASW Uniphase |
| I/-/-/- | HSI Document | Eaton\_NPe\_M2\_ HSI Document | Hardware | ASW Uniphase |
| I/-/-/- | CAN Signal List | Eaton\_NPe\_M3\_M2\_CAN Signal List | System | ASW Uniphase |
| I/-/-/- | OS Tasks Sheet | Eaton\_NPe\_M2\_ OS Tasks Sheet | Customer | ASW Uniphase |
| I/-/-/- | AUTOSAR Service Software Component arxml file | Eaton\_NPe\_M2\_ Autosar Service Software Component arxml file | AUTOSAR | ASW Uniphase |
| I/-/-/- | System Requirement Specifications | Eaton\_NPe\_M3\_M2\_System Requirement Specifications | System | ASW Uniphase |
| I/-/-/- | Test Plan | Eaton\_NPe\_M2\_ \_Test Plan | Customer | ASW Uniphase |
| I/-/-/- | CM checklist V1.0 | R.0033170 \_M2 inverter\_CM\_Checklist | Planning Uniphase | All uniphases |
| -/O/-/- | ^Software Requirement Specification\* | Eaton\_NPe\_M3\_M2\_Software Requirement Specification\* | ASW | Customer, VNV Uniphase, BSW Uniphase |
| -/O/-/- | ^RTE Integrated Code (ASW-BSW Final Build) \* | Eaton\_NPe\_M2\_RTE Integrated Code (ASW-BSW Final Build) \* | ASW | Customer, BSW Uniphase, VNV Uniphase |
| -/O/-/- | ^Test Report | Eaton\_NPe\_M2\_ Test Report | ASW | Customer |
| -/O/-/- | ^Software\_Requirement | Eaton\_NPe\_M2\_300\_Level\_Software\_Requirement | ASW | Customer |
| -/O/-/- | ^Source Code | Eaton\_NPe\_M2\_SourceCode | ASW | Customer, BSW Uniphase, |
| -/O/-/- | ^Documents (If applicable) | Eaton\_NPe\_M2\_ Documents | ASW | Customer |

\* Incremental Product

Prefix \* if product is incremental. Eg: \*Design Document

Product applicable / Delivered earlier in M2 Single Core(Inverter\_Design) and the SDPS in which it is mentioned is stored at : SVN Path

<https://hjph3svn01.kpit.com/svn/R0032999/Program_Management/KPIT_Quality_Process/SDPS>

## **VNV Uniphase**

|  |  |
| --- | --- |
| **Purpose:**  The purpose of this uniphase is to perform following activities:   * HIL Test Case Design * MIL/SIL testing for Model based Inverter Application Software Components (Input processing, output processing and Derating) * BSW PIL unit testing for Complex Device Drivers * Integration testing (Basic CAN signal testing)   Note: ASW (MIL SIL) Testing Not to be considered for multicore releases  **Software Component Coding subphase:**  The scope of this subphase is to prepare test specification referring to SW Component design and execute testing.  **Software Component Integration subphase:**  The scope of this subphase is to prepare the Integration Test Specification, Test Cases referring to the SW architecture design and test the embedded/integrated software on actual target platform.  Safety Project Plan, User Manual and any other related documents from KSAR BSW Product team and Integration team can be referred for integration of KSAR BSW, RTE, test configurations and test applications. | |
| **Entry Criteria:**  Below Artifacts/components are  necessary to start with VnV uniphase   1. Project Proposal PPT 2. SDPS\_ INVERTER DESIGN 3. System Requirements Specification 4. CAN Signal List 5. Hardware software interface document (HSI) 6. Software Requirement Specification 7. Component Requirement Specification 8. Source code 9. ITP/ITR/UTP/UTR templates   **Software Component Verification subphase:**   1. SW Component design 2. SW Component Requirements (400 Level) 3. HSIS (refined) 4. SW Requirements including SSR (300 Level) 5. SW Architecture design 6. AUTOSAR Requirements (if applicable) 7. ISO26262 specifications (mainly Part 6)   **Software Component Integration Verification subphase:**   1. System Requirements including TSR (200 Level) 2. SW Requirements including SSR (300 Level) 3. SW Architecture design 4. Technical Safety Concept (TSC) 5. HSIS 6. AUTOSAR Requirements (if applicable) 7. ISO26262 specifications 8. SW Component Code (in KPIT scope) 9. KSAR BSW + RTE integrated from Integration team 10. SW Component Generators (if applicable) 11. C4K Tool 12. KSAR BSW modules, SW Component User Manual (if applicable) 13. SW Component Configuration Requirements 14. ISO26262 specifications (Mainly Part 6) | **Exit Criteria:**  Below Artifacts/components are necessary to close with VNV uniphase   1. Test cases, reports requirement coverage and traceability for all the VNV milestones will be available in the DNG. 2. MIL/SIL Test scripts 3. BSW PIL test scripts   **Software Component Verification subphase:**   1. SW Component UTP (code coverage test specification) 2. SW Component (UTR) Testing report 3. SW Component STS (functional test specification) 4. SW Component testing report (STR) 5. Katapult (Static code analysis) Report   **Software Component Integration Verification subphase:**   1. Integration Test Specification 2. Integrated Software (in KPIT scope) 3. Integration Testing Report |

* EPMN



### Management and Control

Refer [Section 3.1](#_Management_and_Control)

### Process

* Activity Description
  + **Integration testing**: Based on system requirements, CAN signal list and HSIS document
    - Identify test cases which can cover from HIL test case design
    - Develop BSW test cases and test script
    - Execute integration test cases (Basic CAN signal testing) using control board and CANoe tool
    - Review, rework of integration test cases and test report
    - Upload integration test cases and test reports in the DNG.
    - All Technical Reviews will be performed as per review strategy document (Refer sect. [4.5](#_Review_&_Testing))
    - Organize Technical Screen and Perform Release
  + **ASW (MIL SIL) Testing**: Based on component level (unit level) requirements and MATLAB models
    - Design MIL/SIL test cases and test scripts and execute using MATLAB VNV toolset (Simulink test)
    - Review, rework and rereview of MIL/SIL test cases, test scripts and test reports
    - Upload MIL/SIL test cases, test scripts by giving reference and test reports in the DNG.
    - All Technical Reviews will be performed as per review strategy document (Refer sect. [4.5](#_Review_&_Testing))
    - Organize Technical Screen and Perform Releases

Note: ASW (MIL SIL) Testing Not to be considered for multicore releases

* + **BSW PIL Testing**: Based on component level (unit level) requirements and BSW complex device driver source files
    - Design BSW PIL unit test cases and test scripts and execute using Tessy tool
    - All Technical Reviews will be performed as per review strategy document (Refer sect. [4.5](#_Review_&_Testing))
    - Review rework and rereview of BSW PIL test cases, test scripts and test reports
    - Upload BSW PIL test cases, test scripts by giving reference and test reports in the DNG.
    - Iteration of Integration testing and HIL test case design will be performed based on updated system requirements, CAN signal list and HSIS document.
    - Organize Technical Screen and Perform Release

**Software Component Verification subphase:**

SW Component Code Coverage Testing and SW Component Functional Testing:

**Activity**

In this subphase code will be verified using following methods:

| **Sl. No.** | **Methods** | **ASIL A** | **ASIL B** | **ASIL C** | **ASIL D** | **Method selection/argumentation** | **Work Product** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 1a | Walk-through – a | ++ | + | o | o | Not recommended | - |
| 1b | Pair Programming - a | + | + | + | + | Not Applicable | - |
| 1c | Inspection – a | + | ++ | ++ | ++ | Review and Screen will be conducted | Review, Inspection (Screen) reports. |
| 1d | Semi-formal verification | + | + | ++ | ++ | Flowcharts in EA model will be verified for the correctness and completeness | Review, Inspection (Screen) reports. |
| 1e | Formal verification | o | o | + | + | Not considered | - |
| 1f | Control flow analysis – b, c | + | + | ++ | ++ | Polyspace will be used for verification | Polyspace reports |
| 1g | Data flow analysis – b, c | + | + | ++ | ++ | Polyspace will be used for verification | Polyspace reports |
| 1h | Static code analysis - d | ++ | ++ | ++ | ++ | Polyspace (MISRA C checks) will be used | Polyspace reports |
| 1i | Static analyses based on abstract interpretation - e | + | + | + | + | NA | - |
| 1j | Requirements-based test - f | ++ | ++ | ++ | ++ | Module Functional and Code coverage Testing | Functional and Code coverage Test Report |
| 1k | Interface test - g | ++ | ++ | ++ | ++ | Module Functional and Code coverage Testing | Functional and Code coverage Test Report |
| 1l | Fault injection test - h | + | + | + | ++ | SW Component safety mechanism testing | Functional Test Report |
| 1m | Resource usage evaluation - i | + | + | + | ++ | SW Component Functional Testing | Functional Test Report |
| 1n | Back-to-back comparison test between model and code, if applicable - j | + | + | ++ | ++ | NA | - |
| a For model-based development these methods are applied at the model level, if evidence is available that justifies confidence in the code generator used.  b Methods 1f and 1g can be applied at the source code level. These methods are applicable both to manual code development and to model-based development.  c Methods 1f and 1g can be part of methods 1e, 1h or 1i.  d Static analyses are a collective term which includes analysis such as searching the source code text or the model for patterns matching known faults or compliance with modelling or coding guidelines.  e Static analyses based on abstract interpretation are a collective term for extended static analysis which includes analysis such as extending the compiler parse tree by adding semantic information which can be checked against violation of defined rules (e.g. data-type problems, uninitialized variables), control-flow graph generation and data-flow analysis (e.g. to capture faults related to race conditions and deadlocks, pointer misuses) or even meta compilation and abstract code or model interpretation.  f The software requirements at the unit level are the basis for this requirements-based test. These include the software unit design specification and the software safety requirements allocated to the software unit.  g This method can be used to provide evidence for the integrity of used and exchanged data.  h In the context of software unit testing, fault injection test means to modify the tested software unit (e.g. introduce faults into the software) for the purposes described in 9.4.2. Such modifications include injection of arbitrary faults (e.g. by corrupting values of variables, by introducing code mutations, or by corrupting values of CPU registers).  i Some aspects of the resource usage evaluation can only be performed properly when the software unit tests are executed on the target environment or if the emulator for the target processor adequately supports resource usage tests.  j This method requires a model that can simulate the functionality of the software units. Here, the model and code are stimulated in the same way and results compared with each other.  EXAMPLE In the case of model-based design results of non-floating-point operations can be compared. | | | | | | | |

Table 14 Methods for Verification of Unit Design

**(Ref: Table 7 ISO26262:6-9)**

**Prepare UTP and Test Scripts:**

In this activity, module specific Unit Test Plans (UTP also called as Code coverage Test plan) along with associated test scripts will be prepared.

In this activity, SW Component specific LLD prepared as part of SW Component Design and code will be analyzed, and the unit test cases will be derived. During this activity, unit test cases and/or test scripts will be prepared after analysis of requirements, analysis of equivalence classes, boundary values and branch conditions. Unit Test Plan (UTP) contains test cases for Statement coverage, Branch coverage, test cases for MC/DC coverage and test cases for boundary conditions.

Note: Refer to “Verification and Validation Strategy” Section 4.5 of this document for more details on Unit testing.

Traceability will be provided in the UTP document to Low level design elements of Software Design.

1. Design Ids along with element information will be exported from EA to excel using tool.
2. Design Ids will be provided in UTP
3. Traceability tool uses excel derived in point 1 and point 2 to generate Traceability Report.

**Prepare SW Component Test Plan (Functional test Specification):**

During this activity, test cases are derived for all the functional requirements of SW Component requirement into the SW Component Test Specification document. The preparation of the SW Component test plan involves the below said activities:

* Analyze SW Component requirement and Create test plan covering all requirements
* Identify different Test groups based on functionalities / group of functionalities which shall be tested using one set of configurations
* Identify Key configuration parameter for each test group
* For each test group, Identify Default conditions and commonly used macros which shall be used to pass values as argument and verify out parameter
* For each test case, provide unique and sequential test case id with proper title
* Identify precondition specific to test case
* In test description, provide test steps like invocation of module provided APIs, test APIs and stub functions
* In Expected result, verify return value, OUT parameter for the APIs invoked in test description

There should be a bi-directional traceability created as below,

1. SRS - SW Component Test Specification Mapping
2. SW Component Test Specification - SRS Mapping

Methods to derive the Unit Test cases:

| **Sl. No.** | **Methods** | **ASIL A** | **ASIL B** | **ASIL C** | **ASIL D** | **Argumentation** |
| --- | --- | --- | --- | --- | --- | --- |
| 1a | Analysis of requirements | ++ | ++ | ++ | ++ | Requirements and Design at Unit level and shall be analyzed to come up with Requirement based Unit test cases |
| 1b | Generation and analysis of equivalence classes - a | + | ++ | ++ | ++ | Valid and Invalid regions for range test cases shall be derived |
| 1c | Analysis of boundary values - b | + | ++ | ++ | ++ | Test cases for boundary values shall be derived |
| 1d | Error guessing based on knowledge or experience - c | + | + | + | + | NA. |
| a Equivalence classes can be identified based on the division of inputs and outputs, such that a representative test value can be selected for each class.  b This method applies to interfaces, values approaching and crossing the boundaries and out of range values.  c Error guessing tests can be based on data collected through a “lessons learned” process and expert judgment. | | | | | | |

**Table 15 Methods for deriving test cases for software unit testing**

**(Ref: Table 8 ISO26262:6)**

**Technical and Safety Review of Unit Test Plan:**

During this activity, the Unit Test Plan prepared in the previous activity will be reviewed. Review is to ensure that:

* Test cases and test scripts are derived from the design functions to have the MC/DC and Branch Coverage.
* Unit test cases are in compliance with the SW Component design specification.
* Unit Test cases have been derived by considering the following:
* Analysis of LLD
* Generation and analysis of equivalence classes
* Analysis of boundary values
* UTP prepared is in adherence to Unit Test Plan guidelines which is stored in the configuration management system of the project.

**Technical and Safety Review of SW Component Test Plan:**

During this activity, the Test Plan prepared in the previous activity will be reviewed. Technical review is to ensure that:

* The Module test cases cover all the requirements specified in SW Component requirements
* Test cases have been identified for the functions defined in SW Component design
* The correctness and adequacy of the test cases for performing the tests
* Platform dependent test cases have been identified correctly and the expected outputs are correct for the combinations provided
* There are no duplicate and missing test cases
* Whether all the test cases can be simulated
* Whether the test scripts (wherever required) can be developed using this test plan to test the required functionality
* Whether all the test cases could be executed as per the test plan
* Whether possible negative test cases are covered

**Prepare Test environment for Code coverage Testing**

Target Deployment Port (TDP) will be required to perform the Unit Testing on Tessy Tool.

TDP will be developed by studying the type of target controller and complier manual (e.g Renesas RH850). Developed TDP will be tested by sample unit test cases in the Tessy Unit Testing Tool.

**Execute Unit Test Plan and Generate Unit Test Report**

Prerequisite for this activity is completion of coding and Unit Test Plan. In this activity, the unit test cases will be executed on the target hardware. During unit testing, if there are any defects, those will be analyzed and fixed in code. All unit test cases will be re-executed with updated unit code for correct result.

To evaluate the completeness of test cases and to demonstrate that there is no unintended functionality, the coverage of requirements at the software unit level will be determined and the structural coverage will be measured in accordance with the metrics listed below:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sl. No.** | **Methods** | **ASIL A** | **ASIL B** | **ASIL C** | **ASIL D** | **Method Selection/Argumentation** |
| 1a | Statement coverage | ++ | ++ | + | + | 100% Statement coverage |
| 1b | Branch coverage | + | ++ | ++ | ++ | 100% Branch coverage |
| 1c | MC/DC (Modified Condition/Decision Coverage) | + | + | + | ++ | MC/DC coverage |

**Table 16 Structural coverage metrics at the software unit level**

**(Ref: Table 12 ISO26262:6-9)**

If coverage required is not achieved, then justification will be provided in the test report.

All test cases will be tested for both instrumented and non-instrumented code. The consolidated report will be generated which contains the result of both cases.

The test report will be generated from the Tessy tool for the executed test cases. Code coverage report will be provided. Along with the Unit Test Report, an anomaly report will be generated to justify the uncovered code path. The Unit Test Plan is used to prepare the Unit Test Report. The Unit test report will consist of the test case Id, test result and remarks.

Note: Unit testing on hardware dependent TDP is not needed for SW Component which are target independent. Hence coverage testing for SW Component independent of Target hardware will be done on x86 (PC) environment. While porting or functional testing of these modules on target platform will be performed.

**Technical and Safety Review of Unit Test Report:**

During this activity, the Unit Test Report prepared in the previous activity will be reviewed against review checklist by expert team members. Review comments will be incorporated in Unit Test Report Document. Review is to ensure that:

* Whether Coverage testing is carried out using a Unit testing tool (Tessy) for MC/DC coverage, Branch coverage, Statement coverage
* If instrumented code is used to determine the degree of coverage, it can be necessary to show that the instrumentation has no effect on the test results. This can be done by repeating the tests with non-instrumented code
* A rationale is given for the level of coverage achieved based on accepted dead code (e.g. code for debugging) or code segments depending on different software configurations or code not covered can be verified using complementary methods (e.g. inspections)
* UTR prepared is in adherence to Unit testing guidelines which is stored in the configuration management system of the project
* All the test cases are executed, and all the test cases are passed. If there is any test case that has failed, it should contain the details in Remarks. All the failed test cases should be captured in anomaly report
* The UTP and the script file are matching to each other. Test cases identified in UTP and script are in-line with the Unit testing Check list
* Verify if the compared value of instrumented and non-instrumented code generated in combined report are same
* Check if the test environment is prepared for correct device file (SW Component dependent on Target hardware), compiler version and TDP for specific compiler
* Verify if the test cases mentioned for variant are correct for each module

**Develop Test Application:**

During this activity, one or multiple test applications will be developed by development team (jointly with VnV team) based on the SW Component Test Plan, wherever necessary. The test application will directly invoke the functions that are to be tested (either API or internal global/static function or to know the status of the modules). Test stubs with minimal functionality will be developed for the neighbouring modules, for the testing.

The test cases will be classified into several groups based on the functionality to be tested. One test application per group (e.g. SBC, CPLD) will be developed for executing the test cases belonging to that group and the results will be stored.

**Execute Test cases and Prepare SW Component** **Test Report (Functional Test Report):**

The testing will be carried out in the same order as mentioned in test plan using the code, Test application, generated c source and header files and required stubs. Generation (if applicable) of C source and Header files shall be done using module generator and ECU Configuration Description Files by development team (wherever applicable).

If applicable, testing will be carried out on a PC environment to verify the functionalities. Same test suite will be re-executed using target environment.

If the test case passes in testing, test report document that is generated using the test plan document as base, will be updated for actual output and test result to reflect the actual test observation. Test logs will be captured for all the test cases. If the test case fails and if the failure is due to some discontinuity and can be rectified by product modification, the impacted module will be corrected and all the functional test cases closely associated with that module will be re-run until the test case is successful.

All the module test cases shall run even for any changes done on the code including the bugfixes.

**Technical Review of SW Component Test Report:**

During this activity, the reviewer will check if the SW Component Test Report contains the test results for all the test cases present in the SW Component Test Specifications. The reviewer will check the actual outputs that are documented. The reviewer shall check the test configuration done by development team does not provide any warnings or errors while generating the c-source files.

During this activity, the test applications will be reviewed for correctness with regard to test specification, adequacy and understandability of comments, consistency across SW Components in terms of file and function organization.

Technical review checklist will be used to perform the review and report will be prepared.

**Update Safety Case:**

Update the Safety Case for Unit Test Plan and Unit Testing. This includes capturing evidence of safety activities carried out during this subphase.

**Software Component Integration Verification subphase:**

* Activity Description:

Prepare Safety Test Specification:

As a part of this activity following sub activities shall be carried out:

1. Analyze SW Requirements including SSR (300 Level), prepare Software Test Specification/Plan to test the SSR. Test Plan Generation includes Use Case Identification, Sequence Diagram Definitions/Understanding, Derive Test Case from Use Cases

Software test cases will be identified to cover the requirements provided in DNG. Based on the functionality, test cases will be grouped into several test groups. Each test group will have the purpose and corresponding configuration details that can be used to test the functionality of test cases. The test cases will cover Use cases (if any).

1. Analyze of SW Architecture Design and prepare Test plan to test interfaces within the module.

Interface test cases will be generated to cover the interfaces between the architectural components as well as outside system. For all test cases in the test plan, the traceability will be provided with SW Architecture design

Below are the methods for deriving test cases for software integration testing:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sl. No.** | **Methods** | **ASIL A** | **ASIL B** | **ASIL C** | **ASIL D** | **Argumentation** |
| 1a | Analysis of requirements | ++ | ++ | ++ | ++ | Test cases based on Architecture |
| 1b | Generation and analysis of equivalence classes | + | ++ | ++ | ++ | Most of them at unit level |
| 1c | Analysis of boundary values | + | ++ | ++ | ++ | Most of them at unit level and only few scenarios shall be covered at integration level |
| 1d | Error guessing based on knowledge or experience | + | + | + | + | Test cases based on lessons learnt process and experience |

**Table 17 Methods for Deriving test cases**

**(Ref: Table 11 ISO26262:6-10)**

Below are the methods for Structural coverage at the software architectural level:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sl. No.** | **Methods** | **ASIL A** | **ASIL B** | **ASIL C** | **ASIL D** | **Argumentation** |
| 1a | Function coverage | + | + | ++ | ++ | % coverage shall be mentioned and justified |
| 1b | Call coverage | + | + | ++ | ++ | % coverage shall be mentioned and justified |

**Table 18 Methods for Structural coverage**

**(Ref: Table 12 ISO26262:6-10)**

Methods for deriving test cases for the test of the embedded software (if applicable or partial):

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sl. No.** | **Methods** | **ASIL A** | **ASIL B** | **ASIL C** | **ASIL D** | **Argumentation** |
| 1a | Analysis of requirements | ++ | ++ | ++ | ++ | Test cases based on Architecture |
| 1b | Generation and analysis of equivalence classes | + | ++ | ++ | ++ | Most of them at unit level |
| 1c | Analysis of boundary values | + | + | ++ | ++ | Most of them at unit level and only few scenarios shall be covered at integration level |
| 1d | Error guessing based on knowledge or experience | + | + | ++ | ++ | Test cases based on lessons learnt process and experience |
| 1e | Analysis of functional dependencies | + | + | ++ | ++ |  |
| 1f | Analysis of operational use cases | + | ++ | ++ | ++ | Test cases based on Architecture |

**Table 19 Methods for deriving test cases**

**(Ref: Table 15 ISO26262:6-11)**

Review & Rework of Safety Test Specification:

During this activity, the Safety Test Specification prepared in the previous activity will be reviewed. Review is to ensure that:

* The Safety Test Specification covers all the requirements specified in the DNG
* The Safety Test Specification covers all the interfaces specified in the Architecture and Design
* Test cases have been identified for the functions defined in Design
* The correctness and adequacy of the test cases for performing the tests
* There are no duplicate and missing test cases
* Whether possible negative test cases are covered
* Whether possible Fault Injection test cases are covered

Scopes of Technical Review:

* If there are many changes in a particular document then the review scope will be the complete document.
* If the changes are major but they are pertaining to a particular section, then the review scope will be limited to that section.
* For documentation changes, only the changes will be reviewed.

Technical review checklist will be used to perform the review and report will be prepared.

Integration Testing:

Integration Test Environment:

This activity deals in identifying the test environment (like dbc, ECU extract etc.), configuration of modules and preparing test application as per the use cases, preparation of relevant test scripts for testing the stack.

Develop Test Application:

During this activity, Test applications which are required to develop the Safety Test Specifications will be developed VnV team (may be in some cases support from development team is required). This activity will be done in anticipation with the development of Safety Test Specifications.

**Integration Testing:**

**Execute Test Cases:**

Integrated stack should be tested for all the test cases identified in the Integration test plan and the results should be captured in the Integration test report with sufficient logs or snapshots. Any deviations in the actual test results from the expected test results shall be screened to identify the root cause and fix the affected product. If any fixes are done in the implementation, then all the affected test cases need to be rerun (regression testing) to ensure that the expected results are observed.

**Re-testing with bug fixed code**

If any of the test cases failed and bug is reported, once the bug is fixed, re-execution of the affected test cases shall be done and Test report shall be updated.

**Prepare Integration Test Report**

Based on the test status above activities Integrated Test report document need to be updated with the Pass/Fail status.

**Review and Rework of Integration Test Report:**

In this phase, and Integration Test Report shall be reviewed. Technical review is to ensure

* Whether all the test cases could be executed as per the test plan.
* Test results are provided for all the test cases with adequate logs / snapshots
* Test result is ‘PASS’ for all the test cases.

Technical review checklist will be used to perform the review and report will be prepared.

**Note:** Redmine shall be raised for the bugs. If fixes are not available / not working before the release, then the test case shall be failed and corresponding Redmine shall be captured in ITR against the test case

**Software Qualification Test:**

In this phase, test integrated software using the selected test cases (acceptance test cases). Record software test results and logs.

Test logs should be stored in Redmine that supports the test execution being done on various modules with no failures, prior to closure of the open Redmine issues.

The software integration shall be verified in accordance with ISO 26262-8:2018, Clause 9 by an appropriate combination of methods according to below table for verification of software integration:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sl. No.** | **Methods** | **ASIL A** | **ASIL B** | **ASIL C** | **ASIL D** | **Argumentation** |
| 1a | Requirements-based test | ++ | ++ | ++ | ++ | Test cases based on SSR, Architecture |
| 1b | Interface test | ++ | ++ | ++ | ++ | Interaction between modules at API level |
| 1c | Fault injection test | + | + | ++ | ++ | Create a negative test scenario and perform Fault injection test |
| 1d | Resource usage evaluation | ++ | ++ | ++ | ++ | Validate RAM, ROM and timing measurement on target Hardware |
| 1e | Back-to-back comparison test between model and code, if applicable | + | + | ++ | ++ | Not Applicable |
| 1f | Verification of the control flow and data flow | + | + | ++ | ++ | QAC/Polyspace shall be enabled for the integrated SW |
| 1g | Static code analysis | ++ | ++ | ++ | ++ | QAC/Polyspace shall be enabled for the integrated SW |
| 1h | Static analyses based on abstract interpretation | + | + | + | + | Not Applicable |

**Table 20** **Methods for verification of software integration**

**(Ref: Table 10 ISO26262:6-10)**

Test environments for conducting the software testing:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sl. No.** | **Methods** | **ASIL A** | **ASIL B** | **ASIL C** | **ASIL D** | **Argumentation** |
| 1a | Hardware-in-the-loop | ++ | ++ | ++ | ++ | Execution of integration testing on target microcontroller  (can be at KPIT if applicable or customer place) |
| 1b | Electronic control unit network environments | ++ | ++ | ++ | ++ | Not Applicable |
| 1c | Vehicles | + | + | ++ | ++ | Not Applicable |

**Table 21 Test environments for conducting the software testing (Ref: Table 13 ISO26262:6-11)**

Methods for tests of the embedded software:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sl. No.** | **Methods** | **ASIL A** | **ASIL B** | **ASIL C** | **ASIL D** | **Argumentation** |
| 1a | Requirements-based test | ++ | ++ | ++ | ++ | Test cases based on SSR, Architecture |
| 1b | Fault injection test | + | + | ++ | ++ | Create a negative test scenario and perform Fault injection test |

Table 22 Methods for tests of the embedded software

**(Ref: Table 14 ISO26262:6-11)**

**Update Safety Case:**

Update the Safety Case for Integration Test Plan and Integration Testing. This includes capturing evidence of safety activities carried out during this subphase.

**Confirmation Review and Rework of Safety Case Report:**

During this activity, the Safety case will be circulated to the Safety Consultant for review. The review comments will be incorporated in the Safety case and its associated documents. Documents will be sent for the rework verification. This includes capturing evidence of safety activities carried out during this subphase.

### Screen

|  |  |
| --- | --- |
| Type of Screen | Technical Screen |

* Products

|  |  |
| --- | --- |
| List of Upstream Products | 1. Project Proposal PPT 2. SDPS\_ INVERTER DESIGN 3. System Requirements Specification 4. CAN Signal List 5. Hardware software interface document (HSI) 6. Software Requirement Specification 7. Component Requirement Specification 8. MATLAB Models 9. Source code 10. CM checklist   **Software Component Verification subphase:**   1. SW Component design 2. SW Component Requirements (400 Level) 3. HSIS (refined) 4. SW Requirements including SSR (300 Level) 5. SW Architecture design 6. AUTOSAR Requirements (if applicable) 7. ISO26262 specifications   **Software Component Integration Verification subphase:**   1. System Requirements including TSR (200 Level) 2. SW Requirements including SSR (300 Level) 3. SW Architecture design 4. Technical Safety Concept (TSC) 5. HSIS 6. AUTOSAR Requirements (if applicable) 7. ISO26262 specifications 8. SW Component Code (in KPIT scope) 9. KSAR BSW + RTE integrated from Integration team 10. SW Component Generators (if applicable) 11. C4K Tool 12. KSAR BSW modules, SW Component User Manual (if applicable) 13. SW Component Configuration Requirements |
| List of Downstream Products | 1. ITP  2. ITR  3. PIL reports  4. Test Logs  5.LGM validation reports |
| List of Horizontal Products | 1. Integration Test Plan 2. Integrated SW Test Report 3. PIL test case and Report (Mapping of test case ID and DNG ID) 4. ^HIL test case release (Mapping of test case ID and DNG ID) 5. Integration test case design release 6. Supporting\_Documents .zip 7. Supporting\_document\_structure.xls 8. Test Report 9. LGM Validation Test Report 10. UTP 11. UTR 12. ITP 13. ITR |

* Screen Scope

Upstream View:

Verification & Validation Test Cases and Test Scripts are created as per system, software and hardware requirements

Traceability for test cases to requirement documents, Test cases to test scripts and test reports will be available in the DNG.

Downstream View:

Test Report for MIL/SIL/PIL testing and HIL test cases are complete and results are documented

Quality and Configuration Manager view:

The products are as per the KPIT Screen guidelines for Quality and CM View.

The product will follow CM guidelines for release process as well as with CM audit process.

Software Component Verification subphase:

**Upstream View:**

**SW Component UTP, SW Component Test Plan (Functional test Specification):**

SW Component UTP and SW Component Test Plan (Functional test Specification prepared is correct, consistent and complete in accordance with SW Component Design and code and SW Component functional requirements respectively.

The screener shall verify that the UTP and STS is prepared in accordance with the SW Component Design and code and functional requirements respectively. It shall also be verified that a test case exists for each function of the SW Component Design and code and functional requirements. Also, this view shall verify the compliance of the test plan with respect to the template.

**Downstream View:**

**SW Component UTP, SW Component Test Plan (Functional test Specification):**

SW Component can be configured, generated, compiled and build without any modifications to source files.

The screener shall check each of the test cases documented are clear such that a test application can be developed and can be executed easily.

**Quality & Configuration Management View:**

The screener shall check the products as per KPIT screen guidelines for CM and Quality view.

Software Component Integration Verification subphase:

**Upstream View:**

SW Component Integration Test Specification/Plan:

Test Plan prepared is correct, consistent and complete in accordance with AUTOSAR Requirements, Customer inputs and Acceptance Test Specification.

The screener shall verify that the system test plan is prepared in accordance with the requirements document. It shall also be verified that a test case exists for each of the functional requirements. Also, this view shall verify the compliance of the test plan with respect to the template.

Integration:

* All the modules that are listed in the project scope are present in the integrated SW.
* There are no compilation warnings or errors.
* There is no unused and unintended code in the stack.
* There are no duplicate or unused files in the stack.

Integration Testing:

* All the modules that are listed in the project scope are present in the integrated stack.
* There are no compilation warnings or errors.
* There is no unused and unintended code in the stack.
* There are no duplicate or unused files in the stack.
* All the test cases present in test plan is present in the Test Report.
* Test results are provided for all the test cases.

**Downstream View:**

SW Component Integration Test Specification/Plan:

Integrated SW can be configured, generated, compiled and build without any modifications to source files.

The screener shall check each of the test cases documented are clear such that a test application can be developed and can be executed easily.

Integration:

Integrated SW can be compiled and build without any modifications to source files.

Integration Testing:

* Stack can be compiled and build without any modifications to source files.
* The test environment has been clearly provided in the Test Report.
* The test report and logs are understandable.
* Failed Test cases to be provided with the remarks.

**Quality & Configuration Management View:**

The screener shall check the products as per KPIT screen guidelines for CM and Quality view. Screener shall also check the consistency across modules.

### Store

I/O/P/I - Input Product / Output Product / Pre-requisite Product / Intermediate Product

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| I/O/P/I | Product Name | CI Identifier | Source | Destination |
| I/-/-/- | Project Proposal PPT | Eaton\_NPe\_M2\_Dual\_core\_Inverter\_Program\_V2.0 | KPIT | VNV Uniphase |
| I/-/-/- | \*SDPS\_ INVERTER DESIGN | Eaton\_NPe\_M2\_\_SDPS\_ INVERTER DESIGN | Planning | VNV Uniphase |
| -/I/-/- | ITP\_Template | Eaton\_Inveter\_ITP\_Template | Planning | VNV Uniphase |
| -/I/-/- | ITR\_Template | Eaton\_Inveter\_ITR\_Template | Planning | VNV Uniphase |
| I/-/-/- | CDD Requirement Specification | Eaton\_NPe\_M3\_M2\_400L | Multicore Implementation for M2 Inverter | Customer, Multicore Implementation for M2 Inverter |
| I/-/-/- | AUTOSAR compliant Application Feature Models | Eaton\_NPe\_M2\_ AutoSAR compliant Application Feature Models | ASW | VNV Uniphase |
| I/-/-/- | ASW and BSW Integrated software | Eaton\_NPe\_M2\_NPE\_Integrated\_BSW\_ASW\_software | ASW and BSW | VNV Uniphase |
| I/-/-/- | CAN signal List / DBC | Eaton\_NPe\_M2\_CAN\_Signal\_List | System | VNV Uniphase |
| I/-/-/- | HSIS | Eaton\_NPe\_M2\_HSI Document | HARDWARE | VNV Uniphase |
| I/-/-/- | Fault reaction Plan | NPe\_M2\_Dual\_Core\_Systems\_Fault \_Reaction\_Plan | Systems uniphase | VNV Uniphase |
| I/-/-/- | SW Safety Analysis | Eaton\_NPe\_M2\_Dual\_Core\_SW Safety Analysis Report | Systems uniphase | VNV Uniphase |
| I/-/-/- | Systems Requirements (200L) | NPe\_M3\_M2\_Dual\_Core \_200\_Level\_System\_Requirement | Systems uniphase | VNV Uniphase |
| I/-/-/- | High Level SW Requirement (300L) | NPe\_M3\_M2\_Dual\_Core\_300\_Level\_HighLevel\_SW\_requirement | Systems uniphase | VNV Uniphase |
| I/-/-/- | Ranges for Inputs and outputs | Eaton\_NPe\_M2\_Ranges\_Input\_Output | Systems uniphase | VNV Uniphase |
| I/-/-/- | Internal Parameter List and Ranges | Eaton\_NPe\_M2\_Internal\_Parameter List\_ Ranges | Systems uniphase | VNV Uniphase |
| I/-/-/- | CM checklist V1.0 | R.0033170 \_M2inverter\_CM\_Checklist | Planning Uniphase | All uniphases |
| -/O/-/- | PIL test case and Report (Mapping of test case ID and DNG ID) | Eaton\_NPe\_M2\_Dual\_Core\_PIL\_test\_Report | VNV uniphase | Customer |
| -/O/-/- | ^HIL test case release (Mapping of test case ID and DNG ID) | Eaton\_NPe\_M2\_HIL\_test\_Report | VNV uniphase | Customer |
| -/O/-/- | Integration test case design release | Eaton\_NPe\_M2\_Dual\_Core\_Integration\_Test\_Design\_Release | VNV uniphase | Customer |
| -/O/-/- | Supporting\_Documents .zip | Eaton\_NPe\_M2\_Dual\_Core\_Supporting\_Dcument | VNV uniphase | Customer |
| -/O/-/- | Supporting\_document\_structure.xls | Eaton\_NPe\_M2\_Supporting\_Dcument\_Structure | VNV uniphase | Customer |
| -/O/-/- | Test Report | Eaton\_NPe\_M2\_Dual\_Core\_Test\_Report | VNV uniphase | Customer |
| -/O/-/- | LGM Validation Test Report | Eaton\_NPe\_M2\_Dual\_Core\_LGM\_Validation\_Test\_Report | VNV uniphase | Customer |
| -/O/-/- | UTP | Eaton\_NPe\_M2\_UTP | VNV uniphase | Customer |
| -/O/-/- | UTR | Eaton\_NPe\_M2\_UTR | VNV uniphase | Customer |
| -/O/-/- | ITP | Eaton\_NPe\_M2\_ITP | VNV uniphase | Customer |
| -/O/-/- | ITR | Eaton\_NPe\_M2\_ITR | VNV uniphase | Customer |

\* Incremental Product

Prefix \* if product is incremental. Eg: \*Design Document

^ Product applicable / Delivered earlier in M2 Single Core(Inverter\_Design) and the SDPS in which it is mentioned is stored at : SVN Path

<https://hjph3svn01.kpit.com/svn/R0032999/Program_Management/KPIT_Quality_Process/SDPS>

**Software Component Verification subphase:**

I/O/P/I – Input Product / Output Product / Pre-requisite Product / Intermediate Product

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| I/O/P/I | Product Name | CI Identifier | Source | Destination |
| I/-/-/- | \*SDPS\_ INVERTER DESIGN | Npe\_M2\_Dual\_Core\_SDPS | Planning | Customer |
| I/-/-/- | SW Component Architecture design | Eaton\_NPe\_M3\_M2\_SAD  Eaton\_NPe\_M3\_M2\_SDD | Software Component Design | Customer |
| I/-/-/- | SW Low Level Design | Eaton\_Npe\_M3\_M2\_LLD | Software Component Design | Customer |
| I/-/-/- | SSR | Eaton\_Npe\_M3\_M2\_Software\_Safety\_requirement | Software Safety Requirements | Customer |
| I/-/-/- | Software Component Requirement Specification\* | Eaton\_Npe\_M2\_Software\_Component\_requirement | Software Component Requirements | Customer |
| I/-/-/- | HSIS | Eaton\_Npe\_M2\_HSIS | Customer | Customer |
| I/-/-/- | Software Architecture | Eaton\_Npe\_M3\_M2\_Software Architecture Document | Software Architecture | Customer |
| -/O/-/- | SW Component UTP (code coverage test specification) | Eaton\_Npe\_M2\_UTP | Software Coding and Testing | Customer |
| -/O/-/- | SW Component (UTR) Testing report | Eaton\_Npe\_M2\_UTR | Software Coding and Testing | Customer |
| -/O/-/- | CBA (Static code analysis) Report | Eaton\_Npe\_M2\_CBA report | Software Coding and Testing | Customer |

\* Incremental Product

Prefix \* if product is incremental. Eg: \*Design Document

**Software Component Integration Verification subphase:**

I/O/P/I – Input Product / Output Product / Pre-requisite Product / Intermediate Product

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| I/O/P/I | Product Name | CI Identifier | Source | Destination |
| I/-/-/- | \*SDPS\_ INVERTER DESIGN | Npe\_M2\_Dual\_Core\_SDPS | Planning | Customer |
| I/-/-/- | TSR | Eaton\_Npe\_M2\_Technical\_Safety\_requirement | Customer | Customer |
| I/-/-/- | Software Requirement Specification\* | Eaton\_Npe\_M3\_M2\_Software\_Requirement Specification\* | Software Requirements | Customer |
| I/-/-/- | SSR | Eaton\_Npe\_M3\_M2\_Software\_Safety\_requirement | Software Requirements | Customer |
| I/-/-/- | Software Architecture | Eaton\_NPe\_M3\_M2\_Dualcore\_Architecture | Software Architecture | Customer |
| I/-/-/- | HSIS | Eaton\_Npe\_M2\_HSIS | Customer | Customer |
| I/-/-/- | SW Component User Manual | Eaton\_Npe\_M2\_UM | Software Component User Manual | Customer |
| I/-/-/- | SW Component Safety User Manual | Eaton\_Npe\_M2\_SUM | Software Component User Manual | Customer |
| -/O/-/- | Integration Test Plan | Eaton\_Npe\_M2\_ITP | Software Component Integration and Verification | Customer |
| -/O/-/- | Integrated SW Test Report | Eaton\_Npe\_M2\_ITR | Software Component Integration and Verification | Customer |

# Configuration Management

## Strategy

Roles can be added and customized according to project needs

The list of configuration management activities and RASIC is detailed in below table.

Separate CM plan will be maintained in the SVN:

<https://hjph3svn01.kpit.com/svn/R0032999/Program_Management/KPIT_Quality_Process/SDPS/Configuration_Management_Plan.pptx>

| Configuration Management Activities |  | |  |  | RASIC | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| DM | Prg.M | | PL/PM | | SM | TM | CM | SC | PC |
| Plan for Configuration Management | A | R | | R | | I | I | S | S | S |
| Identify Configurable Items | I | A | | R | | C | S | S | C | C |
| Perform Daily/Frequent Commits | I | I | | A | | I | R | S | I | I |
| Establish Baselines | I | I | | A | | C | S | R | C | C |
| Perform Branching | I | I | | A | | I | C | R | S | S |
| Perform Merging | I | I | | A | | I | S | R | C | C |
| Perform Build Process | I | I | | A | | I | S | R | C | C |
| Maintain Cis Status (in Unipro) | I | I | | A | | I | S | R | C | C |
| Perform Configuration Status Reporting | I | A | | R | | I | I | S | C | C |
| Perform Configuration Audits | I | A | | R | | C | S | Ss | S | S |
| Provide Support & Trainings | I | A | | R | | S | I | R | S | S |
| Provide Access & Permissions | I | A | | R | | I | I | R | S | S |

R – Responsible, A – Accountable, S – Support, I – Informed, C- Consulted

### Version Numbering and Naming Convention

* Version Numbering

Following versioning guidelines shall be followed for versioning the Configuration Items. There shall be two digits version convention followed for all configuration Items as given below

X.Y

Where, X denotes a major change and Y denotes a minor change.

1. Draft versions (not baselined) of the product can use 0.1, 0.2, 0.3 … …
2. First baseline of the item shall use 1.0
3. Subsequent minor changes to the item shall use 1.1, 1.2, 1.3 … …
4. Major change to the configuration item shall use 2.0, 3.0 … …

Major Change: <Define major change along with approval authority in the context of this project>

Minor Change: <Define minor change along with approval authority in the context of this project>

* Naming Conventions

Every Configuration Identifier shall contain three parts as explained below

Npe\_M2\_Product name

Where,

1. “PROJID” defines a short code of the project which shall be prefixed for all Config IDs
2. ITEMTYPE defines a short code for the type of the configuration item being defined.
3. COUNTER defines unique counters to distinguish Config IDs of same ITEMTYPE.
4. Example of Configuration Identifier, ABCD\_SRSDC\_001, ABCD\_DESDC\_002, etc.

* Release Naming Conventions

Software: - As per customer release naming convention, V&V: - V<1, 2, ….> System: - S<X.Y> FUSA: - F<X.Y, ….>

### Baseline

An output work product of a Uniphase is referred as “baseline” when it has been screened and agreed upon and shall be changed only through formal change control procedures. A baselined item cannot be changed without the approval of its designated approval authority (CCB).

Baseline of work product has been kept in Release folder, once all deliverables are ready it has been merging into trunk from branch and then baseline into Release folder.

| Output Work Product | Baseline Criteria |
| --- | --- |
| All Work Products | Screen rework completion and storage of product is a baseline criteria. |

### Branching and Merging

* Development Branches

KPIT’s Store Guidelines will be adhered for developing and maintaining Development Branches.

* Variant Management
* At the time of modules integration, module lead / integration team member will do the merging of code to mainstream.
* At the time of any release milestone, configuration manager / integration lead / integration team member needs to merge the branch to trunk.
* Configuration Manager will create release TAG and keep the deliverables into the Eaton folder.
* Configuration Manager / Lead Integrator will create the next branch to work.
* Trunk will always have the latest released code.
* Once the new branch is created, everyone should start working on this branch

### Software Build Process

NA

## List of Configuration Items

|  |  |
| --- | --- |
| Configurable Items | <https://hjph3svn01.kpit.com/svn/R0032999/Program_Management/KPIT_Quality_Process/SDPS/Configuration_Management_Plan.pptx>   * Change Management Plan * Configuration Management Plan * Hardware Requirement Specification * Hardware Design Document * Schematics * Gerber files * DFMEA reports * Software Requirement Specification * Software Design Document * Architecture document * Source Code * Test Plans * PO * Proposal * MOM * PSR * Project Plans * SDPS * SDPS notes * MPP * PMRP |
| Non-Configurable Items | * Weekly Status Report(s) * Monthly Status Report(s) * MLR Report(s) * Postmortem Report(s) * CSAT and RCA Report(s) * QA Reports(s) * Review strategy Document |

## Configuration Environment

|  |  |
| --- | --- |
| CM Tool | GIT |
| Project Repository | <https://gitlab.kpit.com/epowertrain/inverter_projects/m2_vc0.a> |
| SAR Report Frequency | During release |
| Retention Period | Till project ends |
| Backup Frequency | Monthly |
| Guidelines | <https://sdpm.kpit.com/KPIT.SDPM/guidances/guidelines/git_guidelines_D6BF9028.html?nodeId=ac1caa2a> |

* Project folder structure

Graphical user interface, text, application

Description automatically generated

* Trunk Folder structure:

Graphical user interface, text, application

Description automatically generated

* Program Management folder structure is as follows:

Graphical user interface, text, application

Description automatically generated

* Access and Permissions:

The access rights provide below is an example, this can change based on the CM Strategy, Tools used and the folder structure

|  |  |  |  |
| --- | --- | --- | --- |
| Role | Trunk | Branches | Tags |
| Configuration Manager | Write, Admin | Write, Admin | Write, Admin |
| Delivery Manager | Read | Read | Read |
| Program Manager | Read | Write | Read |
| Project Manager/Project Leader | Read | Write | Read |
| Technical Lead | Read | Write | Read |
| Process Consultant | Read | Read | Read |
| Team Members | Read | Write | Read |
| Safety Manager | Read | Write | Read |

## Change Management

Roles can be added and customized according to project needs

The list of change control activities and RASIC is detailed in below table.

Change management process is covered in attached PPT in appendix section : <Fileame>

| Change Control Activities |  | | RASIC | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| DM | Prg.M | | PM/PL | TM | CCB | CM | PC |
| Initiate Software Change Request | I | A | | R | R | I | R | S |
| Impact Analysis | I | A | | R | S | I | I | C |
| Schedule CCB | I | A | | S | I | I | R | C |
| Approve SCRs | A | R | | R | C | R | S | C |
| SCR Implementation and Closure | I | A | | R | S | I | S | C |

R – Responsible, A – Accountable, S – Support, I – Informed, C- Consulted

* Change Management Tool

|  |  |
| --- | --- |
| Tool Name | Unipro |
| Tool Location/URL | <https://unipro.kpit.com/> |

* CCB Members

|  |  |
| --- | --- |
| Class A – Change Control Board Members | CCB chairperson – Delivery Manager  CCB Members – Program Manager, Project manager/Project leader/ Scrum Master, Process consultant, Configuration manager, SCR Initiator, CBA Head (for any change related to CBA parameter and its threshold value) and other stakeholder as required |
| Class B – Change Control Board Members | CCB Chairperson – Program Manager  CCB Members – Project Manager/Project leader/Scrum Master, Process consultant, Configuration manager, SCR initiator and others stakeholder as required |

## Problem Management

Roles can be added and customized according to project needs

The list of Problem Management activities and RASIC is detailed in below table.

| Problem Management Activities |  | RASIC | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| DM | Prg.M | PM/PL | TM | CM | PC |
| Initiate Problem Request | I | A | R | R | R | S |
| Problem Acknowledgement & Analysis | I | A | R | S | I | C |
| Problem Resolution | I | A | R | I | S | C |
| Problem Review & Closure | I | A | R | C | I | S |
| Problem Trends Analysis | A | R | R | S | I | S |

R – Responsible, A – Accountable, S – Support, I – Informed, C- Consulted

* Problem Management Tool

|  |  |
| --- | --- |
| **Internal Review Defects** | Maintained in Unipro |
| **Internal Screen Defects** | UNIPRO – Screen Management Module |
| **Internal Testing Defects** | All Internal Testing Defects shall be tracked in KIT |
| **External/Customer Reported Defects** | KIT |

* Problem Management SLA

|  |  |
| --- | --- |
| Parameter | SLA |
| <https://hjph3svn01.kpit.com/svn/R0032999/Program_Management/KPIT_Quality_Process/SDPS/Management_and_Control_Plan.pptx>  Please refer above path for SLA | - |

# Quality Assurance

## QA Strategy

Quality Assurance is a set of activities through-out the project that defines, implements and assess the processes to ensure quality of the output work products.

While Testing is a dedicated activity to check conformance to requirements, Quality Control include activities to evaluate and control the quality of output work products; Quality Assurance is a pro-active activity to

1. Define the strategy and processes for all quality related activities (Including Quality Control, Reviews, Testing, etc.)
2. Ensure conformance of process implementation to the defined processes
3. Ensure conformance of product implementation to the defined standards and templates.

### Independence

Quality Assurance is performed through-out the project and all project team members are responsible to ensure Quality of the work products. However, to bring independence in the assessment of Quality Assurance, Process Consultants are identified from the PEG Team.

Independence is ensured by dividing the responsibilities of QA Activities between project team and PEG as explained below.

|  |  |
| --- | --- |
| Project Team | PEG Team |
| Define QA Strategy and processes. | Review and ensure correctness and completeness of the QA Strategy and Processes |
| Execute the project in accordance with the process defined in Project Plan. | Implement Work Process related QA Activities to ensure project execution (By Project Team) adheres to the defined processes and standards. |
| Implement Work Product related QA Activities to ensure adherence to standards and requirements | Ensure Work Product related QA Activities are implemented as per QA Plan |

### Note: To bring independence in work product related quality assurance, KPIT screen process shall be followed where Independent screen team is identified for final VNV. Also, Process Consultant from PEG shall participate by taking the Quality and CM View for technical screensStandards and tools

### STANDARDS AND TOOLS

This section describes the standards and tools employed to perform quality assurance activities.

* Standards

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Name | Source | Version | Date |
| 1 | KPIT Screen Guidelines | sdpm.kpit.com | 1.7 | 21-Apr-2020 |

* Tools

|  |  |  |
| --- | --- | --- |
| # | Tool/Technique Used | Purpose or Rationale for using the tool |
| 1 | Unipro | To perform Screen, Raise & manage Process Improvements. |
| 2 | Drona | CSAT |
|  |  |  |

\*Provide references to SDPS where work products related standards and tools are defined.

## QA Activities, R&R

Roles & activities can be added and customized according to project needs.

The list of Quality Assurance activities and RASIC is detailed in below table.

| Quality Assurance Activities |  | |  |  | RASIC | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| DM | Prg.M | | PM | | SM | QM | SC | PC |
| Defect Prevention Activities | | | | | | | | | |
| Project Initiation (customer) | I | A | | R | | I | I | C | C |
| Project Kickoff (internal) | I | A | | R | | C | C | I | I |
| Uniphase Kickoff | I | I | | R | | I | I | C | C |
| RCA | I | A | | R | | C | C | C | C |
| SDPS Review | I | A | | R | | R | R | C | C |
| Postmortem | I | A | | R | | C | C | C | C |
| Process Improvement | I | I | | R | | I | I | C | C |
| Lessons Learnt | I | A | | R | | I | I | I | I |
| Work Products related | | | | | | | | | |
| Technical Reviews | I | A | | R | | I | I | S | S |
| Screen | I | A | | R | | I | I | C | C |
| Technical Audit | I | S | | R | | I | I | C | C |
| Work process related | | | | | | | | | |
| Process Compliance Checks & Red Flag Reporting | I | I | | A | | S | S | R | R |
| Quality Audit | I | S | | R | | A | A | C | C |
| Release Audit (for each release) | I | A | | R | | I | I | R | R |
| Safety Audit | I | S | | R | | I | I | C | C |

R – Responsible, A – Accountable, S – Support, I – Informed, C- Consulted

### Reviews and Testing

Work Product reviews to ensure product quality are planned in Sec. 4.4 and 4.5.

Please refer to Individual Uniphase(s) for details of Reviews & Screen

## Red Flag Reporting Mechanism

Red Flags are the process non-compliances identified by the process consultants during the course of Project.

|  |
| --- |
| **Step 1:** |
| * Process Consultant shall identify and report the red flag to PL/PM. * PL/PM shall resolve the red flags at the earliest. |
| Step 2: |
| * If the Red Flag is not resolved by PL/PM, Process Consultant shall report the red flags to Program Manager. * Program Manager shall take necessary actions to resolve the Red Flag. |
| Step 3: |
| * If the Red Flag is not resolved, PEG shall report the Red Flag to Delivery Manager. * Delivery Manager shall take necessary actions to resolve the Red Flag. |
| Step 4: |
| * If the Red Flag is not resolved, PEG shall report the Red Flag to Business Leader. * Business Leader shall take necessary actions to resolve the Red Flag.   Note: If Red Flags are not resolved at this level, SBU Head shall be involved as needed. SBU Head has the ultimate authority in case of Conflicts. |

The steps above explain the hierarchy of Red Flag resolution. Red Flags can be reported for a project, group of project or department; either in a single report or multiple reports

## Timing Plan

|  |  |  |
| --- | --- | --- |
| Audits | | |
| Quality Audit Timing | 1-Jan-2022 |  |
| Technical Audit Timing | 25-Jan-2022 |  |
| Rationale, if not planned |  | |
| Remarks | NA | |
| Retrospection | | |
| Participant Details | All Project Team Members | |
| Retrospection Date | 26-Feb-2021 | |
| Frequency | Monthly | |
| Lessons Learned | | |
| Participant Details | All Project Team Members | |
| Frequency | As needed | |
| Process Improvement | | |
| Process Improvement Goals | One per quarter | |
| Process Compliance Checks & Red Flag Reporting | | |
| Timing/Frequency | Weekly | |
| Project Kickoff (Internal) | | |
| Participants | * Project Leader/Project Manager * Program Manager * Project Team Member * Two Delivery Managers * Practice Team Representative * Quality Manager * Support Organization Representatives * Business Lead (Optical) | |
| Timing/Frequency | At the Beginning of the Project | |
| Project Initiation (Customer) | | |
| Participants | * Customer’s Team * Project Leader/Project Manager * Program Manager * Project Team Members * Process Consultant (Optional) * Business Lead (Optional) | |
| Timing/Frequency | At the Beginning of the Project | |
| Uniphase Kickoff | | |
| Participants | * Project Leader/Project Manager * Project Team Members * Process Consultant * Program Manager (Optional) | |
| Timing/Frequency | At the beginning of each Uniphase.  (Optional) At the beginning of each Cycle of Uniphase | |
| Note: | Milestone Kick off will be done instead of Uniphase kick off | |
| RCA | | |
| Participants | * Project Leader/Project Manager * RCA Team (Identified at the beginning of RCA) * Process Consultant | |
| Timing/Frequency | On need basis. | |
| Triggers for conducting RCA | * Low CSAT * Metric Variance of 10% * Customer Reported Defects * Customer Escalation * Work packet RCA strategy | |
| SDPS Review | | |
| Participants | * Project Leader/Project Manager * Process Consultant * Other stakeholders, if required | |
| Timing/Frequency | Before it is screened. | |
| Technical Reviews | | |
| Participants | * Project Leader/Project Manager * Project Team Members | |
| Timing/Frequency | As per the project timing plan. | |

# EPMN Definitions, Abbreviations and Acronyms

* EPMN Definitions

|  |  |  |  |
| --- | --- | --- | --- |
| Objects | | Gates | |
|  | **Uniphase** is a small manageable phase of a process. It is a group of related activities, artefacts and guidance(s) with a common purpose. |  | **Start Gate** is a declaration for a sequence of engineering process which can be composed of Uniphase(s) and/or Activities. |
|  | An **Activity** of project, which can be planned, assigned and tracked. |  | **End Gate** depicts the end/finish of a sequence of activities or Uniphase. |
|  | **Artefact** is an outcome of engineering activity (ies). It can be an input or output in the engineering process. |  | **AND Gate** splits the sequence of activities into multiple paths. All paths shall be traversed. |
|  | **Guidance** is an enabler to perform the activities. A Guidance can be in the form of template, guidelines, checklist, form, instructions manual, training material etc. |  | **OR Gate** splits the sequence of activities into multiple paths. One or more paths can be traversed depending on the conditions. |
|  | **Role** represents a personnel resource or department. A Role can be assigned an activity with following RASIC relationship. “Responsible”/“Accountable”/“Support”/ “Informed”/“Consulted” |  | **XOR Gate** splits the sequence of activities into multiple paths. But only one path shall be traversed depending on the condition. |
| Relationships | | Others | |
|  | **Control Flow** depicts the sequence of activities and/or Uniphase(s). It represents the path(s) of process execution. |  | A **Note** provide textual explanation in an EPMN diagram. |
|  | **Data Flow** depicts the data in form of artefacts to flow between Activities or Uniphase(s). |  | **Lane** of association, all objects in a lane will have (or imply) a default relationship with each other. |
|  | **RASIC flow** establish the relationship between a Role and an Activity. |  |  |

* Alternate Representations

|  |  |  |  |
| --- | --- | --- | --- |
| Objects | | Others | |
|  | An **iterative Uniphase** is executed more than once in a Process. |  | A **pre-requisite connection** requires the predecessor Activity, Uniphase or artefact to be completed before starting the next Activity or Uniphase |
|  | An **iterative activity** is executed more than once for each iteration of the Uniphase. |  | An **Input Data flow** is depicted using a filled circle at the connection. |
|  | A **circular role** helps in easy diagramming and simplification of views. |  | An **Output Data flow** is depicted using an empty circle at the connection. |

* Abbreviation and Acronyms

|  |  |
| --- | --- |
| Acronyms/ Abbreviation | Description |
| BO | Business Objectives |
| CBA | Continuous Build Automation |
| CCB | Change Control Board |
| CI | Configuration Identifier |
| CM | Configuration Manager |
| COTS | Commercial Off-The-Shelf |
| CSAT | Customer SATisfaction |
| DAR | Decision Analysis and Resolution |
| DE | Discontinuities Escaped |
| DF | Discontinuities Found |
| EPMN | Engineering Process Modelling Notation |
| FC | Forced Change |
| KPI | Key Performance Indicator |
| LSL | Lower Specification Limit |
| MLR-L1 | Multi Level Review-Level 1 |
| PC | Process Consultant |
| PEG | Process Excellence Group |
| PO | Purchase Order |
| PM | Project Manager/Project Leader |
| PPM | Process Performance Model |
| Prg.M | Program Manager |
| QA | Quality Assurance |
| RCA | Root Cause Analysis |
| RE | Rework Effort |
| SAR | Store Activity Report |
| SCR | Software Change Request |
| SDPM | Software Development Process Methodology |
| SDPS | Software Development Process Specification |
| SLA | Service Level Agreement |
| SOW | Statement of Work |
| UC | Unforced Change |
| USL | Upper Specification Limit |
| WBD | Wide Band Delphi |
| WE | Work Effort |

# APPENDIX

## Appendix A: ASIL METHODS IN THE PROJECT

The different methods listed in a table contribute to the level of confidence in achieving compliance with the corresponding requirement. Each method in a table is either

1. A consecutive entry (marked by a sequence number in the leftmost column, e.g. 1, 2, 3), or
2. An alternative entry (marked by a number followed by a letter in the leftmost column, e.g. 2a, 2b, 2c).

For consecutive entries, all methods shall be applied as recommended in accordance with the ASIL. If methods other than those listed are to be applied, a rationale shall be given that these fulfil the corresponding requirement.

For alternative entries, an appropriate combination of methods shall be applied in accordance with the ASIL indicated, independent of whether they are listed in the table or not. If methods are listed with different degrees of recommendation for an ASIL, the methods with the higher recommendation should be preferred. A rationale shall be given that the selected combination of methods complies with the corresponding requirement.

For each method, the degree of recommendation to use the corresponding method depends on the ASIL and is categorized as follows:

|  |  |
| --- | --- |
| “++” | Highly Recommended |
| “+” | Recommended |
| “o” | Optional / No recommendation |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Specifying Safety Requirements | | ASIL | ASIL | ASIL | ASIL | Used in Project?  (Yes/No) | Justification, in case if the Response is “No” for methods which are + and ++ |
| Methods | | A | B | C | D |  |  |
| 1a | Informal notations | ++ | ++ | + | + | Yes |  |
| 1b | Semi-formal notations | + | + | ++ | ++ | Yes |  |
| 1c | Formal notations | + | + | + | + | Yes |  |
| Notations for software architectural design | |  |  |  |  |  |  |
| Methods | | A | B | C | D |  |  |
| 1a | Informal notations | ++ | ++ | + | + | NA |  |
| 1b | Semi-formal notations | + | ++ | ++ | ++ | NA |  |
| 1c | Formal notations | + | + | + | + | NA |  |
| Principles for software architectural design | |  |  |  |  |  |  |
| Methods | | A | B | C | D |  |  |
| 1a | Hierarchical structure of software components | ++ | ++ | ++ | ++ | NA |  |
| 1b | Restricted size of software components – a | ++ | ++ | ++ | ++ | NA |  |
| 1c | Restricted size of interfaces – a | + | + | + | + | NA |  |
| 1d | High cohesion within each software component- b | + | ++ | ++ | ++ | NA |  |
| 1e | Restricted coupling between software components- a, b, c | + | ++ | ++ | ++ | NA |  |
| 1f | Appropriate scheduling properties | ++ | ++ | ++ | ++ | NA |  |
| 1g | Restricted use of interrupts- a, d | + | + | + | ++ | NA |  |
| a – In methods 1b, 1c, 1e and 1g “restricted” means to minimize in balance with other design considerations. | |  |  |  |  |  |  |
| B – Methods 1d and 1e can, for example, be achieved by separation of concerns which refers to the ability to identify, encapsulate, and manipulate those parts of software that are relevant to a particular concept, goal, task, or purpose. | |  |  |  |  |  |  |
| C – Method 1e addresses the limitation of the external coupling of software components. | |  |  |  |  |  |  |
| D – Any interrupts used have to be priority-based. | |  |  |  |  |  |  |
| Mechanisms for error detection at the software architectural level | |  |  |  |  |  |  |
| Methods | | A | B | C | D |  |  |
| 1a | Range checks of input and output data | ++ | ++ | ++ | ++ | NA |  |
| 1b | Plausibility check – a | + | + | + | ++ | NA |  |
| 1c | Detection of data errors – b | + | + | + | + | NA |  |
| 1d | External monitoring facility – c | o | + | + | ++ | NA |  |
| 1e | Control flow monitoring | o | + | ++ | ++ | NA |  |
| 1f | Diverse software design | o | o | + | ++ | NA |  |
| a – Plausibility checks can include using a reference model of the desired behavior, assertion checks, or comparing signals from different sources. | |  |  |  |  |  |  |
| B – Types of methods that may be used to detect data errors include error detecting codes and multiple data storage. | |  |  |  |  |  |  |
| C – An external monitoring facility can be, for example, an ASIC or another software element performing a watchdog function. | |  |  |  |  |  |  |
| Mechanisms for error handling at the software architectural level | |  |  |  |  |  |  |
| Methods | | A | B | C | D |  |  |
| 1a | Static recovery mechanism – a | + | + | + | + | NA |  |
| 1b | Graceful degradation – b | + | + | ++ | ++ | NA |  |
| 1c | Independent parallel redundancy- c | o | o | + | ++ | NA |  |
| 1d | Correcting codes for data | + | + | + | + | NA |  |
| a – Static recovery mechanisms can include the use of recovery blocks, backward recovery, forward recovery and recovery through repetition. | |  |  |  |  |  |  |
| B – Graceful degradation at the software level refers to prioritizing functions to minimize the adverse effects of potential failures on functional safety. | |  |  |  |  |  |  |
| C – Independent parallel redundancy can be realized as dissimilar software in each parallel path. | |  |  |  |  |  |  |
| Notations for software unit design | |  |  |  |  |  |  |
| Methods | | A | B | C | D |  |  |
| 1a | Natural language | ++ | ++ | ++ | ++ | NA |  |
| 1b | Informal notations | ++ | ++ | + | + | NA |  |
| 1c | Semi-formal notations | + | ++ | ++ | ++ | NA |  |
| 1d | Formal notations | + | + | + | + | NA |  |
| Design principles for software unit design and implementation | |  |  |  |  |  |  |
| Methods | | A | B | C | D |  |  |
| 1a | One entry and one exit point in subprograms and functions – a | ++ | ++ | ++ | ++ | NA |  |
| 1b | No dynamic objects or variables, or else online test during their creation- a, b | + | ++ | ++ | ++ | NA |  |
| 1c | Initialization of variables | ++ | ++ | ++ | ++ | NA |  |
| 1d | No multiple use of variable names- a | + | ++ | ++ | ++ | NA |  |
| 1e | Avoid global variables or else justify their usage- a | + | + | ++ | ++ | NA |  |
| 1f | Limited use of pointers- a | o | + | + | ++ | NA |  |
| 1g | No implicit type conversions- a, b | + | ++ | ++ | ++ | NA |  |
| 1h | No hidden data flow or control flow- c | + | ++ | ++ | ++ | NA |  |
| 1i | No unconditional jumps- a, b, c | ++ | ++ | ++ | ++ | NA |  |
| 1j | No recursions | + | + | ++ | ++ | NA |  |
| a – Methods 1a, 1b, 1d, 1e, 1f, 1g and 1i may not be applicable for graphical modelling notations used in model-based development. | |  |  |  |  |  |  |
| B – Methods 1g and 1i are not applicable in assembler programming. | |  |  |  |  |  |  |
| C – Methods 1h and 1i reduce the potential for modelling data flow and control flow through jumps or global variables. | |  |  |  |  |  |  |
| Methods for software unit testing | |  |  |  |  |  |  |
| Methods | | A | B | C | D |  |  |
| 1a | Requirements-based test – a | ++ | ++ | ++ | ++ | NA |  |
| 1b | Interface test | ++ | ++ | ++ | ++ | NA |  |
| 1c | Fault injection test – b | + | + | ++ | ++ | NA |  |
| 1d | Resource usage test – c | + | + | + | ++ | NA |  |
| 1e | Back-to-back comparison test between model and code, if applicable – d | + | + | ++ | ++ | NA |  |
| a – The software requirements at the unit level are the basis for this requirements-based test. | |  |  |  |  |  |  |
| B – This includes injection of arbitrary faults (e.g. by corrupting values of variables, by introducing code mutations, or by corrupting values of CPU registers). | |  |  |  |  |  |  |
| C – Some aspects of the resource usage test can only be evaluated properly when the software unit tests are executed on the target hardware or if the emulator for the target processor supports resource usage tests. | |  |  |  |  |  |  |
| D – This method requires a model that can simulate the functionality of the software units. Here, the model and code are stimulated in the same way and results compared with each other. | |  |  |  |  |  |  |
| Methods for deriving test cases for software unit testing | |  |  |  |  |  |  |
| Methods | | A | B | C | D |  |  |
| 1a | Analysis of requirements | ++ | ++ | ++ | ++ | NA |  |
| 1b | Generation and analysis of equivalence classes – a | + | ++ | ++ | ++ | NA |  |
| 1c | Analysis of boundary values – b | + | ++ | ++ | ++ | NA |  |
| 1d | Error guessing – c | + | + | + | + | NA |  |
| a – Equivalence classes can be identified based on the division of inputs and outputs, such that a representative test value can be selected for each class. | |  |  |  |  |  |  |
| B – This method applies to interfaces, values approaching and crossing the boundaries and out of range values. | |  |  |  |  |  |  |
| C – Error guessing tests can be based on data collected through a “lessons learned” process and expert judgment. | |  |  |  |  |  |  |
| Structural coverage metrics at the software unit level | |  |  |  |  |  |  |
| Methods | | A | B | C | D |  |  |
| 1a | Statement coverage | ++ | ++ | + | + | NA |  |
| 1b | Branch coverage | + | ++ | ++ | ++ | NA |  |
| 1c | MC/DC (Modified Condition/Decision Coverage) | + | + | + | ++ | NA |  |
| Methods for software integration testing | |  |  |  |  |  |  |
| Methods | | A | B | C | D |  |  |
| 1a | Requirements-based test – a | ++ | ++ | ++ | ++ | NA |  |
| 1b | Interface test | ++ | ++ | ++ | ++ | NA |  |
| 1c | Fault injection test – b | + | + | ++ | ++ | NA |  |
| 1d | Resource usage test – cd | + | + | + | ++ | NA |  |
| 1e | Back-to-back comparison test between model and code, if applicable – e | + | + | ++ | ++ | NA |  |
| a – The software requirements at the architectural level are the basis for this requirements-based test. | |  |  |  |  |  |  |
| B – This includes injection of arbitrary faults in order to test safety mechanisms (e.g. by corrupting software or hardware components). | |  |  |  |  |  |  |
| C – To ensure the fulfilment of requirements influenced by the hardware architectural design with sufficient tolerance, properties such as average and maximum processor performance, minimum or maximum execution times, storage usage (e.g. RAM for stack and heap, ROM for program and data) and the bandwidth of communication links (e.g. data buses) have to be determined. | |  |  |  |  |  |  |
| D – Some aspects of the resource usage test can only be evaluated properly when the software integration tests are executed on the target hardware or if the emulator for the target processor supports resource usage tests. | |  |  |  |  |  |  |
| E – This method requires a model that can simulate the functionality of the software components. Here, the model and code are stimulated in the same way and results compared with each other. | |  |  |  |  |  |  |
| Methods for deriving test cases for software integration testing | |  |  |  |  |  |  |
| Methods | | A | B | C | D |  |  |
| 1a | Analysis of requirements | ++ | ++ | ++ | ++ | NA |  |
| 1b | Generation and analysis of equivalence classes – a | + | ++ | ++ | ++ | NA |  |
| 1c | Analysis of boundary values – b | + | ++ | ++ | ++ | NA |  |
| 1d | Error guessing – c | + | + | + | + | NA |  |
| a – Equivalence classes can be identified based on the division of inputs and outputs, such that a representative test value can be selected for each class. | |  |  |  |  |  |  |
| B – This method applies to parameters or variables, values approaching and crossing the boundaries and out of range values. | |  |  |  |  |  |  |
| C – Error guessing tests can be based on data collected through a “lessons learned” process and expert judgment. | |  |  |  |  |  |  |
| Structural coverage metrics at the software architectural level | |  |  |  |  |  |  |
| Methods | | A | B | C | D |  |  |
| 1a | Function coverage – a | + | + | ++ | ++ | NA |  |
| 1b | Call coverage – b | + | + | ++ | ++ | NA |  |
| a – Method 1a refers to the percentage of executed software functions. This evidence can be achieved by an appropriate software integration strategy. | |  |  |  |  |  |  |
| B – Method 1b refers to the percentage of executed software function calls. | |  |  |  |  |  |  |
| Tool classified as TCL3 | |  |  |  |  |  |  |
| Methods | | A | B | C | D |  |  |
| 1a | Increased confidence from use | ++ | ++ | + | + | Yes |  |
| 1b | Evaluation of tool development process | ++ | ++ | + | + | Yes |  |
| 1c | Validation of the software tool | + | + | ++ | ++ | Yes |  |
| 1d | Development in accordance with a safety standard | + | + | ++ | ++ | Yes |  |
| Tool classified as TCL2 | |  |  |  |  |  |  |
| Methods | | A | B | C | D |  |  |
| 1a | Increased confidence from use | ++ | ++ | ++ | + | Yes |  |
| 1b | Evaluation of tool development process | ++ | ++ | ++ | + | Yes |  |
| 1c | Validation of the software tool | + | + | + | ++ | Yes |  |
| 1d | Development in accordance with a safety standard | + | + | + | ++ | Yes |  |

## Appendix B: EPMN File



## APPENDIX C: SUPPORTING DOCUMENTS

* Npe M2Milestone Plan: (CR will be required in the Unipro if any changes to milestone plan)

[https://hjph3svn01.kpit.com/svn/R0032999/Program\_Management/KPIT\_Commercial\_Documents/Proposals/Milestone Chart – Eaton M2 (VF-1.0a)Program V8.xlsm](https://hjph3svn01.kpit.com/svn/R0032999/Program_Management/KPIT_Commercial_Documents/Proposals/Milestone%20Chart%20-%20Eaton%20M2%20(VF-1.0a)Program%20V8.xlsm)

* Communication Plan: (CR will not be required in the Unipro if any changes to communication plan)



* Change Management Plan: (CR will be required in the Unipro if any changes to change management plan)



* Configuration\_Management\_Plan: (CR will be required in the Unipro if any changes to configuration management plan)



* Management\_and\_Control\_Plan



* Release\_Management\_Plan



* Project Schedule:



* Intragroup Coordination: (CR will not be required in the Unipro if any changes to Intragroup Coordination)



* Templates:



* Estimation sheet:



## APPENDIX D: Product KPI and Acceptance Goals

|  |  |
| --- | --- |
| **Description** | **Goal** |
| RAM | Software shall not use more than 95 % of each type/zone of RAM memory |
| Software shall not use more than 95 % of ECU NVRAM memory. |  |
| Avaible:32KB CPU0,32K CPU1, 640K cluster RAM |  |
| ROM | Software shall not use more than 95 % of ECU ROM memory. |
| CPU Load | Shall be in the range of than (~70 to 75%).  \*Note: CPU load measurement KPI is provided by customer to measure CPU load after integration on EATON ASW, NPE ASW and KPIT BSW does not necessarily mean that KPIT needs to monitor the KPI alone. In case the parameter is above decided value then it is responsibility of EATON, Npe and KPIT to combinedly resolved and optimized |
| Tasks timing (100us, 1ms) Monitoring and expectation | 100us task should be less than 60us)  1ms task should be less than 700 us |
| ISR timings and Expectation | Execution time of task should be less than 70% of its periodicity |
| XCP and CAN Communication operation | Communication and Connection should be working |
| INCA Rasters Loading | Loading will be done as follows: |
| XCPChannel0 | 95%-100% |
| XCPChannel1 | 95%-100% |
| XCPChannel2 | 95%-100% |
| Polling 100ms | Polling values of variables shall work |
| Polling 500ms | Polling values of variables shall work |
| Polling 1sec | Polling values of variables shall work |
| Calibration Upload from ECU to INCA | Should be Working. |
| Calibration Change | Shall be able to change Calibration |
| UDS Services Check | Request response working. |
| Polling | Should be Working. |
| Bootloader flash and ULP test | Able to download via DiagAlyser tool |
| Check PWM’s on Control board | PWM’s visible on CRO |
| Check PWM’s with Gate Driver board Connected | PWM's visible on CRO |
| Bootloader | Should be flashed successfully. |
| ULP file flash (DiagAlyser) | Able to download via DiagAlyser tool |
| ULP File Flash Time (DiagAlyser) | Timing to be noted and shared in Test Report |
| Cal File Flash (DiagAlyser) | Able to download via DiagAlyser tool |
| Cal File Flash Time (DiagAlyser) | Timing to be noted and shared in Test Report |
|  |  |

## Appendix E : PMRP

## 9.6 APPENDIX F : Acceptance

## 