MathWorks
AUTOMOTIVE CONFERENCE 2023
Europe

How to achieve full coverage of configurable code through dynamic testing and static analysis with Polyspace

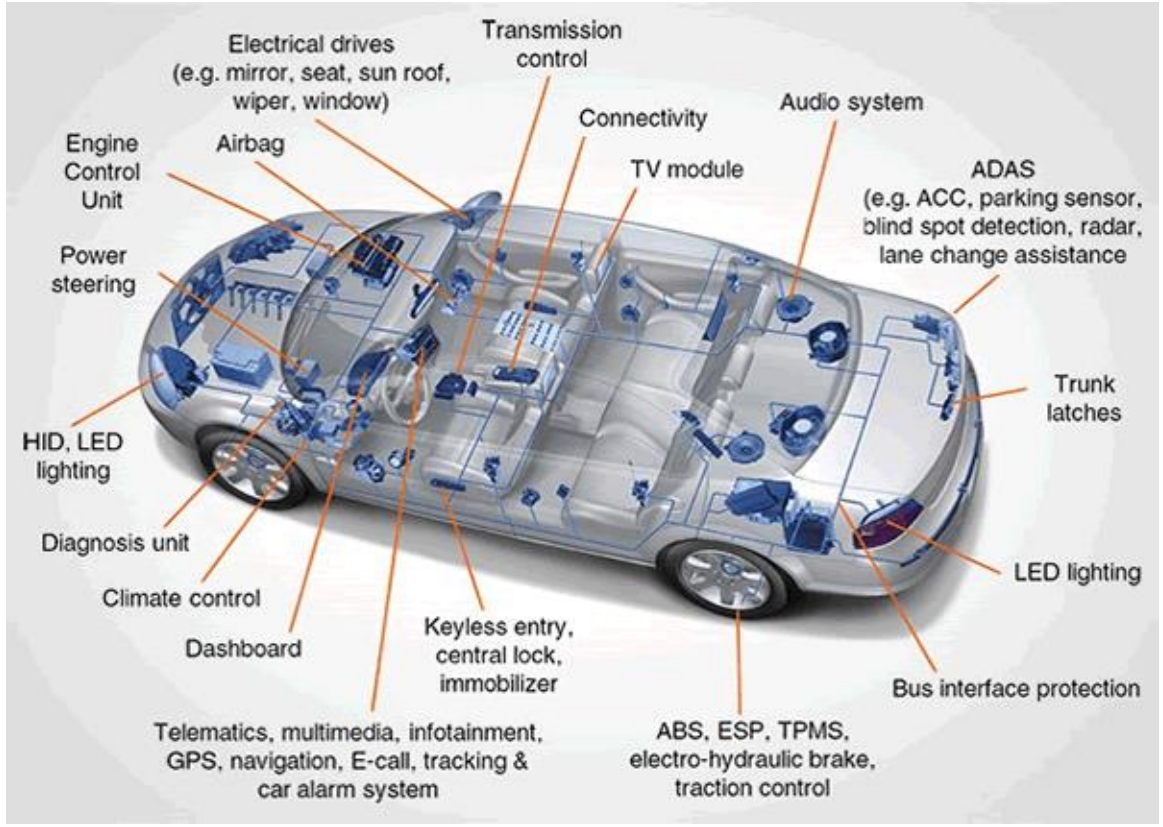
Cinzia Flavia TOMASELLO, STMicroelectronics

Agenda

- Automotive connectivity evolves
- Certification is key to deliver safe and robust Software-defined vehicles
- STMicroelectronics AUTOSAR software products
- STMicroelectronics methodology
- Verification and Validation process
- Leverage Polyspace to analyze Software variants

Automotive connectivity evolves

- Electronic systems are a fundamental part of any automotive environment
- Electronic control units (ECU) to control sophisticated engine functions:
  - Anti-lock braking systems, traction control systems, park assistance systems, etc..
- Internet of Things (IoT) exposes ECUs to hacker attacks:
  - We are connected to the vehicles using Cellular, Wi-Fi, Bluetooth, etc..
- Complex ECU system caused increase of embedded software and networking in vehicles



Certification is key to deliver safe and robust Software-defined vehicles

- ISO 26262 for functional safety
- ISO/SAE 21434 for cyber security
- Static Analysis evaluation is fundamental to ensure
  - Software quality
  - Compliance with regulations
  - Ensure that code follows coding standards as MISRA, Cert-C and CWE
- STMicroelectronics relies on Polyspace to fulfill ISO 26262 and ISO/SAE 21434 standard requirements

STMicroelectronics AUTOSAR software products: development challenges

- STMicroelectronics develops AUTOSAR MCAL drivers for Automotive customers.
- Use of AUTOSAR system architecture reduce development cost and avoid re-development of SW for similar application
- Drivers are developed for PowerPC and ARM architecture and are configurable for each customer
- High level of driver configuration leads to a **huge number of software variants**

Driver	Number of Boolean preprocessor macro (#define)	Number of SW variants considering only Boolean parameters
MCU	58	$2^{58} = 288*10^{15}$
CAN	50	$2^{50} = 112*10^{13}$

- Fulfill Automotive **safety and security standards**

Methodology to verify MCAL driver variants

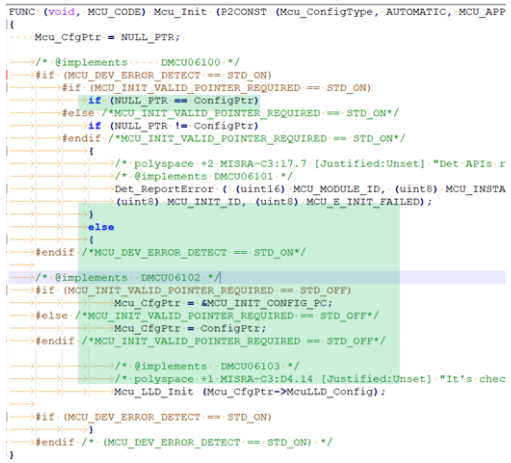
- The AUTOSAR software team defined a methodology to select the minimal subset of variants to obtain the full structural coverage of the configurable code
- Variant selection
  - Human and iterative process based on developer expertise
- Measure completeness of variants subset
  - Metrics: Statement Coverage, Decision Coverage, MC/DC
  - Aggregate code coverage: an internal developed tool aggregates the coverage results for each software variant generating a report containing the aggregated coverage score and map.

Aggregated code coverage

- Aggregate results ensure goodness of variant set

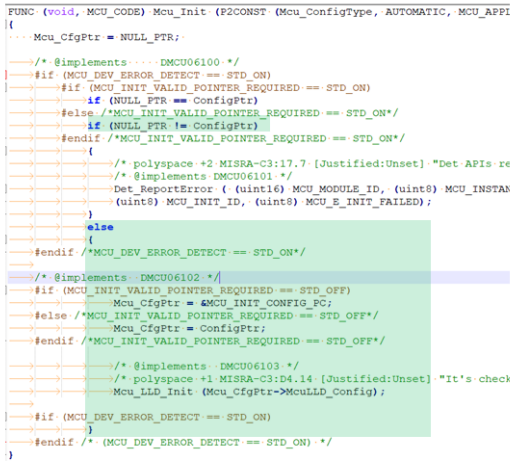
Variant
MCU_DEV_ERROR_DETECT=STD_ON
MCU_INIT_VALID_POINTER_REQUIRED= STD_ON

Reachable lines = 10 - Tested Line = 7
Statement coverage score = 70%



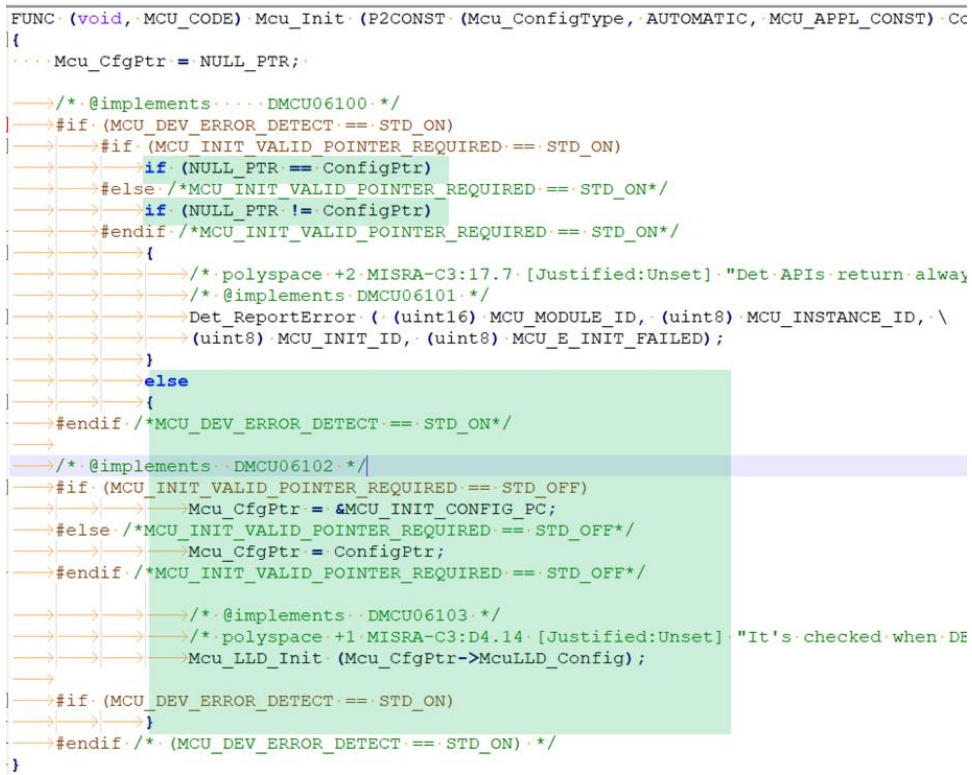
Variant
MCU_DEV_ERROR_DETECT=STD_ON
MCU_INIT_VALID_POINTER_REQUIRED= STD_OFF

Reachable lines = 10 - Tested Line = 7
Statement coverage score = 70%



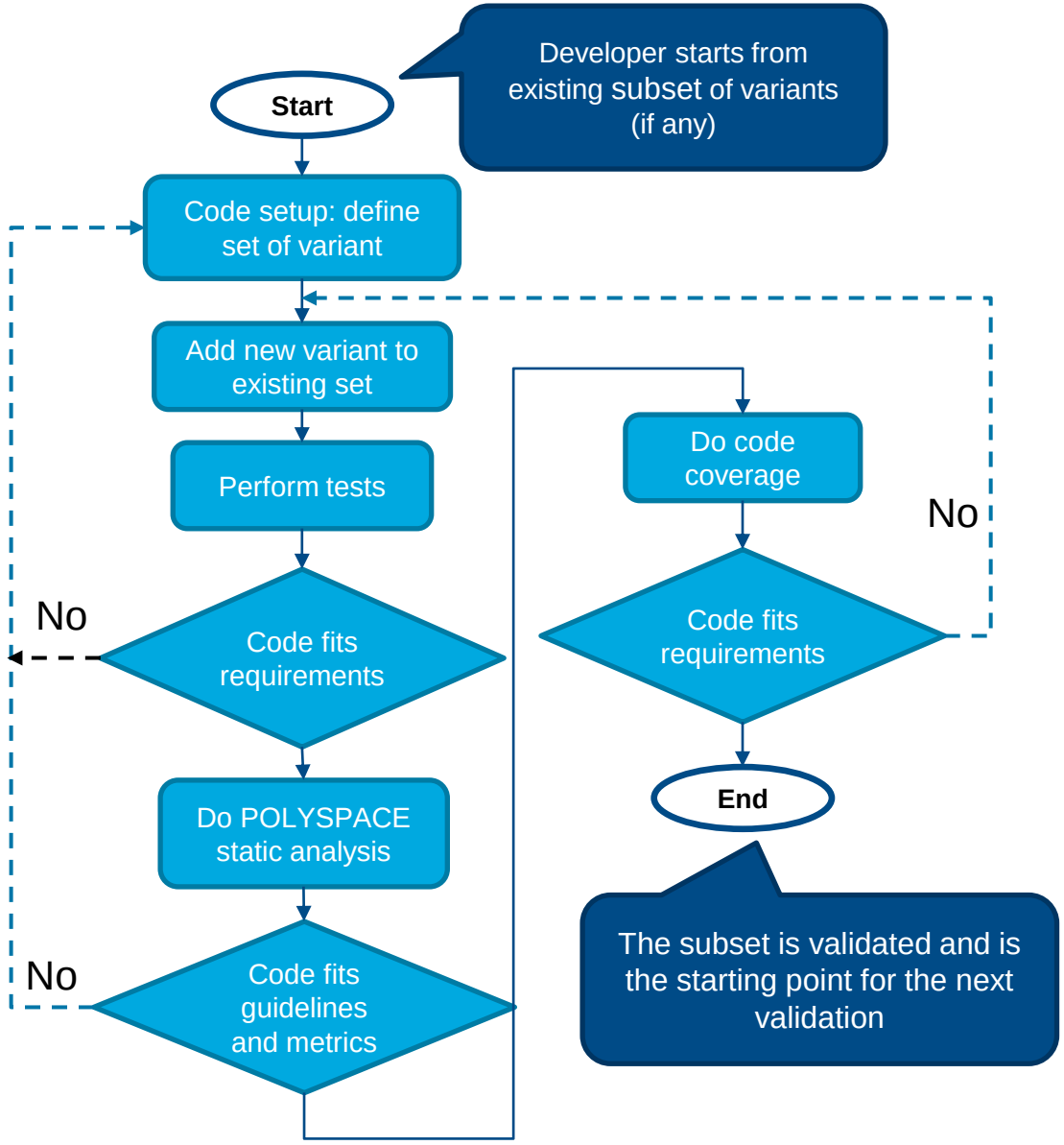
Aggregate

Reachable lines = 11 - Tested Line = 8
Statement coverage score = 72%



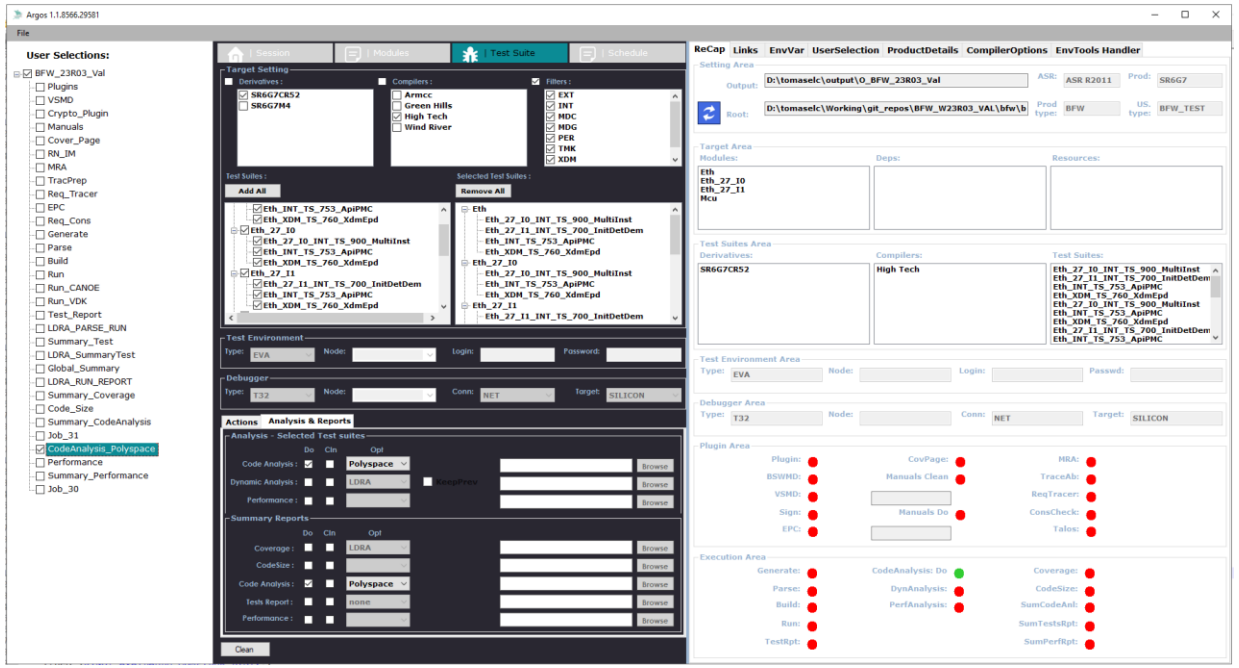
Verification and Validation process

- Each SW variant is verified
  - Functional test on target
  - Code Coverage (Statements, Branch Coverage, MC/DC Coverage)
  - HIS metrics (e.g. Cyclomatic Complexity, Comment Density – using Polyspace Bug Finder)
  - Static Analysis (MISRA, Cert-C, CWE – using Polyspace Bug Finder)
  - Custom naming convention (using Polyspace Bug Finder)
- Consolidate results for each category



Leverage Polyspace to analyze SW variants

- Framework (BFW) to select single variant and run VnV
  - User can select architecture/compiler/target and run one of the test mentioned in the VnV process
  - The BFW collect all user selections to invoke the testing tool
- STMicroelectronics relies on Polyspace for static code analysis
  - SW variants compliant with ISO 26262
  - Fully configurable using script
  - Supports all architectures and compilers



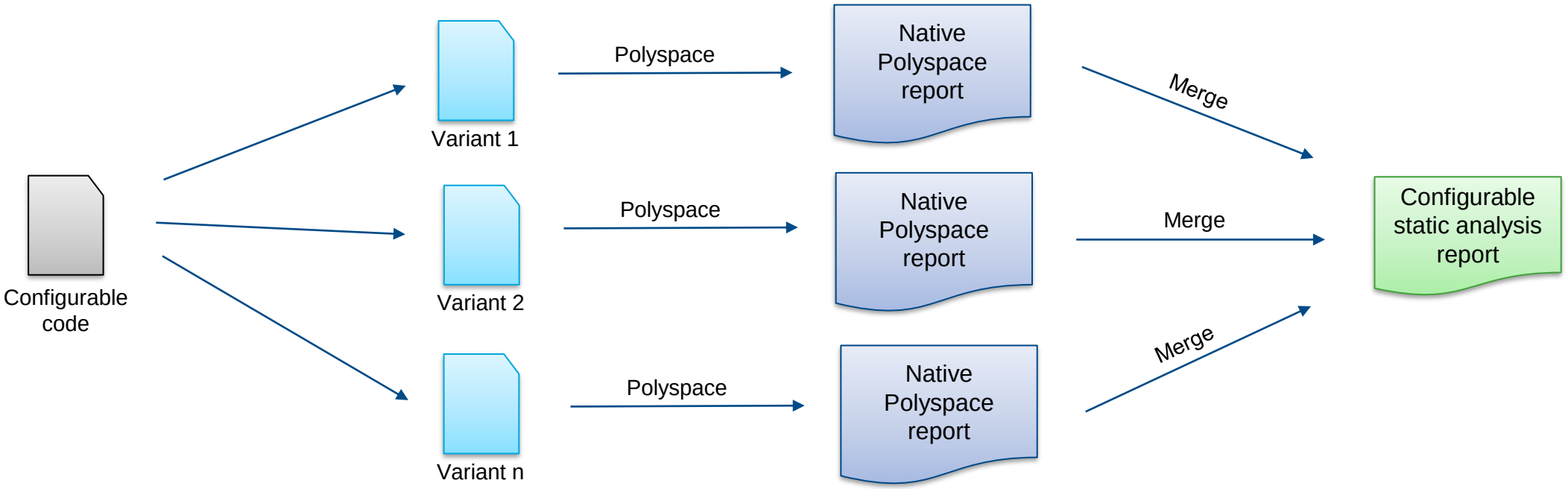
Managing Polyspace findings

- The AUTOSAR software quality team wrote a document[(1)] that defines which static analysis metrics and which thresholds shall be applied to the AUTOSAR software
- Polyspace gives a severity level for defect
  - Defect with HIGH impact must be fixed in configurable code
- MISRA C:2012
  - Mandatory: the standard does not permit deviation from these rules. The violation must be solved in configurable code
  - Required/Advisory: the standard permits only the deviations that are recorded and authorized. The violation is derogated directly in code using a specific notation

(1) Static_Analysis_Metrics_&_Coding_Guidelines

Polyspace results of configurable software

- Polyspace can generate analysis report for each variant analyzed
- An aggregated report is generated from single variant analysis to measure code quality of the complete configurable software



Results

- Applying this methodology, the number of variants to be verified to ensure completeness of the code analysis and the structural coverage decrease drastically

Driver	Number of Boolean preprocessor macro (#define)	Number of SW variants considering only Boolean parameters	Subset of variants	Coverage score	Compliance with Static Analysis Metric Guidelines
MCU	58	$2^{58} = 288*10^{15}$	177	100%	100%
CAN	50	$2^{50} = 112*10^{13}$	179	100%	100%

- Structural coverage is verified
- Thanks to Polyspace, errors in the software are found earlier, before the delivery to customer, and quality standards are fully met

Take away

- Significant improvement of
  - productivity of ST development team
  - the quality of the configurable software
- Reusable framework beyond firmware development
  - Other teams (e.g. Safety Library team) are adopting this framework
- Possible to extend this method to other software verification activities

Next steps

- Automatic extraction of the smallest software variants
- Improve code metrics consolidation across variants
- Extend code verification to formal code verification (e.g. Polyspace Code Prover)

Thank you!
Any questions?