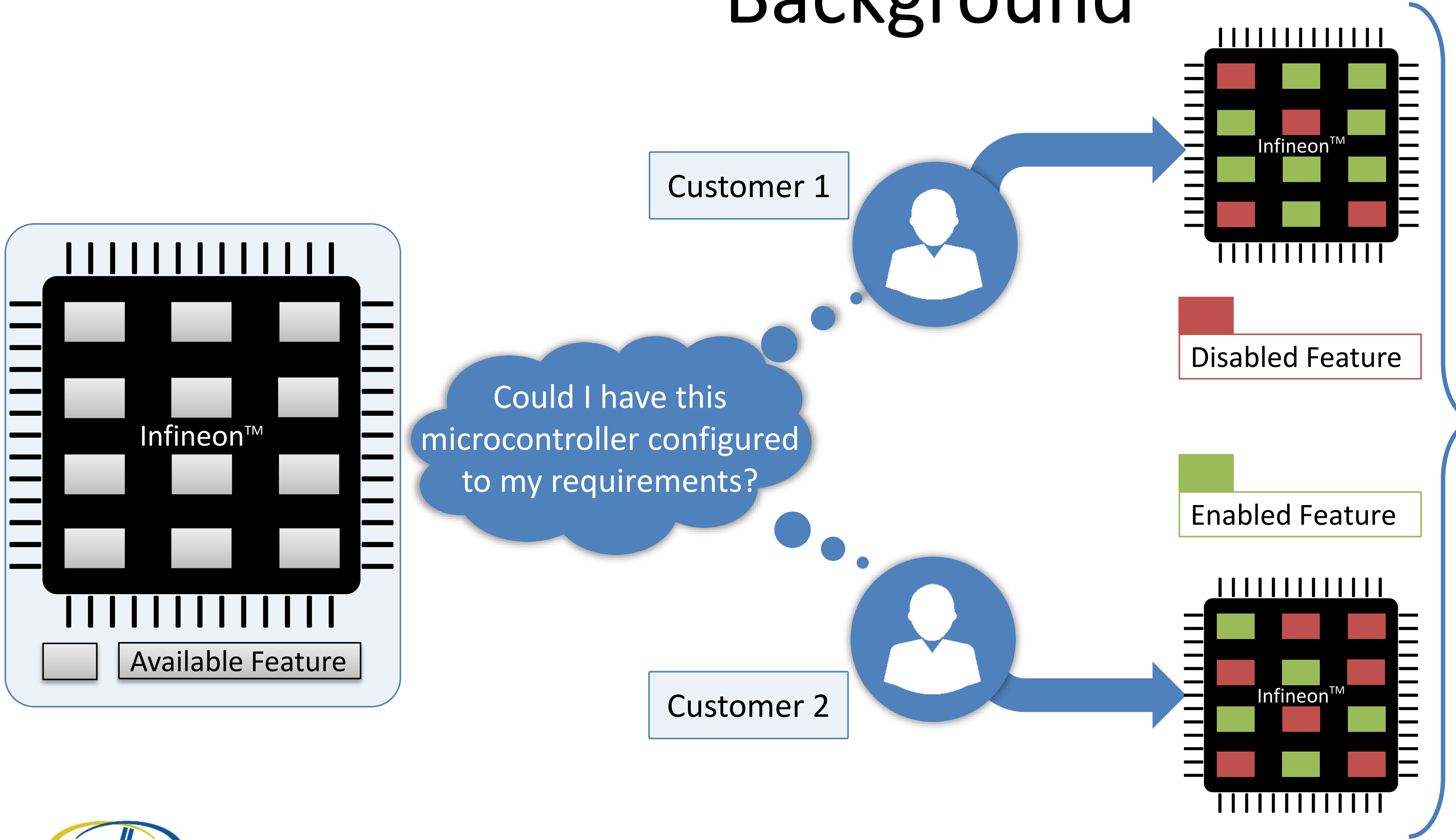
Automatic Generation of Infineon Microcontroller Product Configurations

Prateek Chandra, Leily Zafari, Boyko Traykov

Infineon Technologies

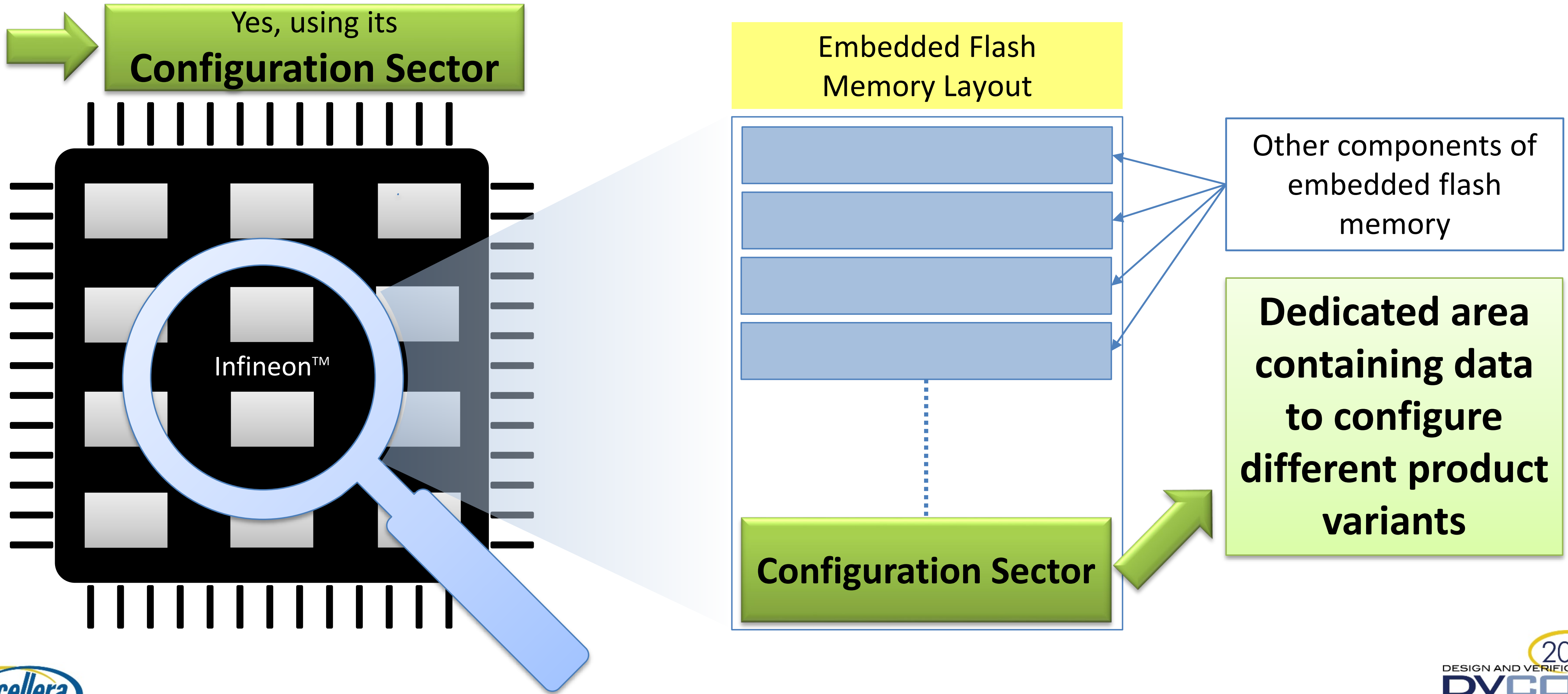
Agenda

Background



There is a need for **easily configurable microcontrollers** to meet customer's requirements

Configurability of a microcontroller


Yes, using its
Configuration Sector
Infineon™
Embedded Flash Memory Layout
Other components of embedded flash memory
Configuration Sector
Dedicated area containing data to configure different product variants

How a configured variant is produced?

Step 3

Create the required output file(s) for each variant

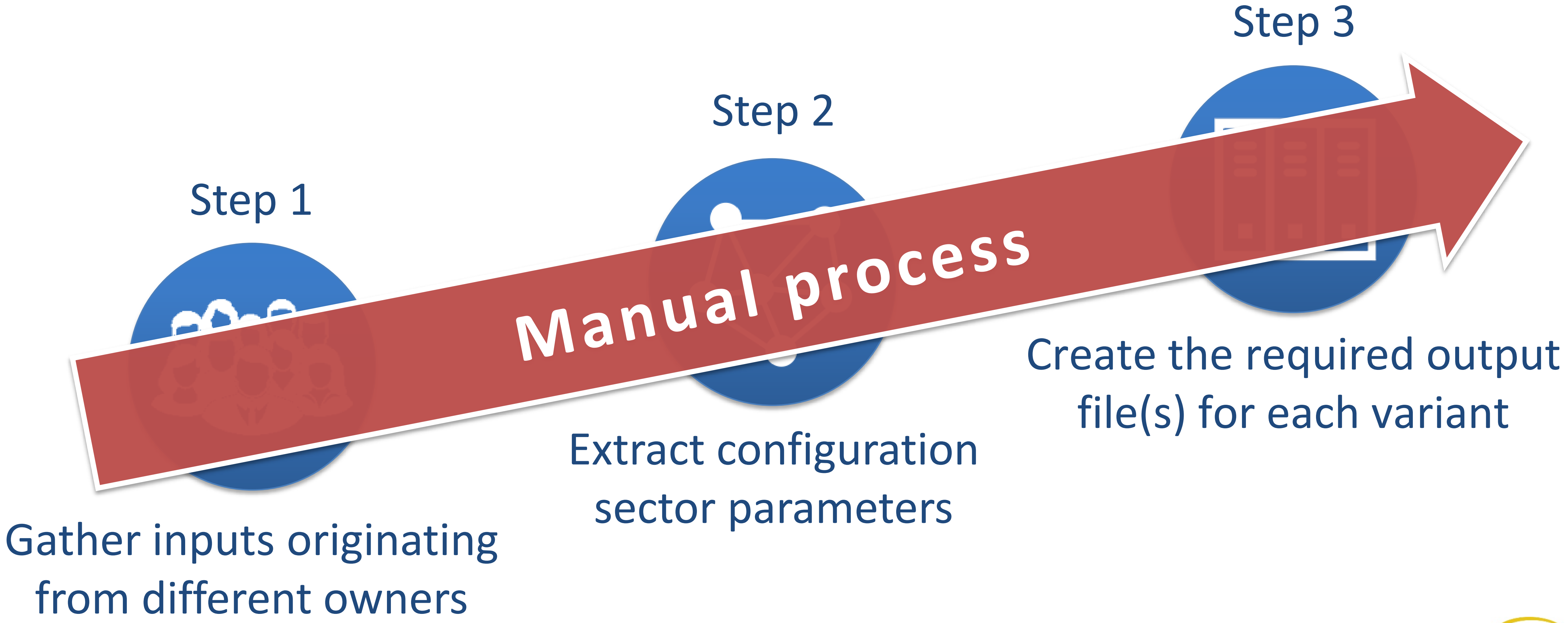
Step 2

Extract configuration sector parameters

Step 1

Gather design data from respective owners

Producing a configured variant


Step 1
Step 2
Step 3
Manual process
Gather inputs originating from different owners
Extract configuration sector parameters
Create the required output file(s) for each variant

Is the manual approach feasible?

NO

Gathering Input Data

Different owners responsible for contributing to the input

Data Traceability

Difficult to trace the origin of data from the heterogeneous inputs

Verification Gaps

Unreliable/incomplete verification and test results

Consistency

Missing or erroneous configured features; defective devices delivered

Problems with manual approach in detail

Step 1

Gather design data from respective owners

Input owners based at multiple sites

Incoherent communication

Effort is huge for complex designs

Problems with manual approach in detail

Step 2

Extract configuration sector parameters

Keeping track of the origin of parameters

No version control of the design files

May end up with undesired configuration

Problems with manual approach in detail

Step 3

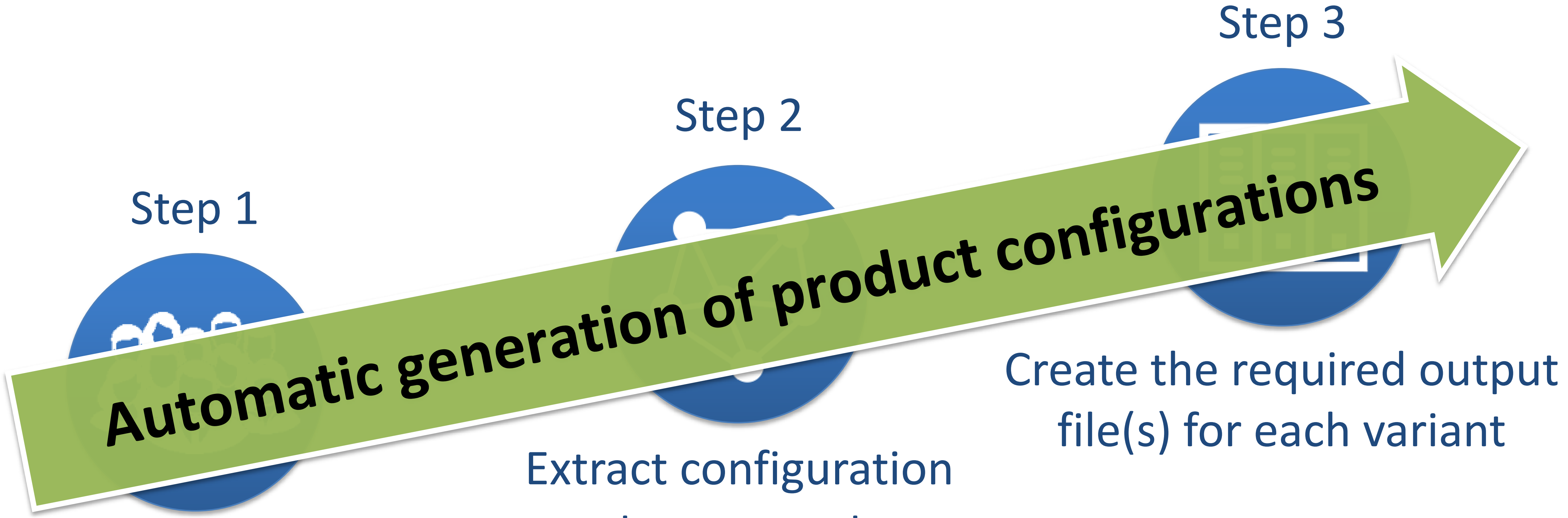
Create the required output file(s) for each variant

Is zero-defect delivery ensured?

Is the input and output data consistent?

What is the Quality of Delivarables?

Proposed Solution

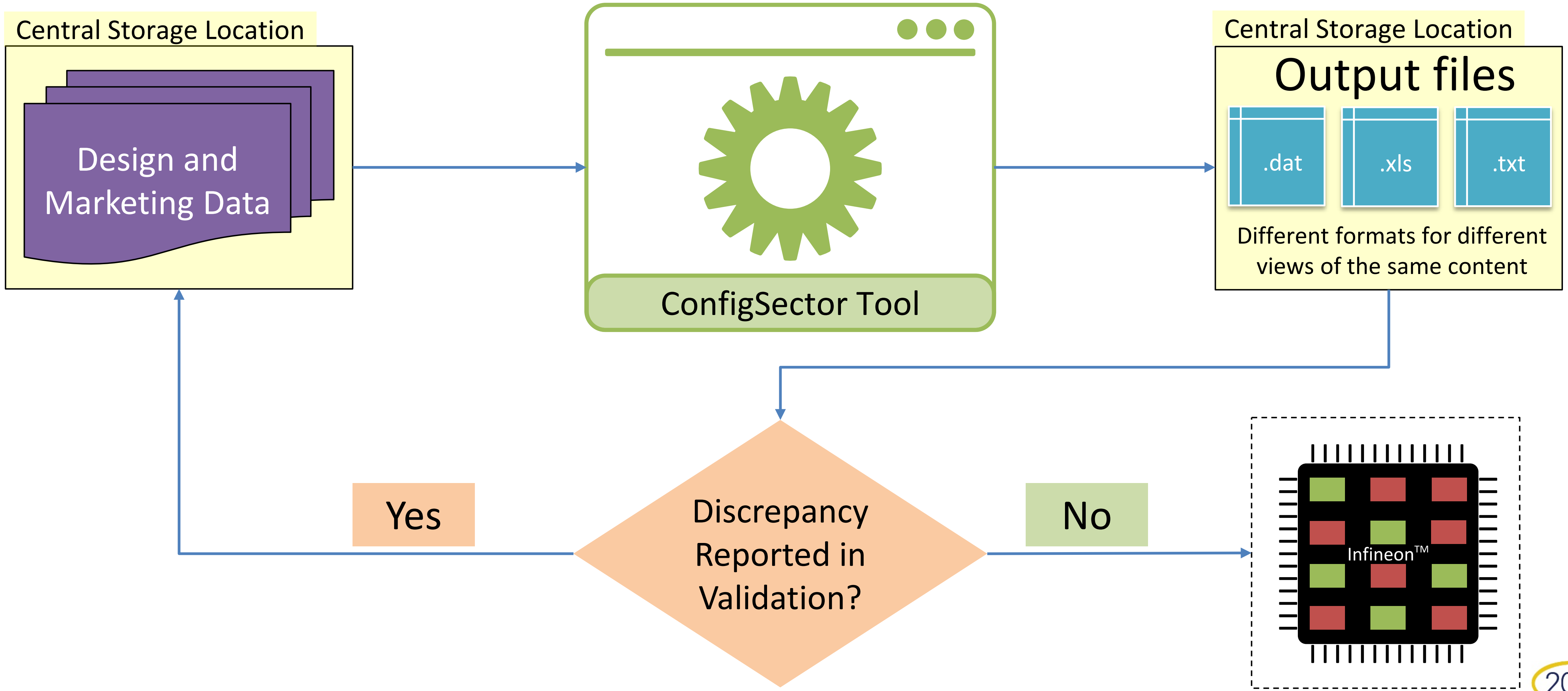


Gather design data from respective owners

Extract configuration sector parameters

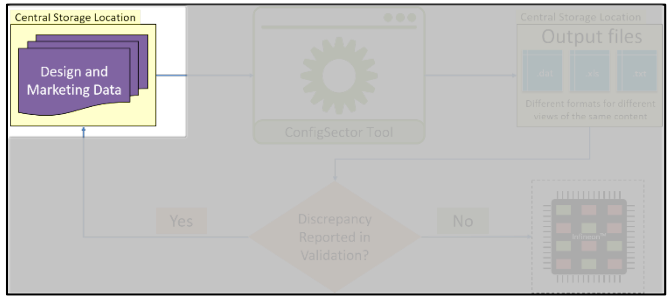
Create the required output file(s) for each variant

Workflow

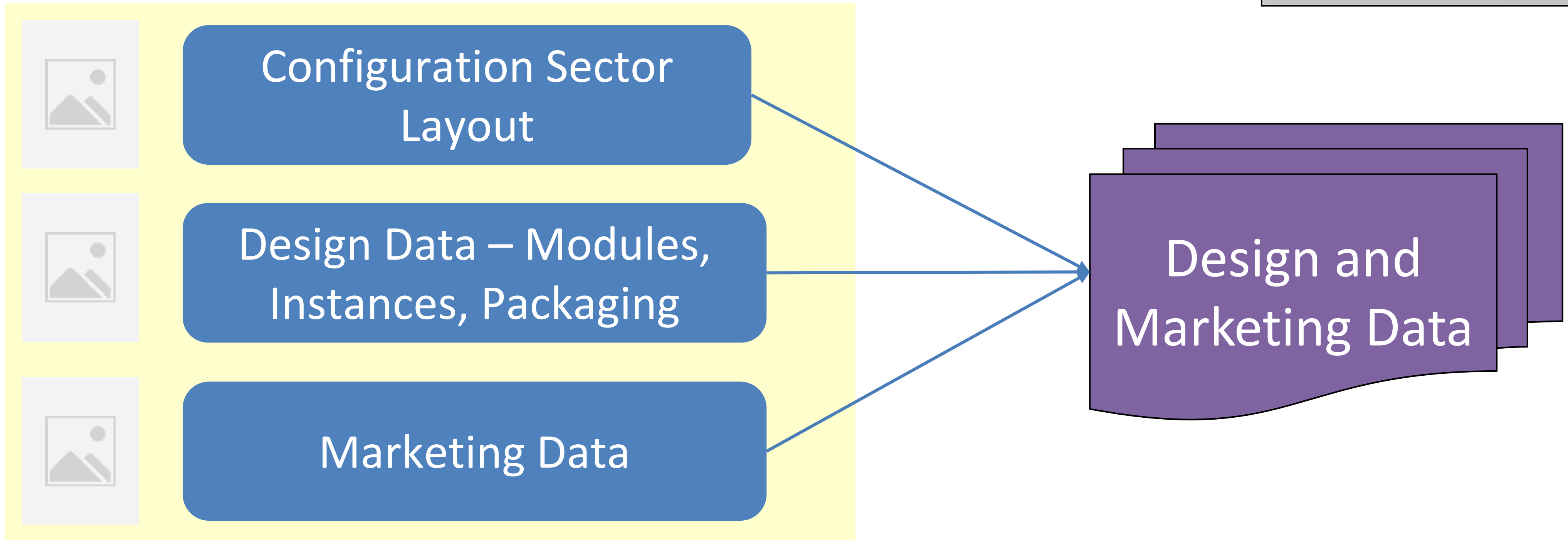

Central Storage Location
Design and Marketing Data
ConfigSector Tool
Central Storage Location
Output files
.dat
.xls
.txt
Different formats for different views of the same content
Discrepancy Reported in Validation?
Yes
No
Infineon™

Features

Input files present in central version controlled baseline system



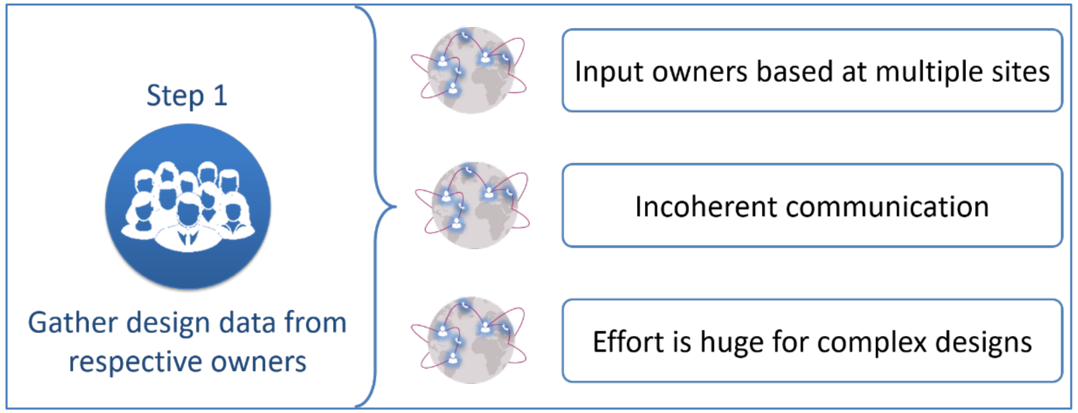
Central Storage Location
Design and Marketing Data
ConfigSector Tool
Central Storage Location
Output files
Different formats for different views of the same content
Yes
Discrepancy Reported in Validation?
No



Configuration Sector Layout
Design Data – Modules, Instances, Packaging
Marketing Data
Design and Marketing Data

Features

Input files present in central version controlled baseline system


Step 1
Gather design data from respective owners
Input owners based at multiple sites
Incoherent communication
Effort is huge for complex designs

No dependency on data owners

✓ Every input file is version controlled

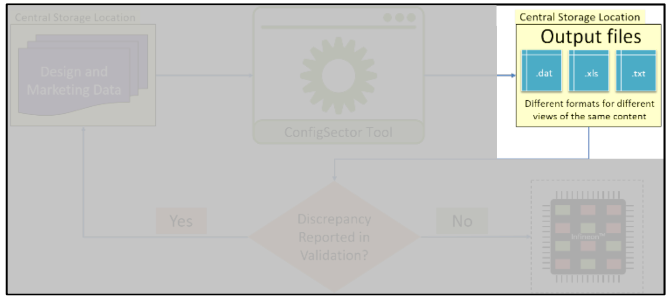
✓ Tool automatically searches for relevant input files

Features

Multiple views of the same output

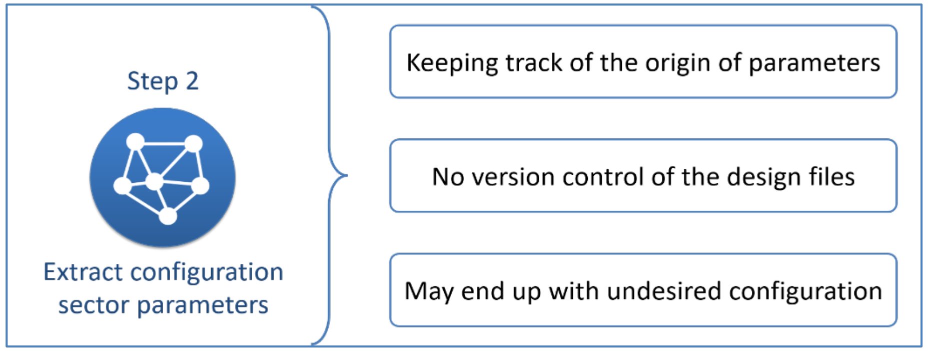
Output files

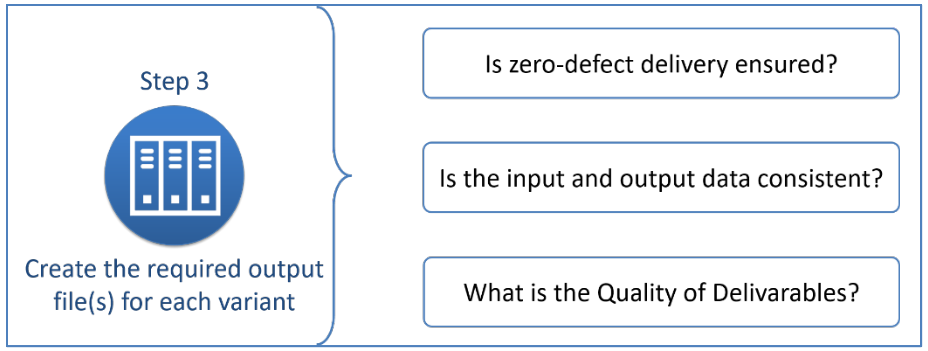
".dat" Files	For loading in the Configuration Sector
".txt" File	For easy comparison for the testers
Excel Sheets	For easy overview
XML File	For the tool developers


Central Storage Location
Design and Marketing Data
ConfigSector Tool
Central Storage Location
Output files
.dat
.xls
.txt
Different formats for different views of the same content
Yes
Discrepancy Reported in Validation?
No

Features

Output files released in central version controlled baseline system


Step 2
Extract configuration sector parameters
Keeping track of the origin of parameters
No version control of the design files
May end up with undesired configuration


Step 3
Create the required output file(s) for each variant
Is zero-defect delivery ensured?
Is the input and output data consistent?
What is the Quality of Delivarables?

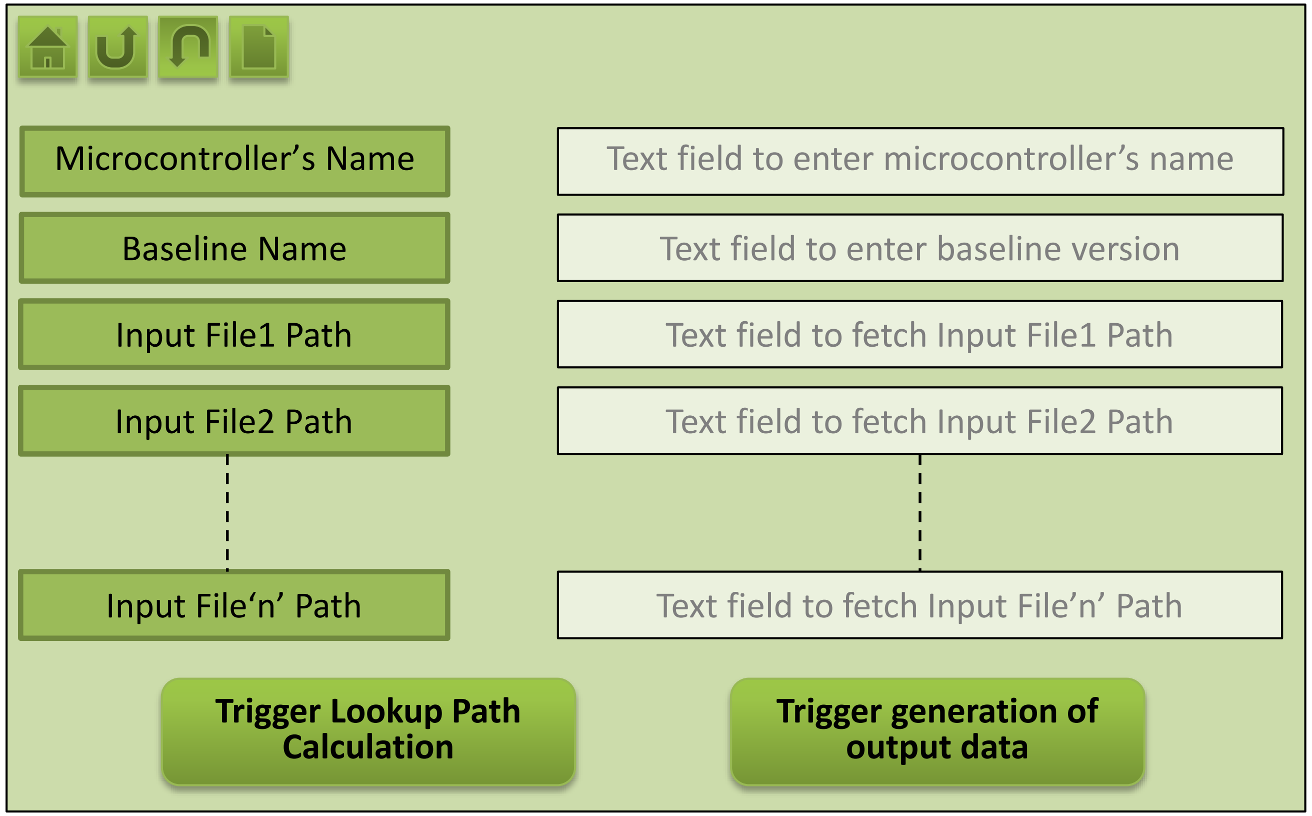
✓ Output XML file stores the origin

✓ Output ".txt" file for testers to catch undesired configuration

✓ Consistent output;
Zero defect delivery

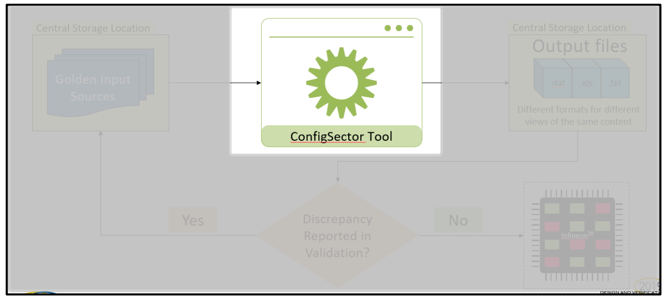
Features

Platform independent GUI



Microcontroller’s Name	Text field to enter microcontroller’s name
Baseline Name	Text field to enter baseline version
Input File1 Path	Text field to fetch Input File1 Path
Input File2 Path	Text field to fetch Input File2 Path
Input File‘n’ Path	Text field to fetch Input File’n’ Path

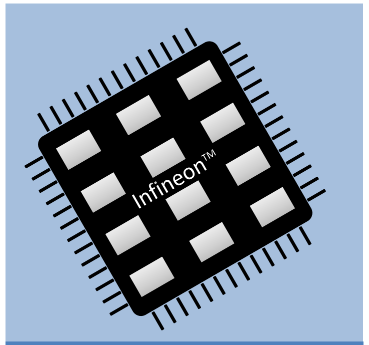
Trigger Lookup Path Calculation
Trigger generation of output data



✓ Single click generation

✓ Hard stop if any input is missing

Features



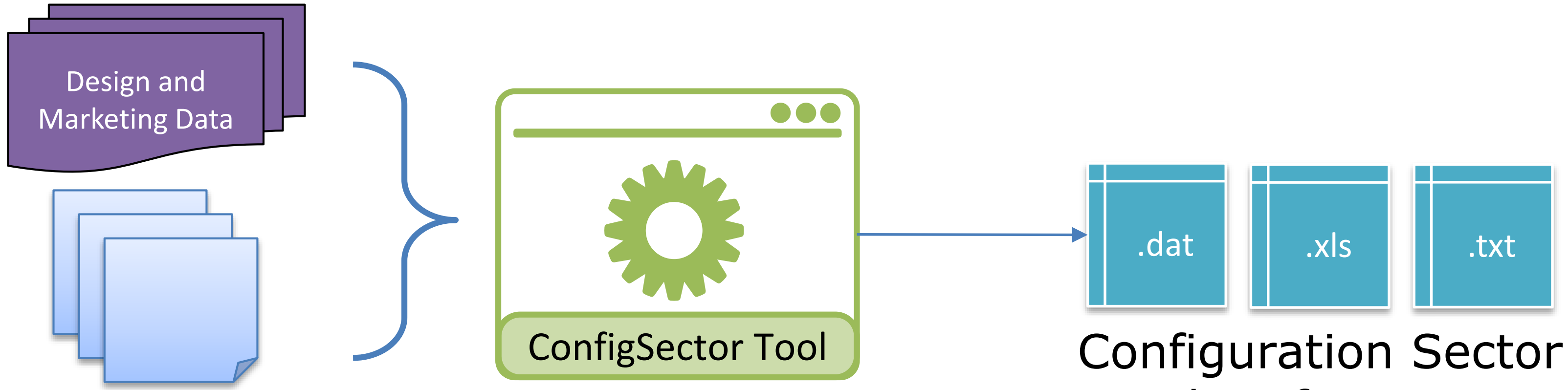
Taped out

After some time…

New configuration needed

But no design data can be modified !!!

Config Sheet overwrites CFS values from design sources



Config sheet for new configurations

Configuration Sector data for new configuration

Results

With manual approach		With tool usage
1 week	Generation Time	1 hour
2 days	Change Requests	1 hour
Manual calculation based on some rules; repetitive and time consuming	ChipID Calculation	Automatic unique ChipID calculation; easy tracking of configured variants

Benefits

Configuration Sector Data for all the variants for a microcontroller in a single run

Possible to have product variants even after design tapeout

Reduced time to market; thus reduction in costs incurred; ease of maintenance

Conclusion

“Correct-by-construction” methodology makes the flow robust

CFS flow boosts the productivity by cutting down on manual labor

Zero defect deliveries instils more trust in the customers

Questions