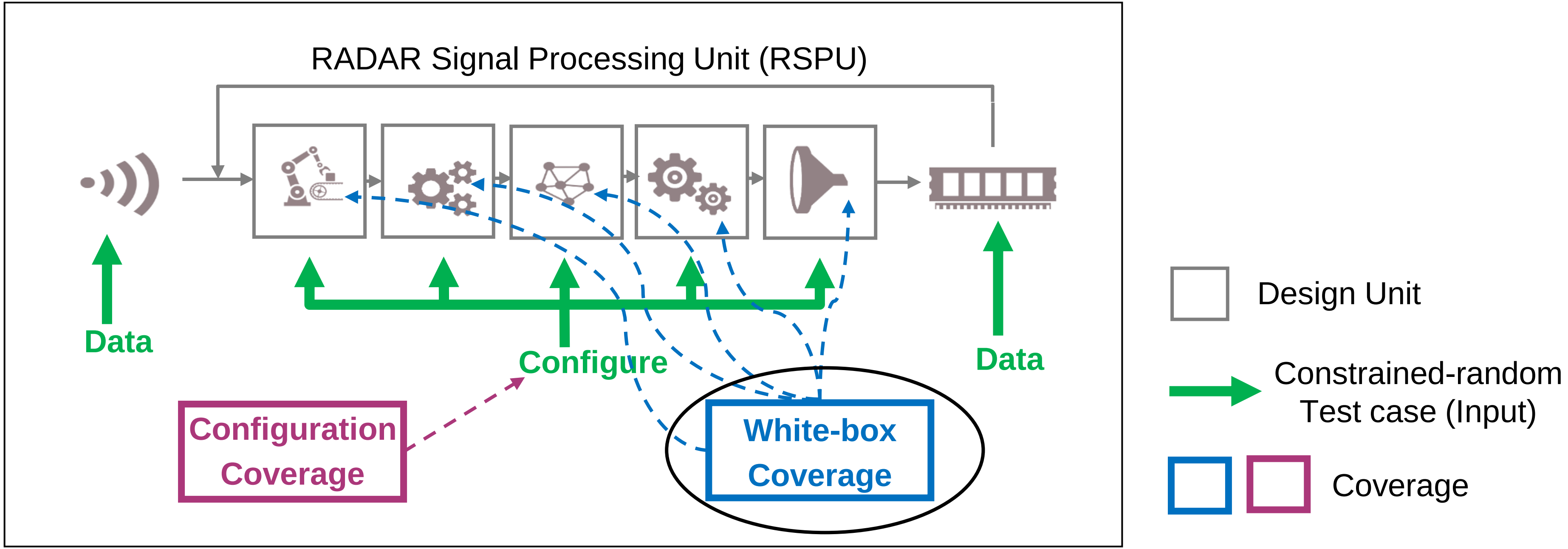
Novelty-Driven Verification: Using Machine Learning to Identify Novel Stimuli and Close Coverage

DVCon U.S. 2021

Tim Blackmore, Rhys Hodson *Infineon Technologies*
Sebastian Schaal *Luminovo GmbH*

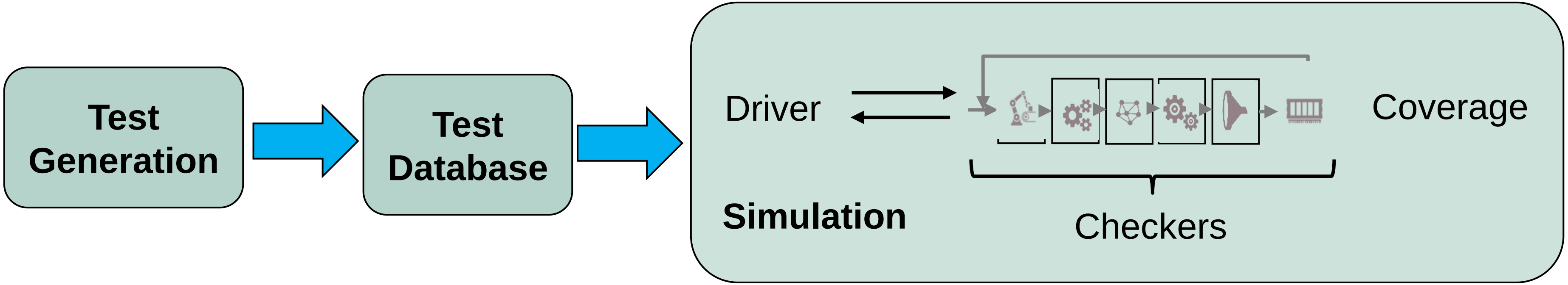
Aim and Context

› Use ML to *close coverage* more quickly when we have *no (obvious) deterministic relationship* between a test and the coverage we wish to hit

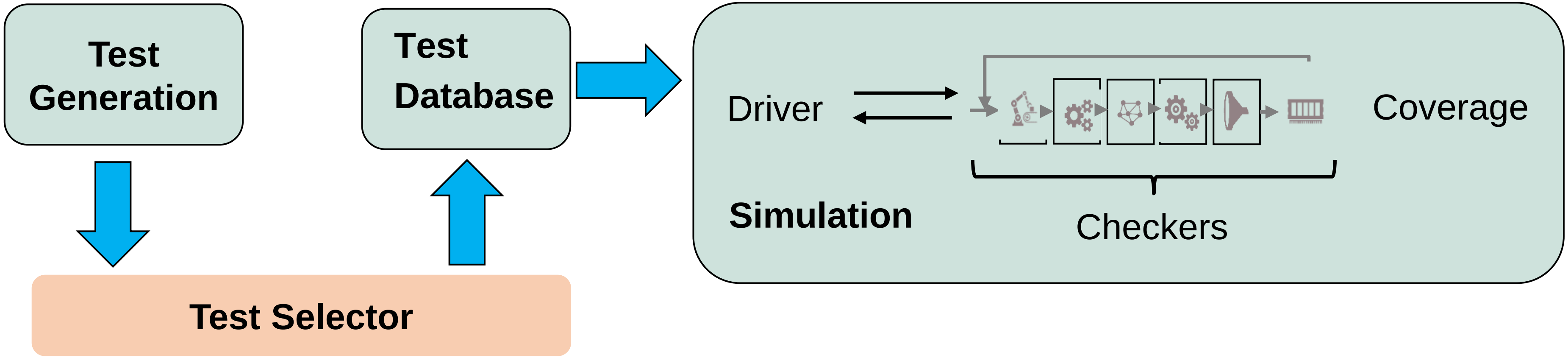


The Real-World Test Bench

- De-coupled Generation and Simulation Environments
- Introduced to avoid random instability
- Many other benefits ...

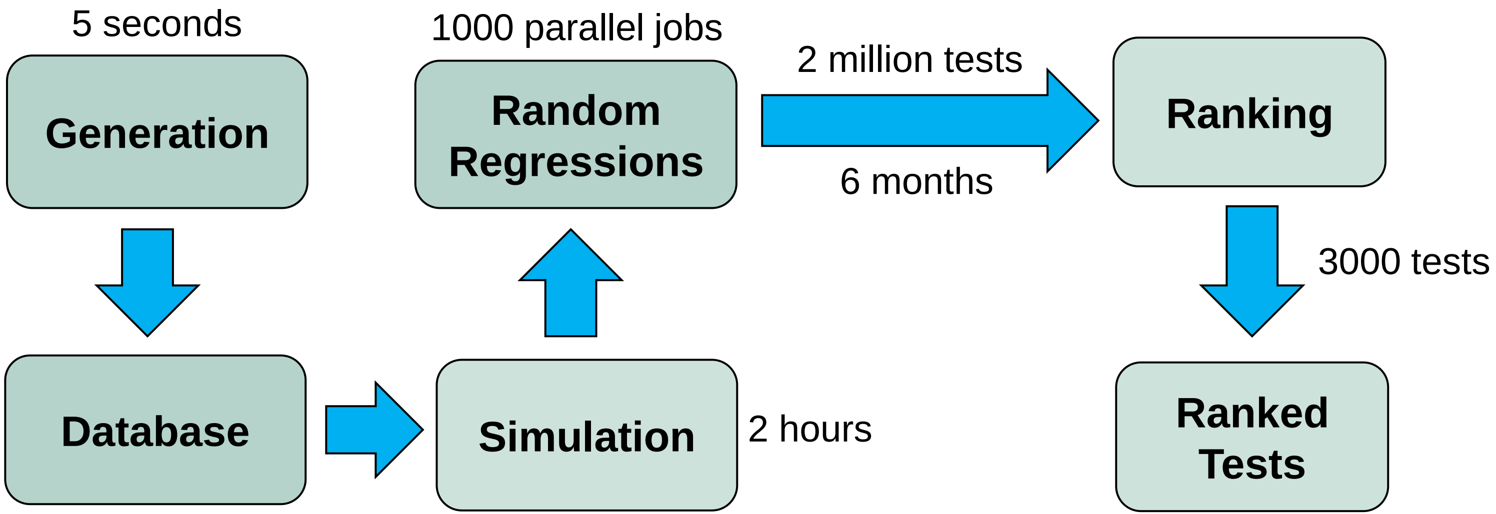


Test Selector



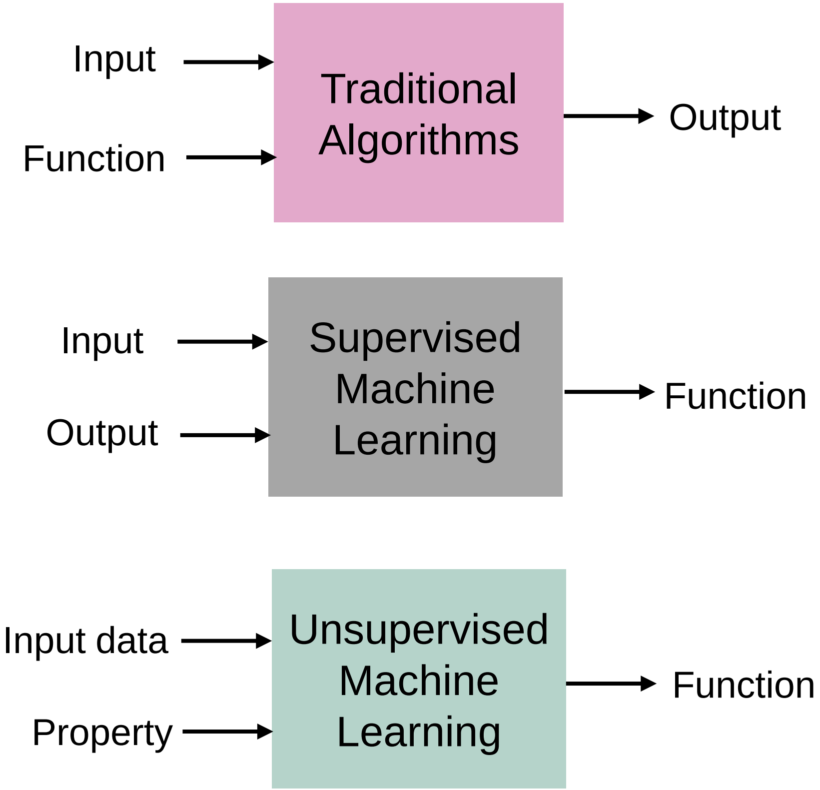
- Idea is to use ML techniques to select tests that are most likely to add to coverage
- Why bother?
- Why use ML?

Why bother? - Potential Benefit



- Focus is on reducing the 2 million simulations
- Since simulation time dominates generation time saving will be realised in schedule saving

Why ML for test selection?

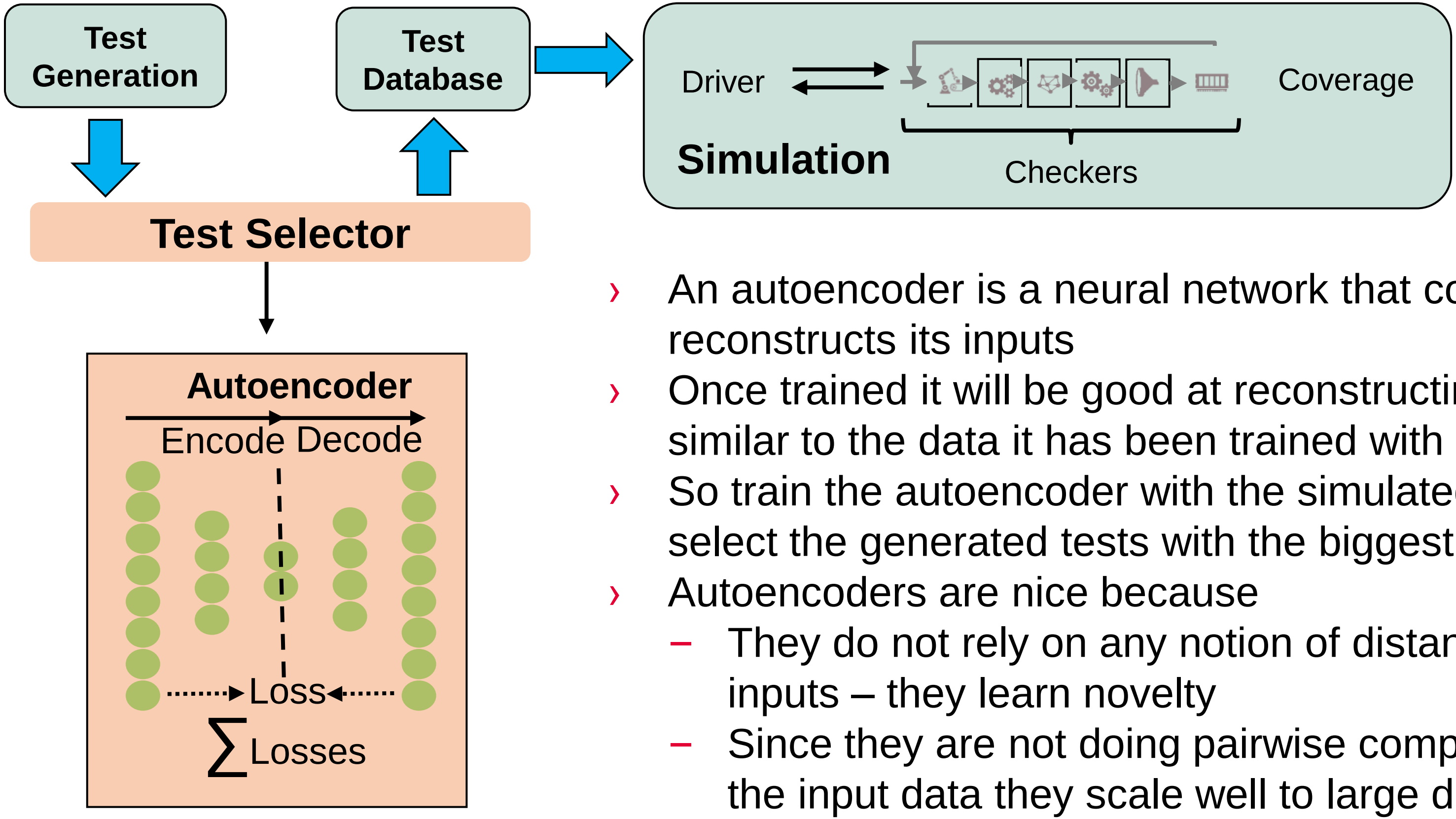


Not suitable – not easy to describe rules that relate tests to the white-box functional coverage they hit

Not suitable – cannot learn how to hit coverage that has not been hit (no postive samples to learn from)

Properties include detecting anamolies in data. Hmmm – **maybe suitable**. Perhaps anomalous tests are more likely to hit new coverage?

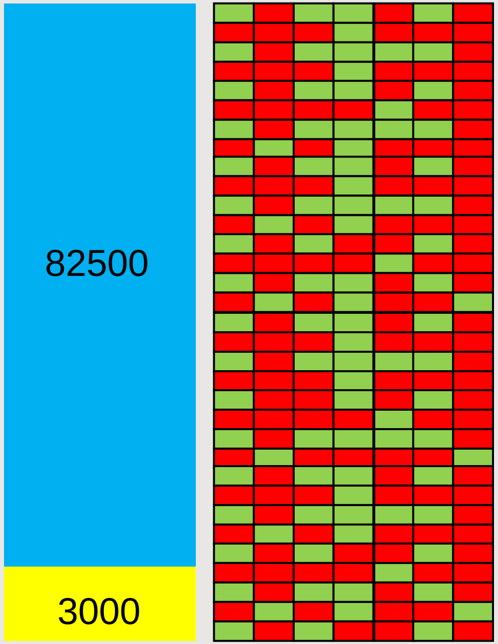
Autoencoders as Anomaly/Novelty Detectors



- An autoencoder is a neural network that compresses and reconstructs its inputs
- Once trained it will be good at reconstructing data that is similar to the data it has been trained with
- So train the autoencoder with the simulated tests and select the generated tests with the biggest loss
- Autoencoders are nice because
  - They do not rely on any notion of distance between inputs – they learn novelty
  - Since they are not doing pairwise comparisons between the input data they scale well to large data sets

Lets give it a go …

Test-Coverage Database



Simulated randomly generated tests for 1 week using 1000 licenses

- About 82500 tests
- Coverage on white-box functional coverage collected and put in database
- 3000 ranked tests added to give 100% coverage

An ordering of tests was determined by novelty according to autoencoder

- Autoencoder was trained after each 1000 tests
- Full re-training takes approx. 1 hour on single CPU

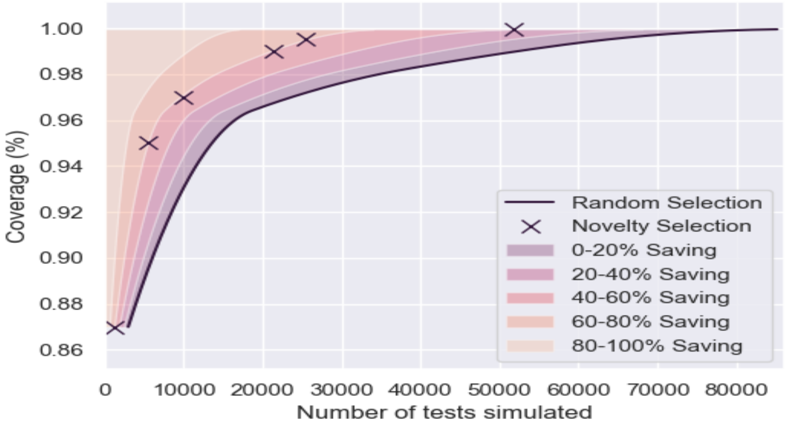
Coverage was collected from database as though tests were simulated in order selected to see how quickly coverage was closed

A second ordering was determined by random selection and coverage collected

- In fact this was done 10 times and ordering that closed coverage fastest selected

Experimental Results

Coverage	Number of Tests with Random Selection	Number of Tests with Novelty Selection	Saving in Number of Tests Simulated	% Saving in Number of Tests Simulated
87%	3000	1200	-1800	-60%
95%	13600	5400	-8200	-60%
97%	23700	9950	-13750	-58%
99%	52350	21300	-31050	-59%
99.5%	63500	25400	-38100	-60%
99.95%	85150	51800	-33350	-40%



Novelty selection achieved 99.5% coverage with 60% fewer tests than random selection

› Only 10’s of white-box coverage items left to hit

Conclusion

If experimental results replicated in real project then use of a test selector based on novelty detection

- Would save over one million simulations
- Enable coverage closure over 3 months earlier

We are now using a test selector based in novelty detection in the verification of the RSPU and looking to use it on other projects

Open Questions

Can novelty be regarded as a reasonable proxy for coverage?

- Open up the possibility of using other machine learning techniques to generate (rather than select) novel tests

Does novelty help beyond coverage closure?

- Explore more design behavior?
- Find more bugs more quickly?

Can novelty help address limitations with coverage-driven verification?

- Should novel tests hit coverage?
- Does this give an insight into incomplete and buggy coverage models?

Questions

Q: Is this already available in EDA tools?
A: No - I don't think so

- In particular, some related but different ML applications in EDA tools (from the last DVClub)
  - Cadence's Xcelium ML is aimed at reducing simulation cycles to hit same coverage with a randomized test suite
    - We're trying to hit *new* coverage
  - Breker use ML to optimize test suites for graph-based verification
    - We have no known relationship between inputs and coverage

Questions

Q: Is the approach original?

A: Not fully, …

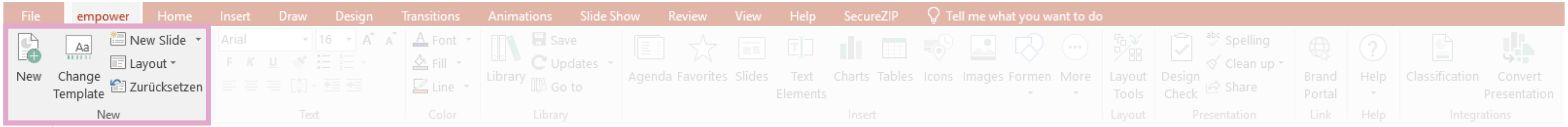
- General principle is well described in ‘Data Mining In EDA - Basic Principles, Promises, and Constraints’ (Wang, Abadir) DAC 2014
- References ‘Functional test selection based on unsupervised support vector analysis’ (Guzey, Wang, Levitt, Foster) DAC 2008

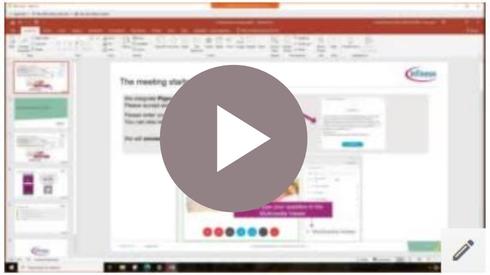
A: but, …

- The ML techniques we use are original and have some advantages
- Previous publications used only very small examples
- We demonstrate the approach works well on an example approaching real-world complexity
  - Uses real-world design and test bench
  - Actual (white-box) functional coverage model
- ML provides real benefit

Part of your life. Part of tomorrow.

empower® is the successor of Infineon Powerpoint Tool





What is empower?

Empower offers all the existing functions of Infineon Powerpoint Tool and in addition improved features and an enriched toolset that make working with PowerPoint easier. You will be able to save time on formatting while creating professional presentations with Infineon corporate design.

Empower is available in Infineon Apps store for installation

[How to install empower?]

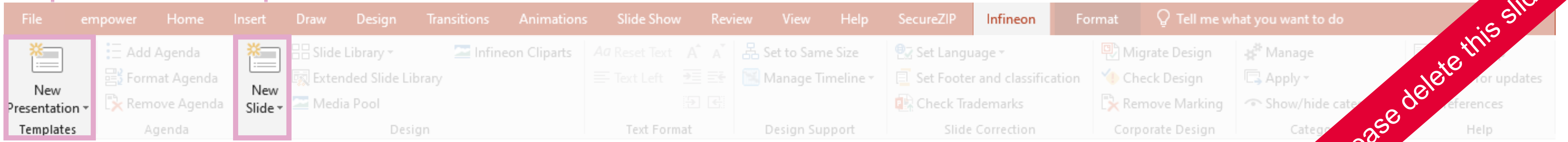
[Intranet Site]

[End user FAQ]

[Community]

[Videos]

[Contact]



Please delete this slide!