An Assembler Driven Verification Methodology (ADVM)

John S MacBeth and Ken Gray, *Verilab GmbH*

Dietmar Heinz, *Infineon Technologies AG*

Presentation Overview

- Typical assembler test issues
- Our solution = ADVM
- Summary

Typical Assembler Test Issues

- **Hardwired values within the code**
- **Derivative specific information embedded into the test code**
- **Lack of comments or documentation that describes the purpose of the test, its relation to the test-plan and its dependencies**
- **Code repetition where functions could/should be used**
- **No mechanisms for encapsulating best practices for testing or performing common tasks**
- **In summary:**
  - **The core problem is a lack of test PORTABILITY**

The Principle Goal of the ADVM = Test Portability

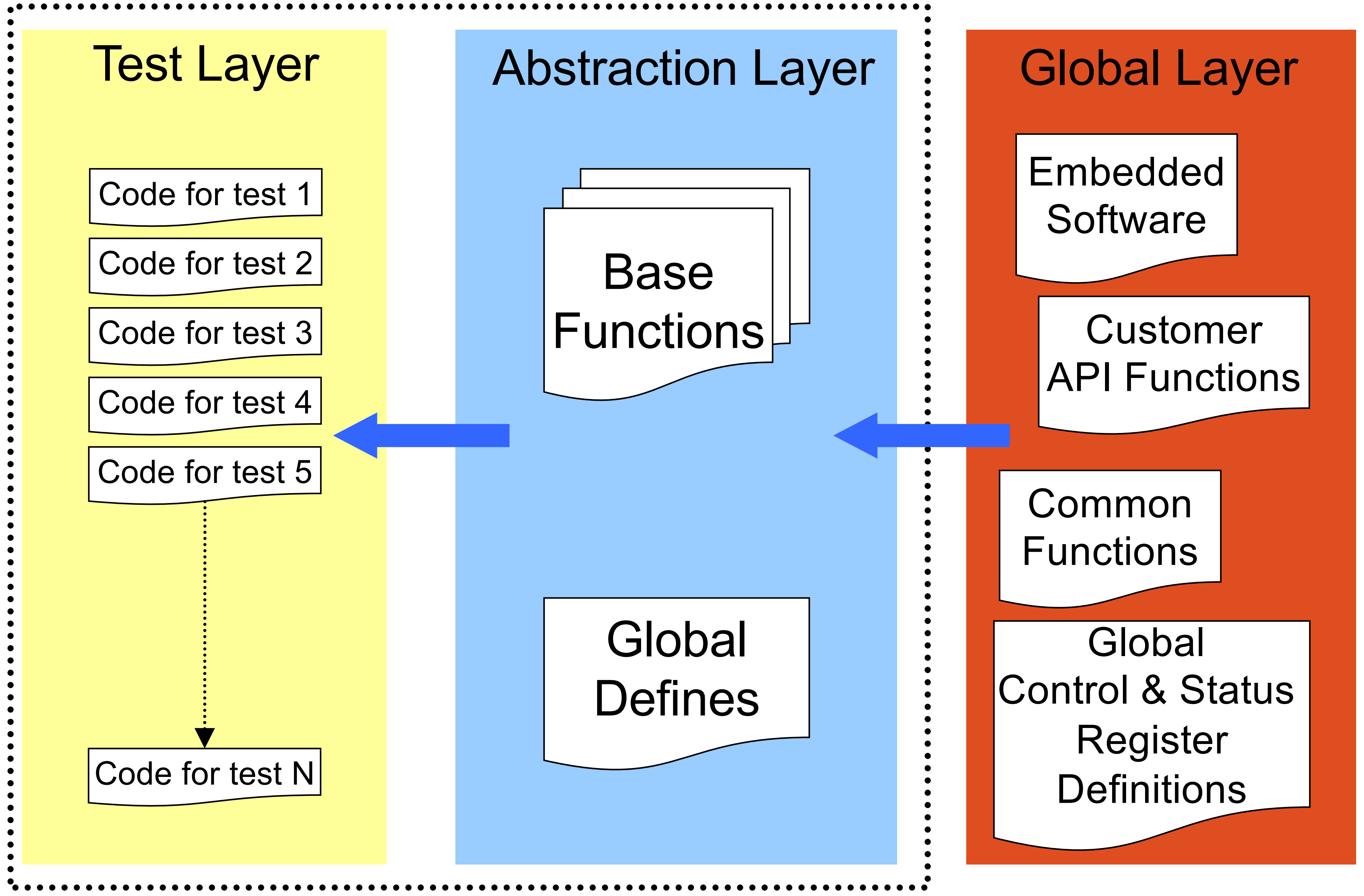
- **What can a typical directed test run on?**

		Product Derivatives		
		CFX4000P	**BO722P**	**M88xx**
Test Platforms	**Silicon**	No	No	No
	VHDL	Yes	Yes	No
	Golden Ref	Yes	Yes	No
	FPGA Emulator	No	No	No
	Hardware Emulator	No	No	No

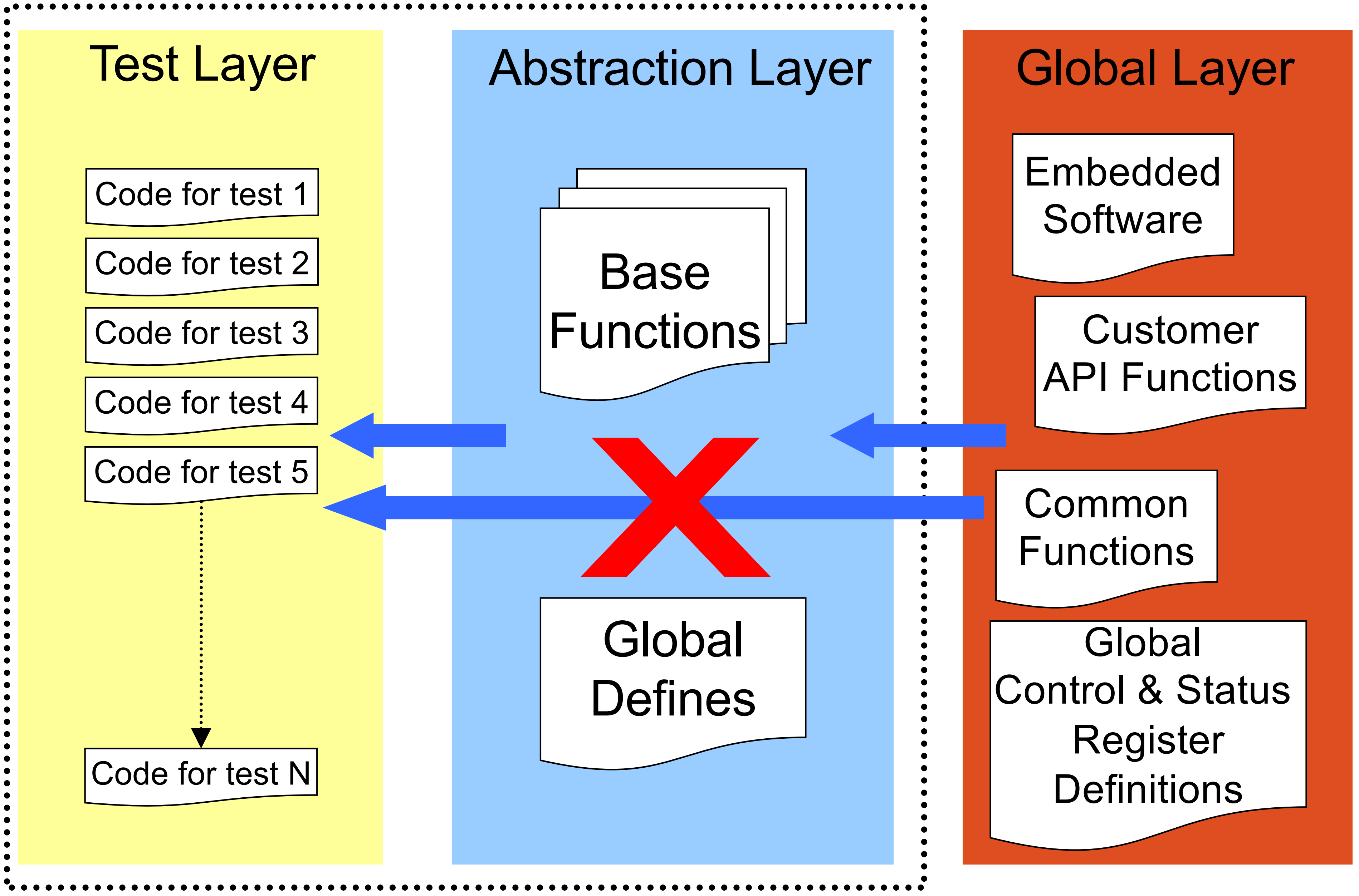
- **What can an ADVM directed test run on?**

		Product Derivatives		
		CFX4000P	**BO722P**	**M88xx**
Test Platforms	**Silicon**	Yes	Yes	Yes
	VHDL	Yes	Yes	Yes
	Golden Ref	Yes	Yes	Yes
	FPGA Emulator	Yes	Yes	Yes
	Hardware Emulator	Yes	Yes	Yes

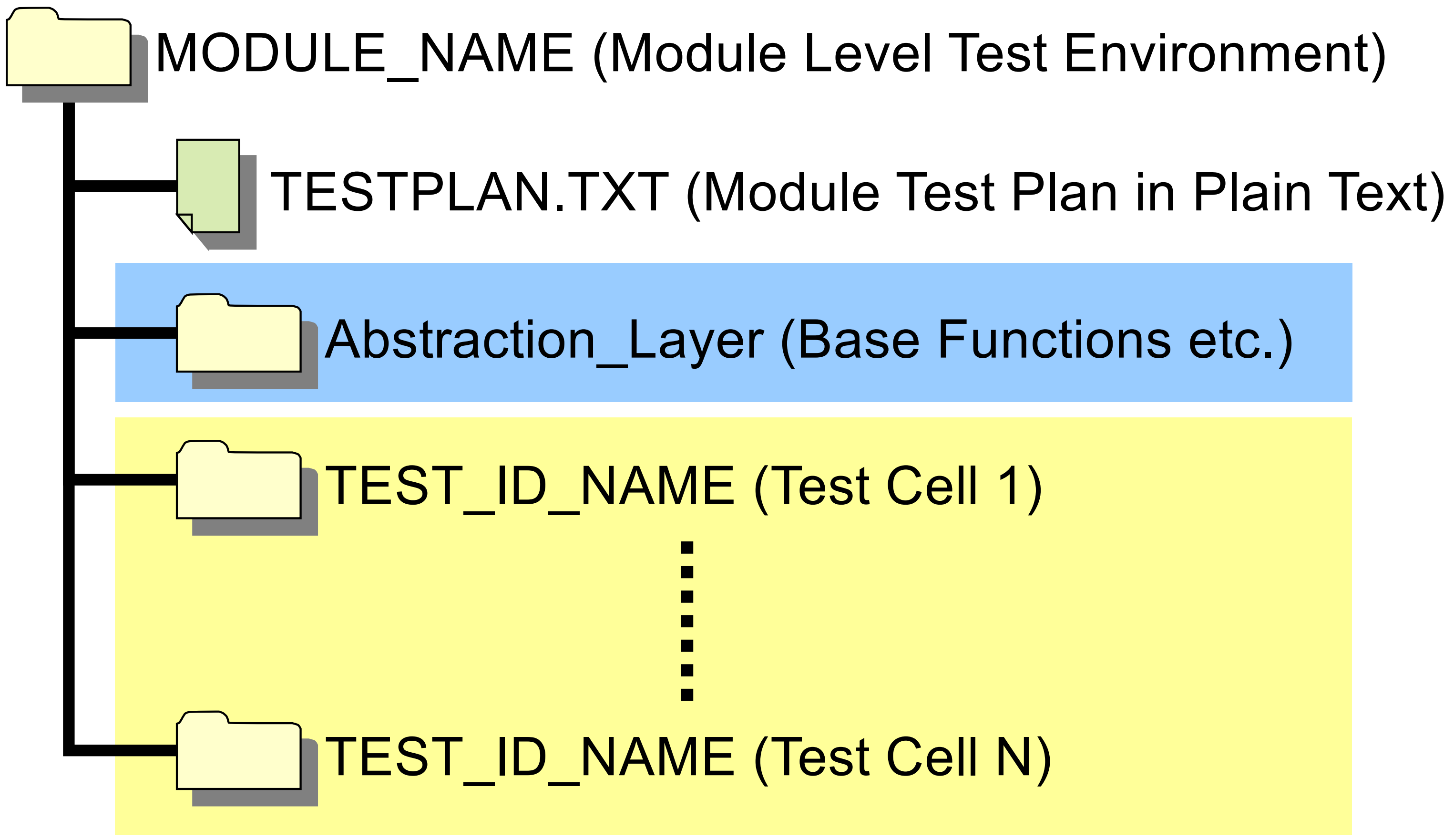
ADVM Test Environment Structure 1



ADVM Test Environment Structure 2



ADVM Test Environment File Structure


MODULE_NAME (Module Level Test Environment)
TESTPLAN.TXT (Module Test Plan in Plain Text)
Abstraction_Layer (Base Functions etc.)
TEST_ID_NAME (Test Cell 1)
TEST_ID_NAME (Test Cell N)

ADVM Code Example 1

Test Layer

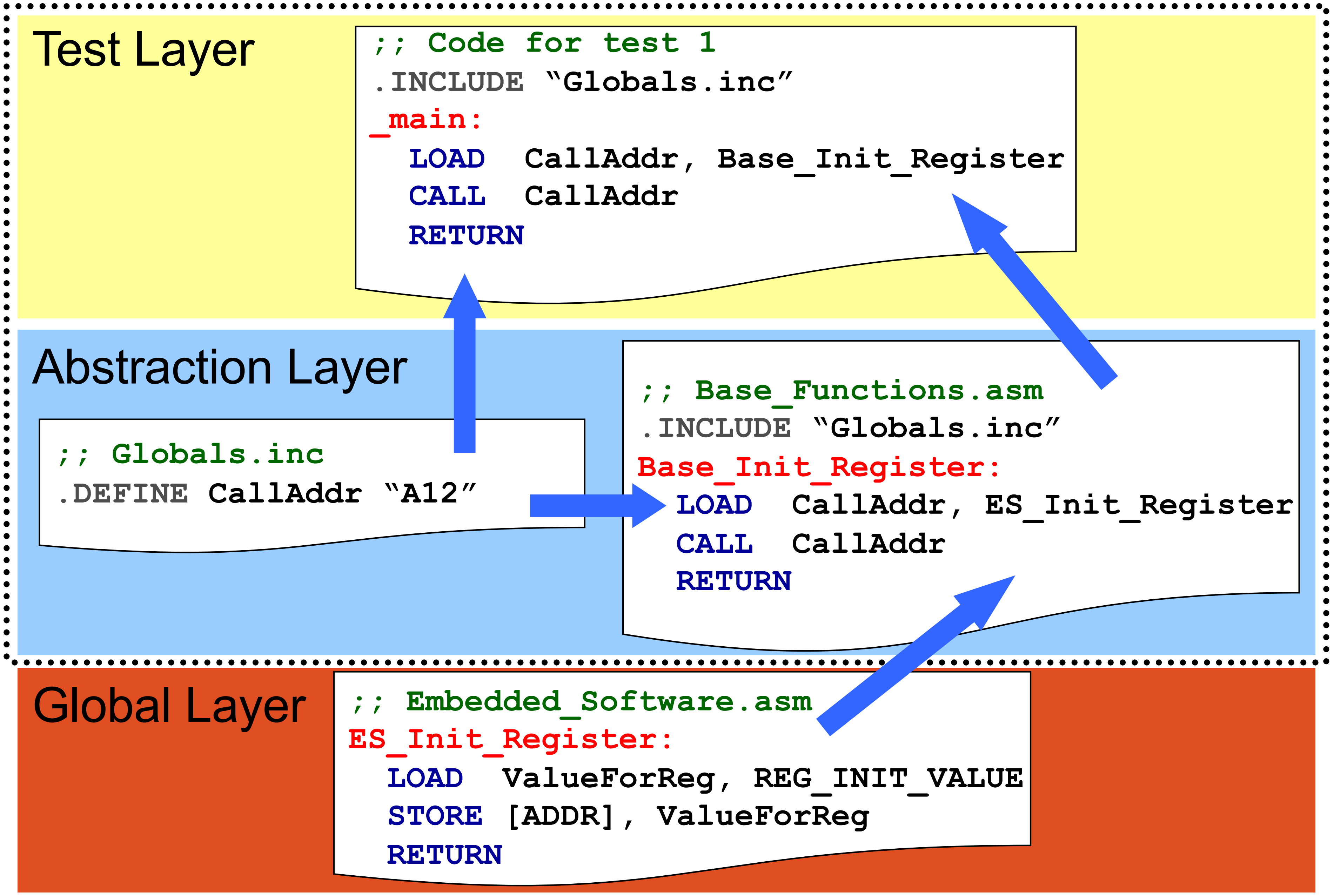
```
;; Code for test 1
.INCLUDE "Globals.inc"
TEST_PAGE .EQU TEST1_TARGET_PAGE
_main:
    :
  INSERT d14, d14, TEST_PAGE, PAGE_FIELD_START_POSITION, PAGE_FIELD_SIZE
    :
```

```
;; Code for test 2
.INCLUDE "Globals.inc"
TEST_PAGE .EQU TEST2_TARGET_PAGE
_main:
    :
  INSERT d14, d14, TEST_PAGE, PAGE_FIELD_START_POSITION, PAGE_FIELD_SIZE
    :
```

Abstraction Layer

```
;; Globals.inc
PAGE_FIELD_SIZE .EQU 5
PAGE_FIELD_START_POSITION .EQU 0
TEST1_TARGET_PAGE .EQU 8
TEST2_TARGET_PAGE .EQU 7
```

ADVM Code Example 2



Test Layer

```
;; Code for test 1
.INCLUDE "Globals.inc"
_main:
  LOAD   CallAddr, Base_Init_Register
  CALL   CallAddr
  RETURN
```

Abstraction Layer

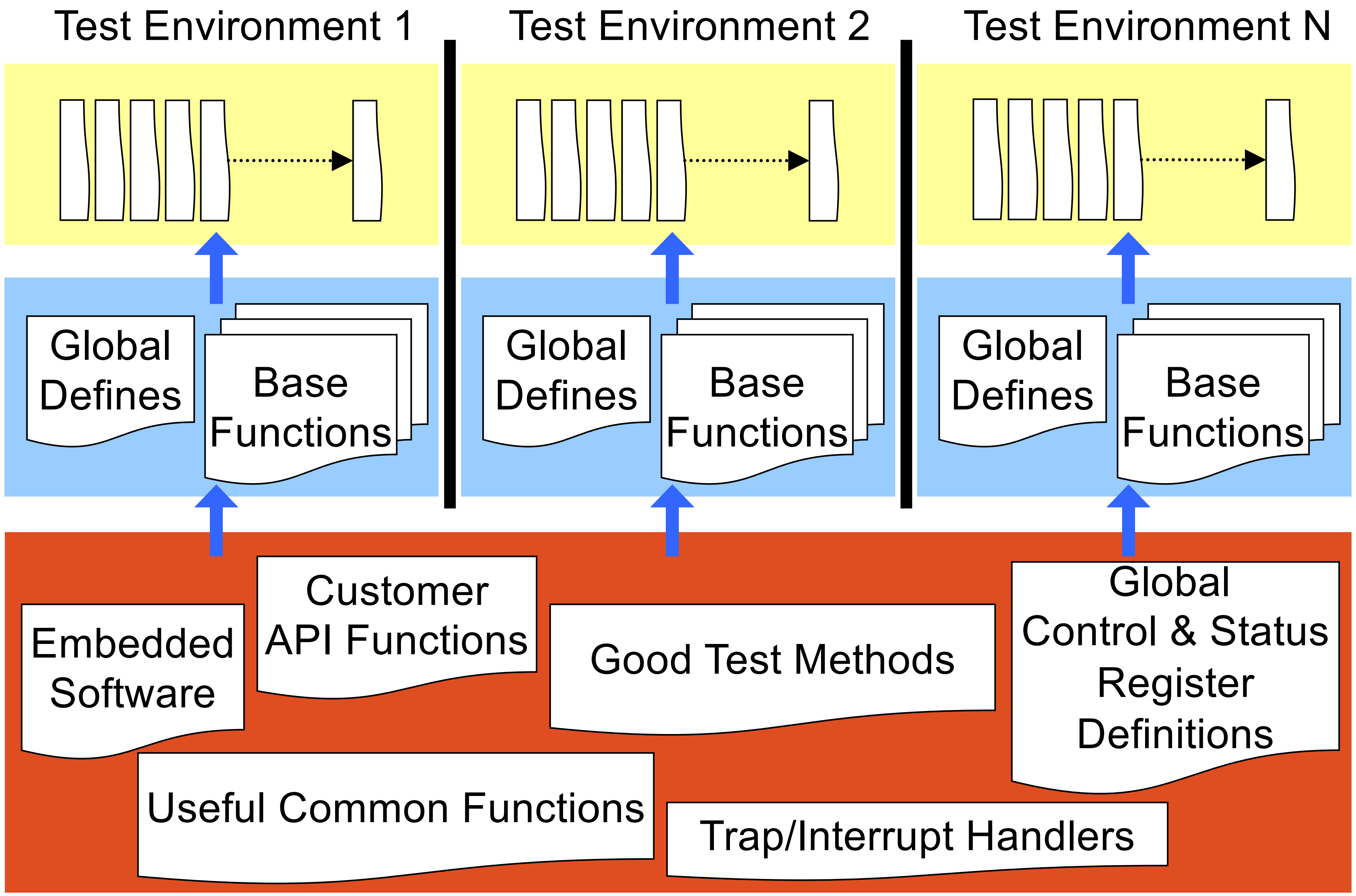
```
;; Globals.inc
.DEFINE CallAddr "A12"
```

```
;; Base_Functions.asm
.INCLUDE "Globals.inc"
Base_Init_Register:
  LOAD   CallAddr, ES_Init_Register
  CALL   CallAddr
  RETURN
```

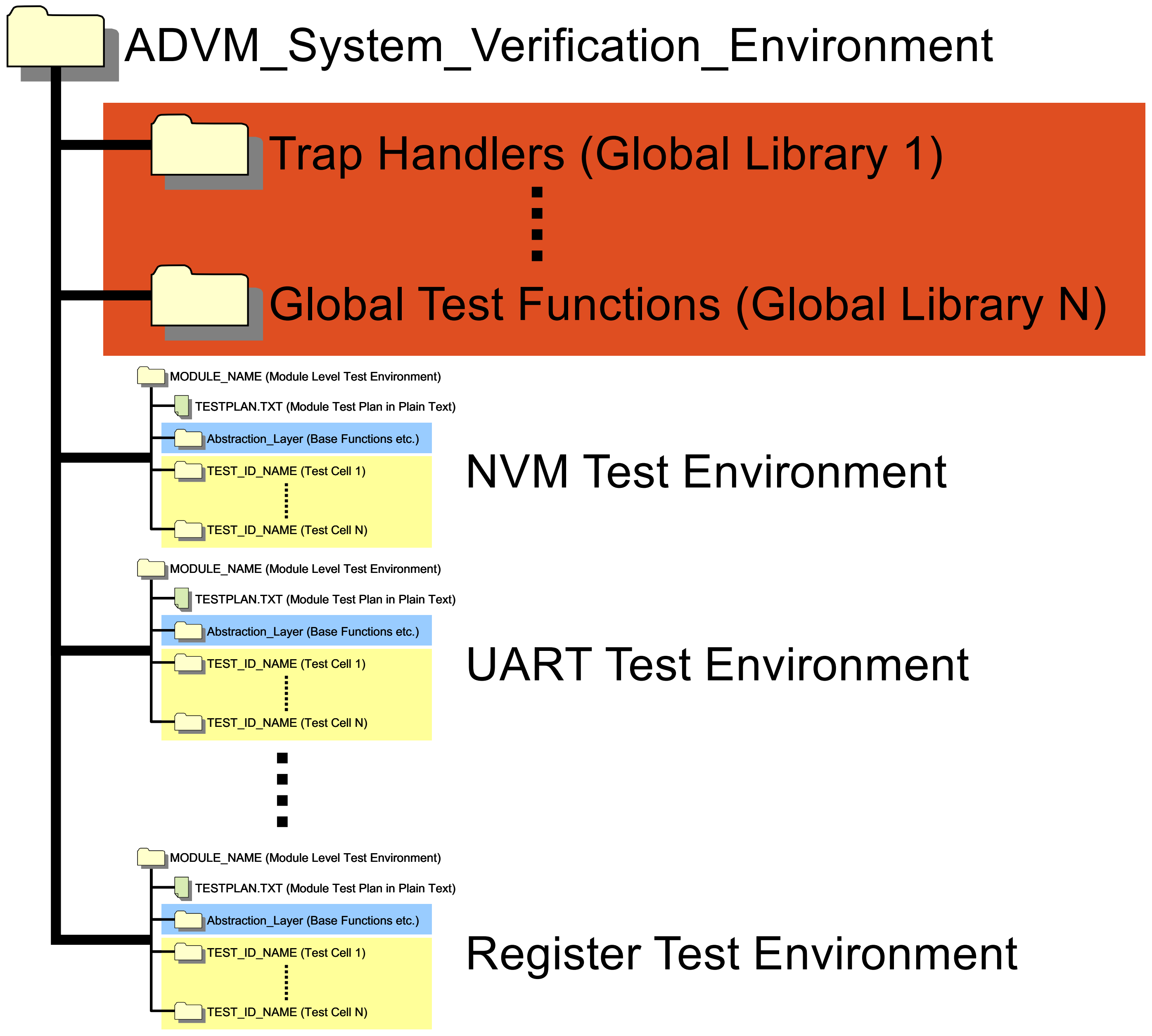
Global Layer

```
;; Embedded_Software.asm
ES_Init_Register:
  LOAD   ValueForReg, REG_INIT_VALUE
  STORE  [ADDR], ValueForReg
  RETURN
```

ADVM Overall Test Environment Structure



ADVM Overall Environment File Structure



ADVM_System_Verification_Environment
Trap Handlers (Global Library 1)
Global Test Functions (Global Library N)
MODULE_NAME (Module Level Test Environment)
TESTPLAN.TXT (Module Test Plan in Plain Text)
Abstraction_Layer (Base Functions etc.)
TEST_ID_NAME (Test Cell 1)
TEST_ID_NAME (Test Cell N)
NVM Test Environment
MODULE_NAME (Module Level Test Environment)
TESTPLAN.TXT (Module Test Plan in Plain Text)
Abstraction_Layer (Base Functions etc.)
TEST_ID_NAME (Test Cell 1)
TEST_ID_NAME (Test Cell N)
UART Test Environment
MODULE_NAME (Module Level Test Environment)
TESTPLAN.TXT (Module Test Plan in Plain Text)
Abstraction_Layer (Base Functions etc.)
TEST_ID_NAME (Test Cell 1)
TEST_ID_NAME (Test Cell N)
Register Test Environment

Revision Control & Release Management

- Owner of the test environment is responsible for:
  - Creating stable releases of the environment
  - Running regressions
  - Maintaining all source code
  - Creating and maintaining required documentation

Summary

- All aspects are resolved by the abstraction layer
- Rapid porting to new derivatives is achieved since the abstraction layer is inherited by all tests
- Provides a consistent method for the creation of tests
- Provides a better method of test control, that allows specific corner cases to be investigated (as the need arises)
- Once the base functions for each environment have been created the test development time is significantly reduced
- This methodology can be adopted independently of existing tests

Any Questions?

Email to: john.macbeth@verilab.com