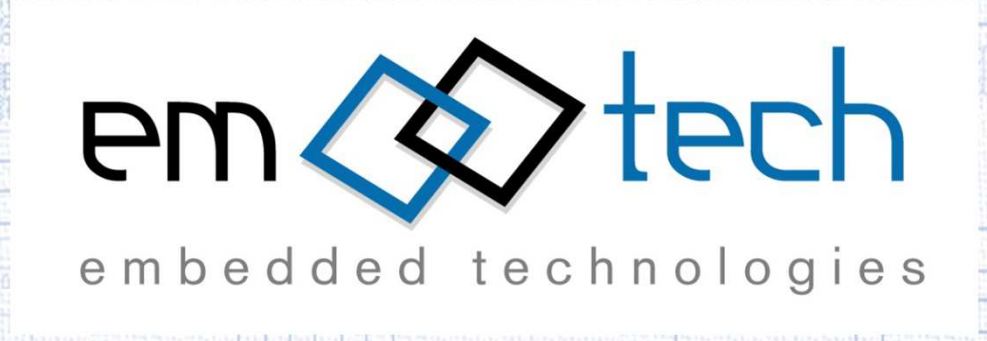


Diseño y verificación de sistemas complejos con FPGA

Conceptos básicos:

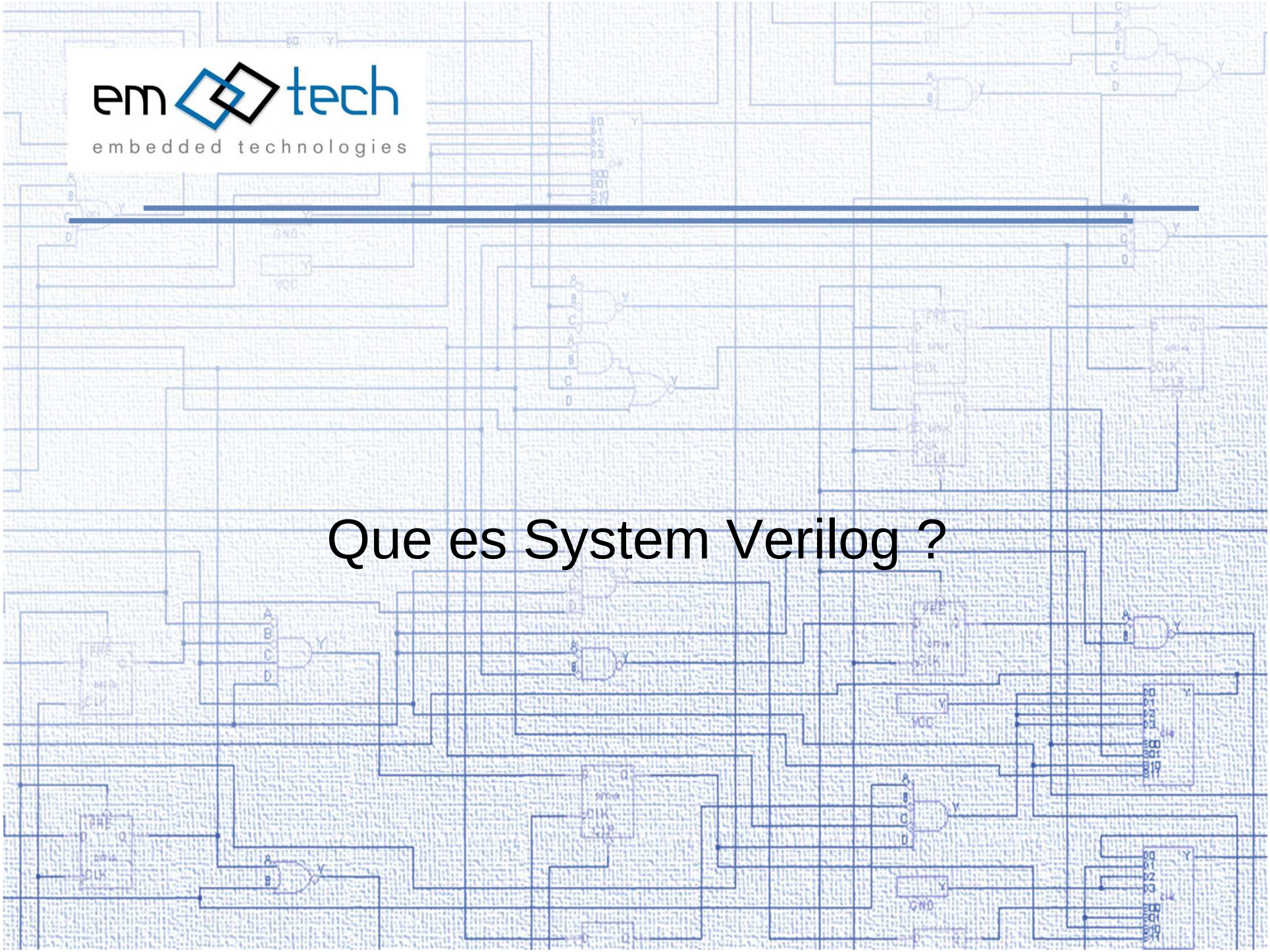
System Verilog, ABV Y OVM/UVM

Germán B. Berterreix

em tech
embedded technologies

System Verilog

- System Verilog
- Assertion-Based Verification
- OVM / UVM



Que es System Verilog ?

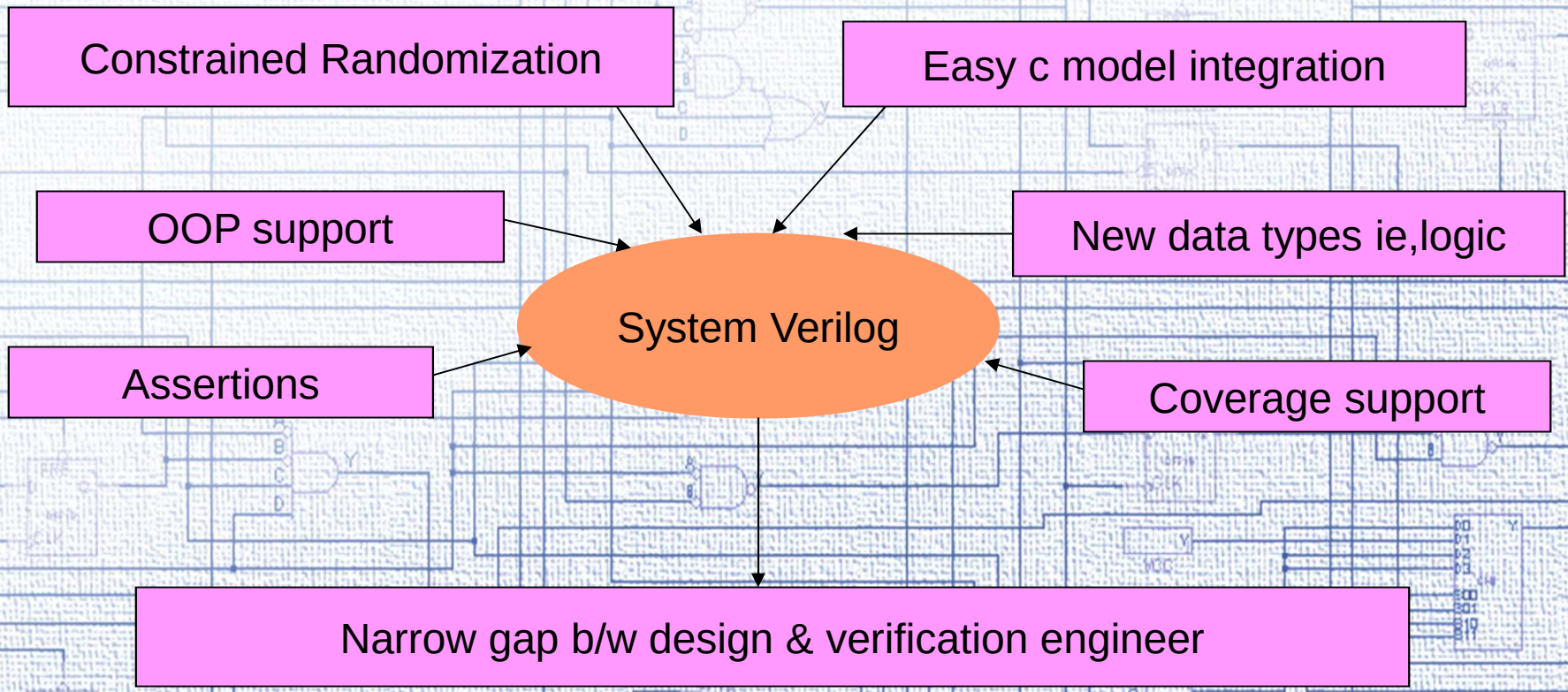
Que es System Verilog ?

- System Verilog es un lenguaje de descripción de hardware y verificación.
- Posee características heredadas de Verilog HDL, VHDL, C y C++
- Añade funciones ampliadas para Verilog
- System Verilog es un súper conjunto de Verilog
- Es compatible con todas las característica de Verilog

em tech
embedded technologies

Por que System Verilog ?

Por que System Verilog?

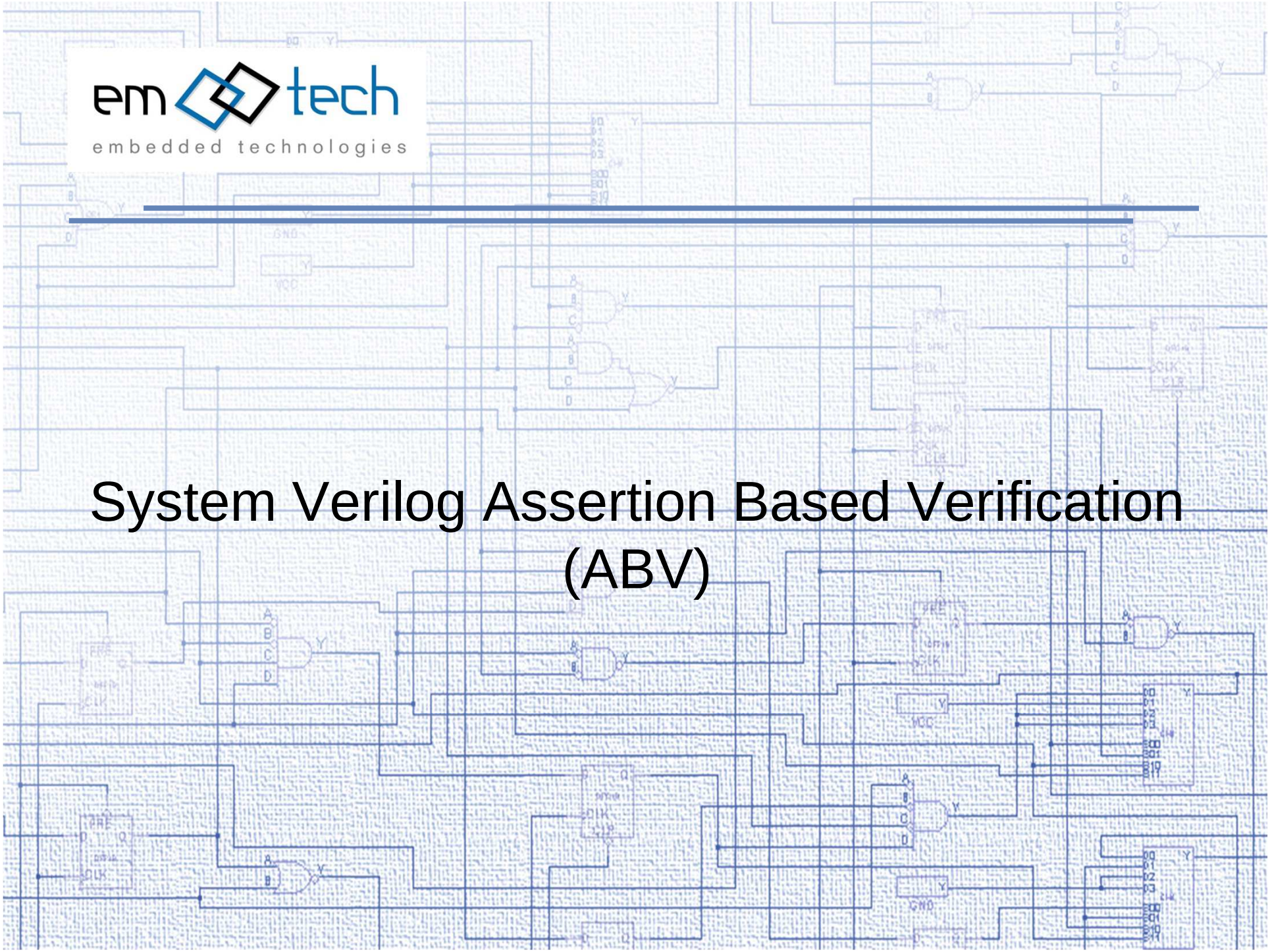


System Verilog: Algunas características

- Diseño a nivel de modulo
- Simulación a nivel de compuertas
- Verificación a nivel de sistema
- El tipo de datos es de dos estados – 0, 1
- Las memorias son dinámicas por naturaleza
- Mejor manejo de memoria
- Menor código RTL y de verificación, por lo tanto menor numero de bugs
- Mayor nivel de abstracción

em tech
embedded technologies

System Verilog Assertion Based Verification (ABV)



Aserciones – Que son?

- Una aserción es una expresión que, si es falsa, indica un error
- Es usada para debugging atrapando los errores “can't happen” (Es decir los errores que no pueden ocurrir)
- Una sentencia condicional que comprueba un comportamiento especifico y muestra un mensaje si este ocurre
- Generalmente son utilizadas como “monitores” para buscar malos comportamientos
  - También utilizados para alertar sobre ciertos comportamientos en especial
- Para nuestro propósito:
  - Una aserción es una descripción concisa de un comportamiento deseado o no deseado.

Assertions-Based Verification

- Las soluciones utilizando Assertion-based verification (ABV) han estado ganando popularidad
  - Las aserciones son declaraciones de suposiciones del diseñador o intención del diseño
  - Las aserciones deben ser inherentemente reutilizables
- Esto, NO REEMPLAZA, los test de simulación tradicionales
  - Tanto la observabilidad como la controlabilidad del diseño pueden ser mejoradas
  - Las aserciones permiten la verificación formal

System Verilog: Ejemplos

Ejemplo1:

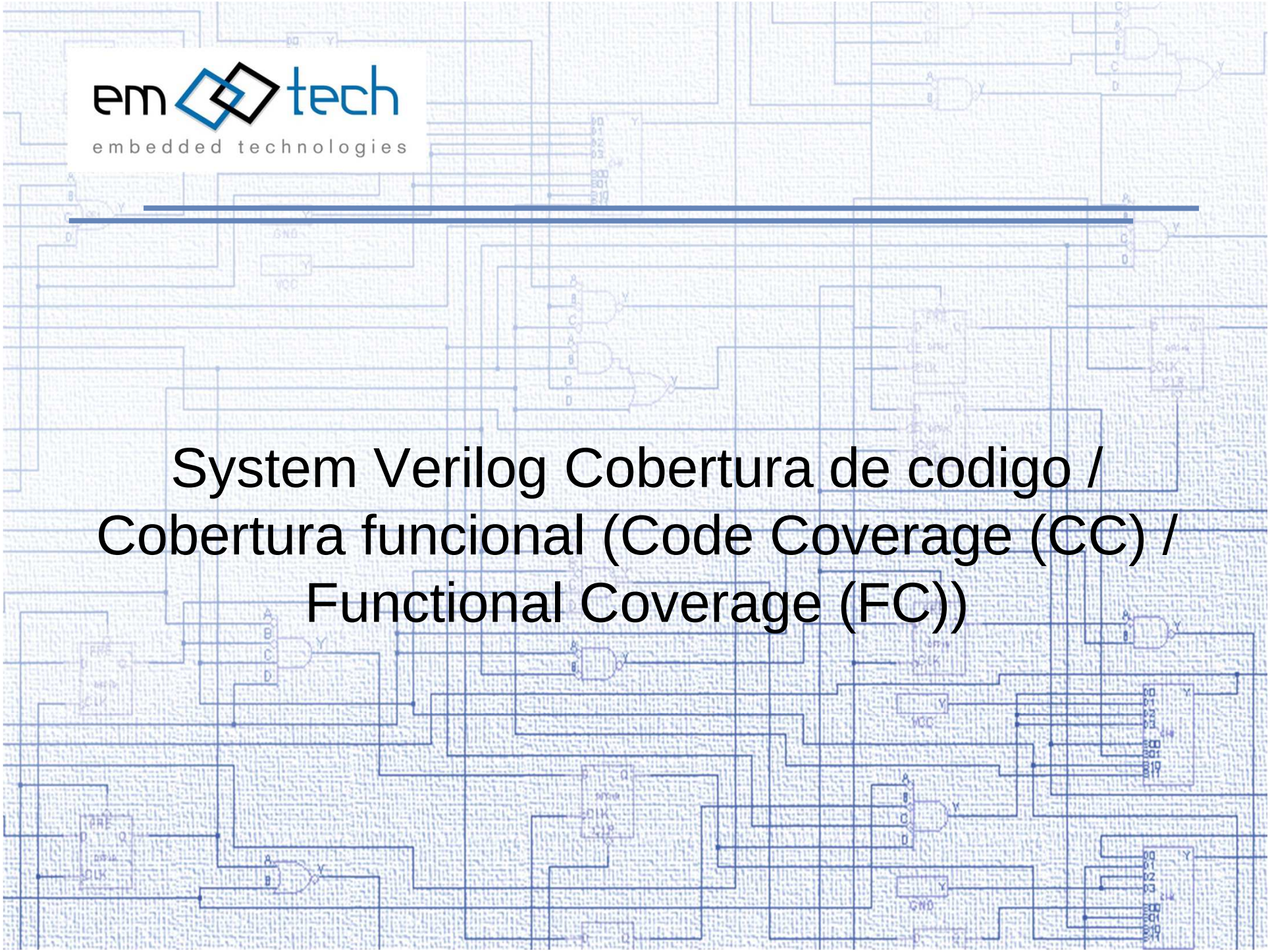
Queremos saber cuantas veces es violada la condición de que el valor “A” sea igual al valor de “B”.

```
Igual: assert always (A == B) $display(“OK, A es igual a B”);
                else $error(“Algo está mal”);
```

Ejemplo2:

Verificar que los valores de memoria no permanecen fijos durante toda la simulación

```
no_change_addr : cover {addr /= prev(addr)};
```



System Verilog Cobertura de codigo / Cobertura funcional (Code Coverage (CC) / Functional Coverage (FC))

System Verilog: CC / FC

Definición:

La **cobertura de código** (CC) es una medida (porcentual) en las pruebas de software que mide el grado en que el código fuente de un programa ha sido testeado. Sirve para determinar:

- La calidad del test que se lleva a cabo
- Las partes criticas del código que no han sido testeadas y las que si han sido testeadas

Tipos de cobertura:

- Líneas / Ramas
- Funciones
- Condiciones

La **cobertura funcional** (FC) por su parte, percibe el diseño de un usuario o de un punto de vista del sistema.
Algunas preguntas típicas:

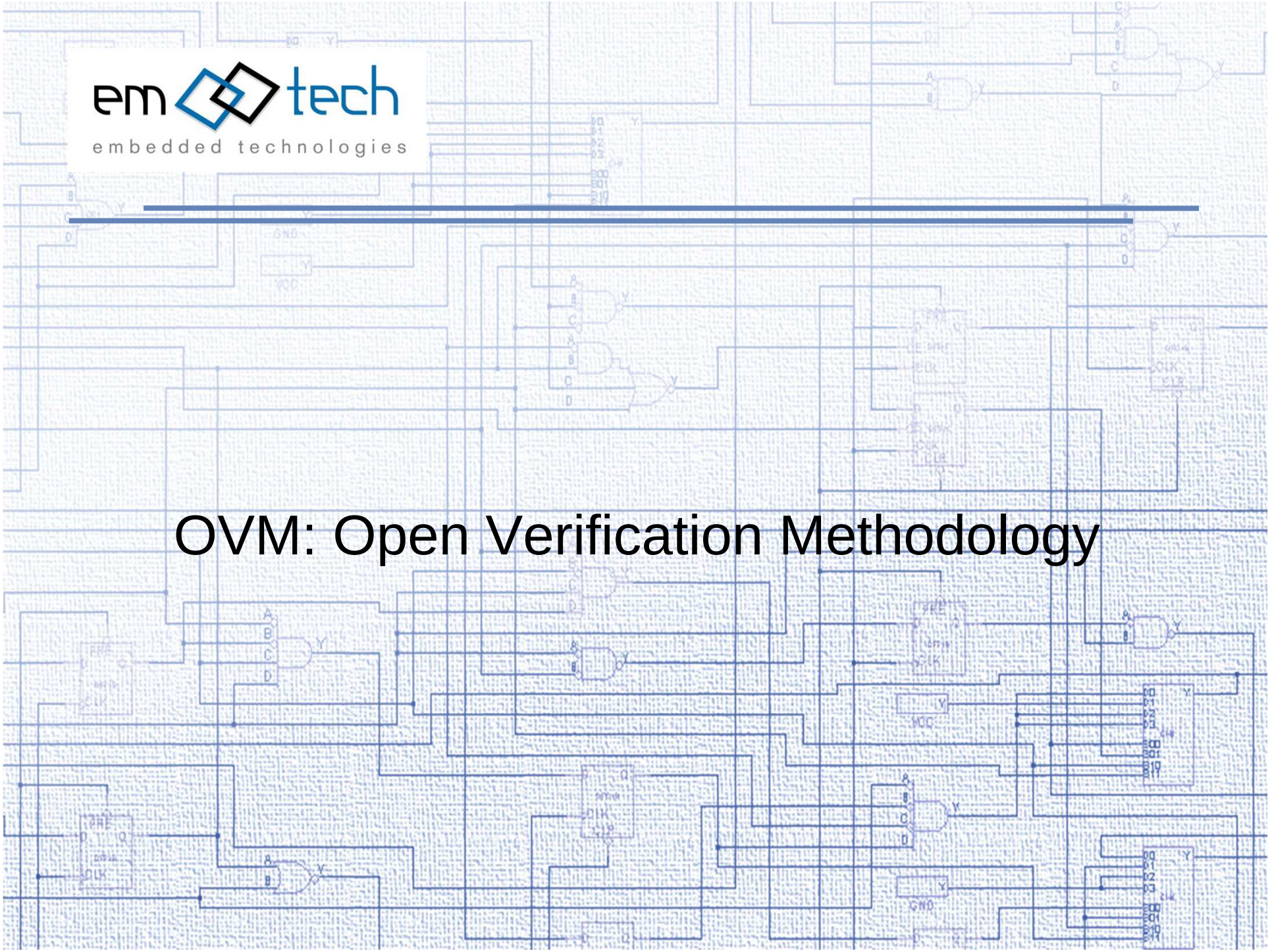
- Se ha cubierto la totalidad de los escenarios típicos?
- Se han cubierto los casos de error?
- Se han cubierto los “corner cases”?
- Se han cubierto los protocolos?

Pero existen 2 preguntas mágicas que todo equipo de diseño deben hacerse y responderse:

- Mi chip funciona correctamente?
- He terminado de verificar el chip?

em tech
embedded technologies

OVM: Open Verification Methodology



System Verilog: OVM

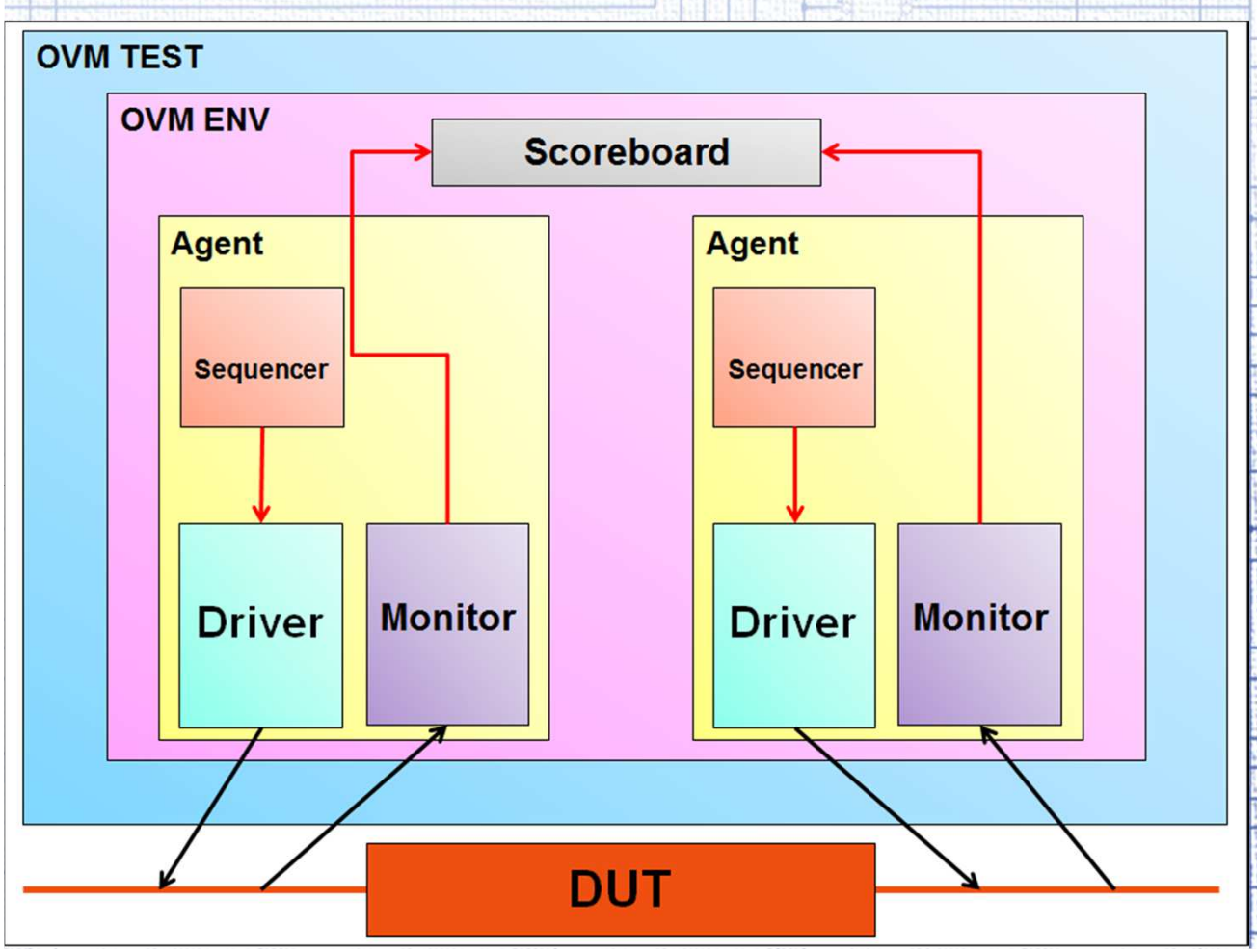
Definición:

OVM (Open Verification Methodology) es una metodología para la verificación funcional que utiliza SystemVerilog y sus librerias, y principalmente utilizando la simulación

OVM fue creado por Cadence y Mentor Graphics, basado en la metodología de verificación existentes en ambas compañías.

Los conceptos de reutilización utilizados en OVM se derivan principalmente de URM (Universal Reuse Methodology)

OVM



Como podemos observar al utilizar OVM, tenemos dos áreas bien definidas, OVM TEST y OVM ENV.

De esta forma vemos que el DUT o DUV es una instancia mas dentro de la utilización de OVM.

- Agent
- Sequencer
- Driver
- Monitor

System Verilog: UVM

UVM (Universal Verification Methology) es una metodología estandarizada para la verificación de diseños de circuitos integrados. UVM se deriva principalmente de OVM.

La biblioteca de clases UVM trae mucha automatización para el idioma SystemVerilog. y al contrario que las anteriores metodologías desarrolladas independientemente por los vendedores de simuladores, es un estándar Accellera con el apoyo de varios proveedores: Aldec, Cadence, Mentor y Synopsys.


em tech
embedded technologies
Gracias