# A methodology for SoC top-level validation

Electronic Design Process '02

Sylvan Dissoubray

Monterey



## Corporate Profile

- Founded in Nov.1999
- 15+ M€ raised in 2000 and 2001
- 100 employees in 5 countries
- 30 consultants
- 30+ large corporate customers
- 200+ seats deployed

- Major shareholders
  - *Management & staff*
  - *GALILEO Partners*
  - *CDC Innovation*
  - *INNOTECH*
  - *Advanced Capital Europe*
  - *Intel Capital*
  - *THALES Corporate Ventures*
  - *INRIA-Transfert*

## Esterel Studio Applications

- System Architecture Specifications
  - Wireless core platforms
  - Consumer electronics core platforms
  - Microprocessor chipsets
- Top-level Validation of SoCs
  - Wireless platforms
  - Set top boxes
  - DVD chipsets
  - MPEG decoders
- Telecom & Security Protocol Development (SW / UML)
  - UMTS: RRC, RLC layers
  - Bluetooth
  - Secured military communication
  - Smart-card security
  - HiperLAN2



## SoC top-level validation challenges

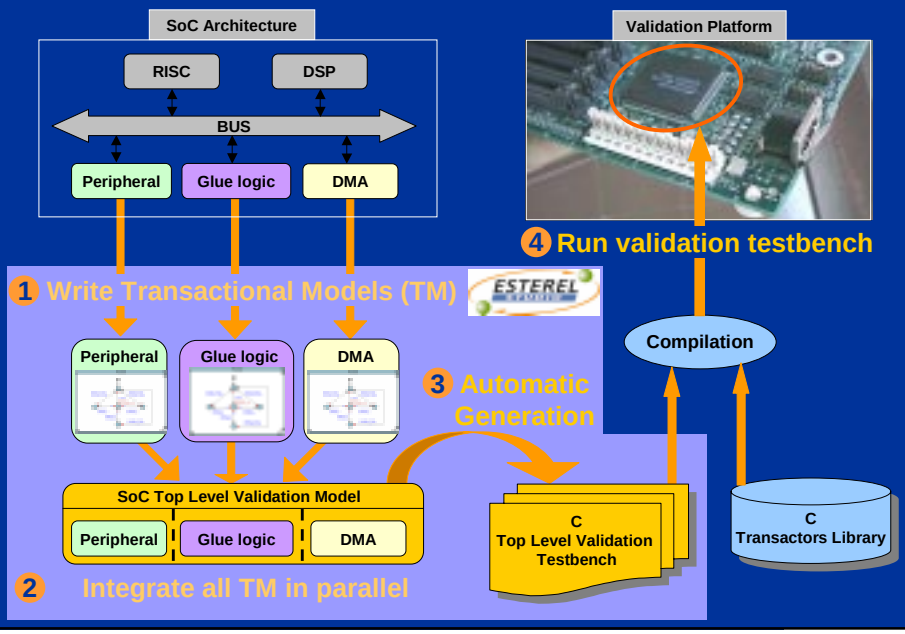
- SoC's maximized reuse but design costs did not drop down
  - Because cost moved to SoC's functional verification consumes up to 70% overall design cycle budget (ITRS)
- Why is it so difficult to verify SoC's ?
  - Multi-core, multi-service, multi-traffic SoC's => extreme concurrency
  - More reuse of blocks and IPs => less and less knowledge of components internal behavior
- The SoC integrator dilemma
  - Functional coverage
    requires a lot of test cases,
    it is difficult to reach corner cases
  - Time
    writing test cases is time consuming,
    running integration level tests is slow

## SoC Top Level Validation: objectives

- Generate automatically a Test Suite to validate IP inter-operability and concurrency
- High level transactional approach
- Obtain the best coverage in the most effective manner

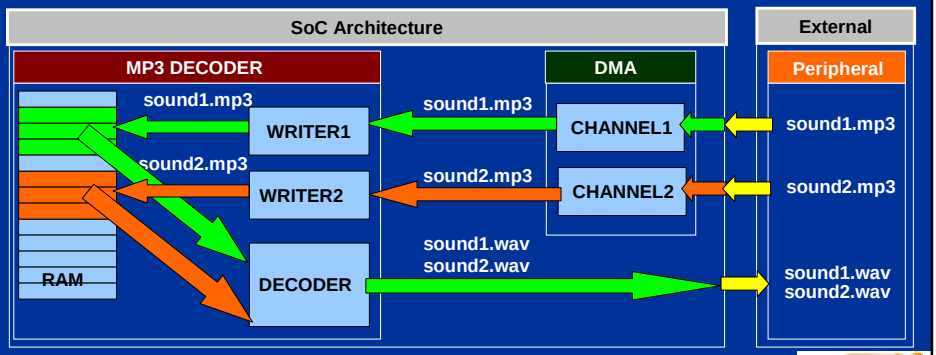



## SoC top-level validation method



# Example: top-level validation of a MP3 SoC

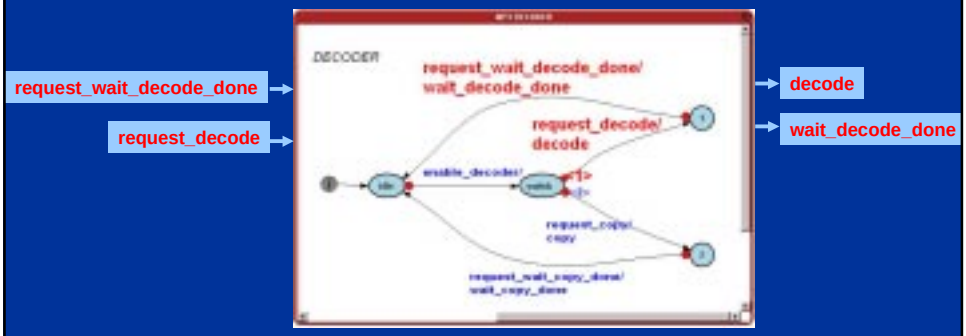
- **The MP3 system**
  - The DMA block can ***transfer***
  - The WRITER can ***write*** into RAM
  - The DECODER can:
    - ***Copy*** the stream
    - ***Decode*** the stream into another format (wav, pcm, etc.)
  - WRITER and DECODER are concurrent and synchronized: the WRITER is responsible to start the DECODER







# 1 Write Transactional Models (TM): the DECODER




- Internals of block behavior need not be described in Esterel Studio
- not Bus Cycle Accurate level, not Cycle Accurate level

## 2 Put All TM Models in Parallel




## 3 Automatically Generate TM coverage tests

## 3 Generated tests cover 100% states of the TM

```
%-------------------------------------------------------
% Test 4
% Coverage 1
%-------------------------------------------------------
!reset ;
;
request_transfer request_write ;
request_wait_transfer_done request_decode ;
request_wait_write_done request_wait_decode_done ;
%-------------------------------------------------------
```

TM Inputs sequences

**DUV coverage report**

- Coverage on D:/Esterel-Tech/Docs/Demos/V02/SC/TLV/DUV/OneChannelBreaker6_DUT
- Reachable states number: 28
- Reachable transition number: 131
- Files are CHRONOLOGICALLY sorted

| Scenario | Scenario coverage | | | | Cumulated coverage | | | |
|---|---|---|---|---|---|---|---|---|
| | States | % | Transitions | % | States | % | Transitions | % |
| running_long | 28 | 100 | 131 | 100 | 28 | 100 | 131 | 100 |

*Created on 02/28/2002 16:14:36 by Esterel Studio version 1.1.6*

Coverage Report



## 3 Collected Outputs

```
! reset;
;
% Outputs:
request_transfer request_write ;
% Outputs: transfer write
request_wait_transfer_done request_decode ;
% Outputs: wait_transfer_done decode
request_wait_write_done request_wait_decode_done ;
% Outputs: wait_write_done wait_decode_done
```

Script to C
or other languages

**C Test Case**

```
void run_test()
{
  transfer();
  write();
  wait_transfer_done();
  decode();
  wait_write_done();
  wait_decode_done();
}
```

## 4 From Transactional Level to Simulation Platform

**C Top Level Validation Testbench**

```
void run_test() {
  transfer();
  write();
  wait_transfer_done();
  decode();
  wait_write_done();
  wait_decode_done();
}
```

TRANSACTIONS

**Transactor Libraries**

```
void transfer()
{
  *(*int)DMA_CHANNEL_REG = (int) REG_INIT;
   initPort(0x00000001);
   …
}

void write() {
…
}
```

**Simulation Platform**



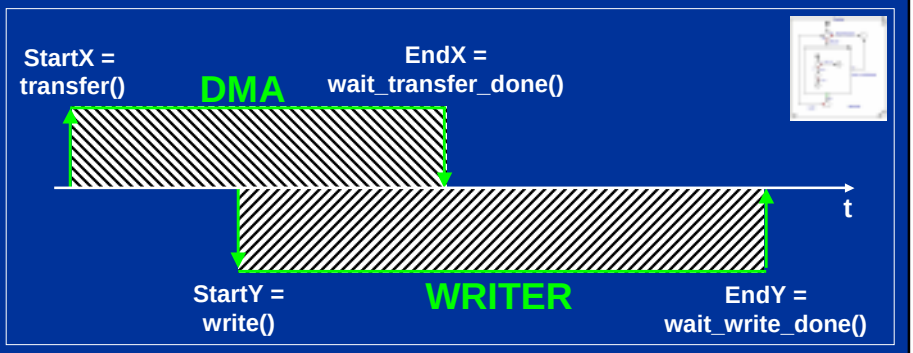

## 4 Transaction execution profile on a CPU or simulation platform

- The generated testbench cover interoperability and concurrent behaviors of SoC internal blocks

```
void run_test()
{
  transfer();
  write();
  wait_transfer_done();
  decode();
  wait_write_done();
  wait_decode_done();
}
```

# Refinement 1: Adding constraints to direct test generation

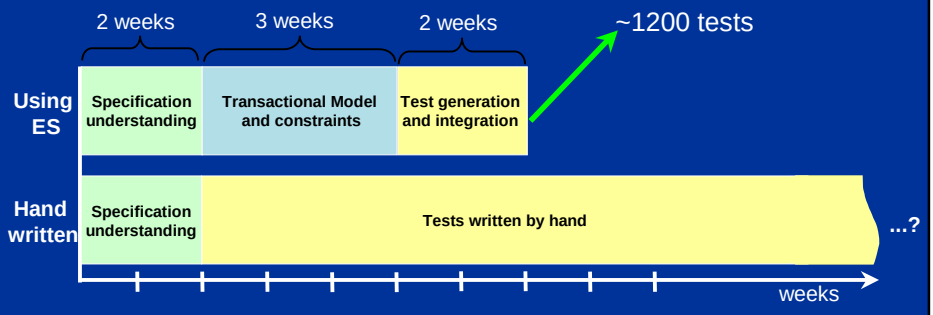
- Fine Tune concurrency
- Take into account system constraint and restriction
- Match the test plan
- Use of existing libraries






# Productivity Gain

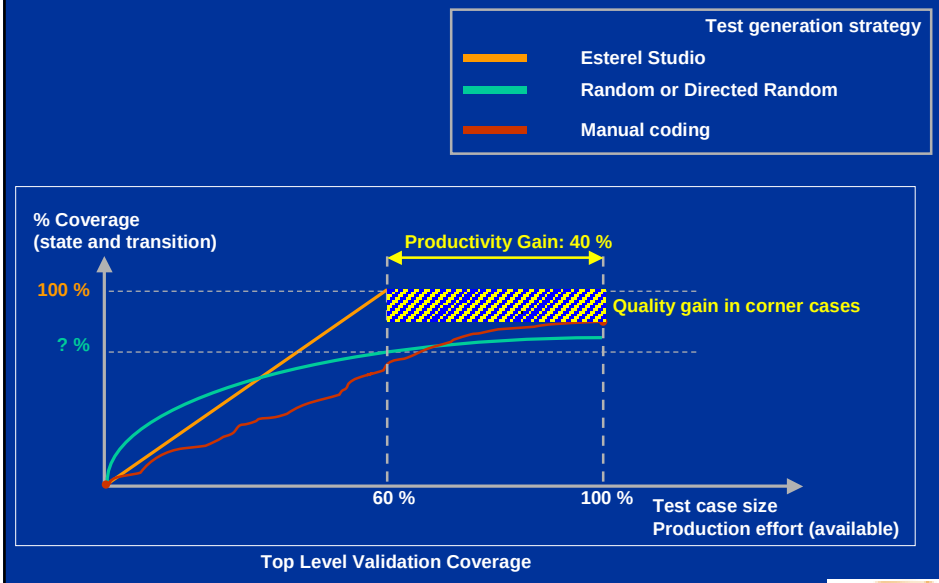
- Schedules




- Generated Test Cases

| Files | Configuration | | | | | | | Model | | Generated Tests | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | pes1 | scd1 | vld1 | pes2 | scd2 | vld2 | alter | states | trans | cov. type | # of tests |
| dec7_conf1_seqdc_S_out.esi | X | X | X | | | | | 121 | 449 | state | 50 |
| dec7_conf1_seqdc_T_out.esi | X | X | X | | | | | | | transition | 364 |
| dec7_conf12_seqdc_S_out.esi | X | X | | X | X | | | 486 | 5043 | state | 278 |
| dec7_conf12va_seqdc_S_out.es | X | X | X | X | X | X | X | 7115 | 85620 | state | 1188 |

# Generated test cases are compact and efficient



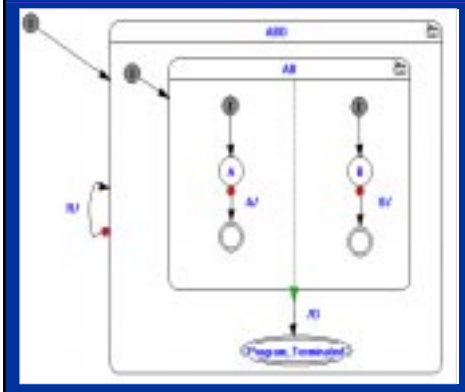

Top Level Validation Coverage




# Esterel Studio SoC top-level validation: benefits summary

- Applicable to actual CPUs as well as any simulation platform
- Requires minimal blocks and IPs documentation
- The generated testbench is extremely efficient to cover concurrent behaviors of SoC internal blocks and interoperability
- Applicable to multi-core designs
- Generated testbench shows excellent coverage vs. size efficiency and is good at reaching corner cases

## Esterel Studio



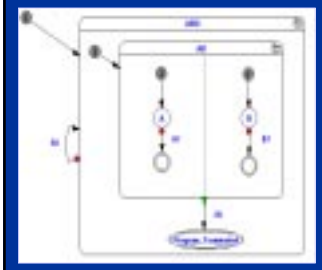
```
loop
   [ await A  ||  await B ] ;
   emit O
each  R
```

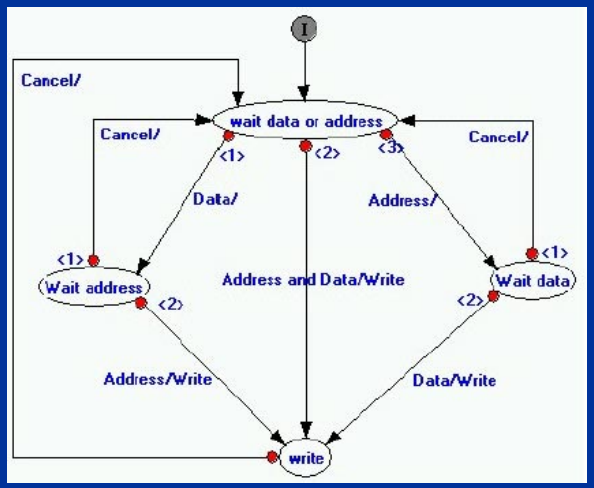
## C++ Esterel style

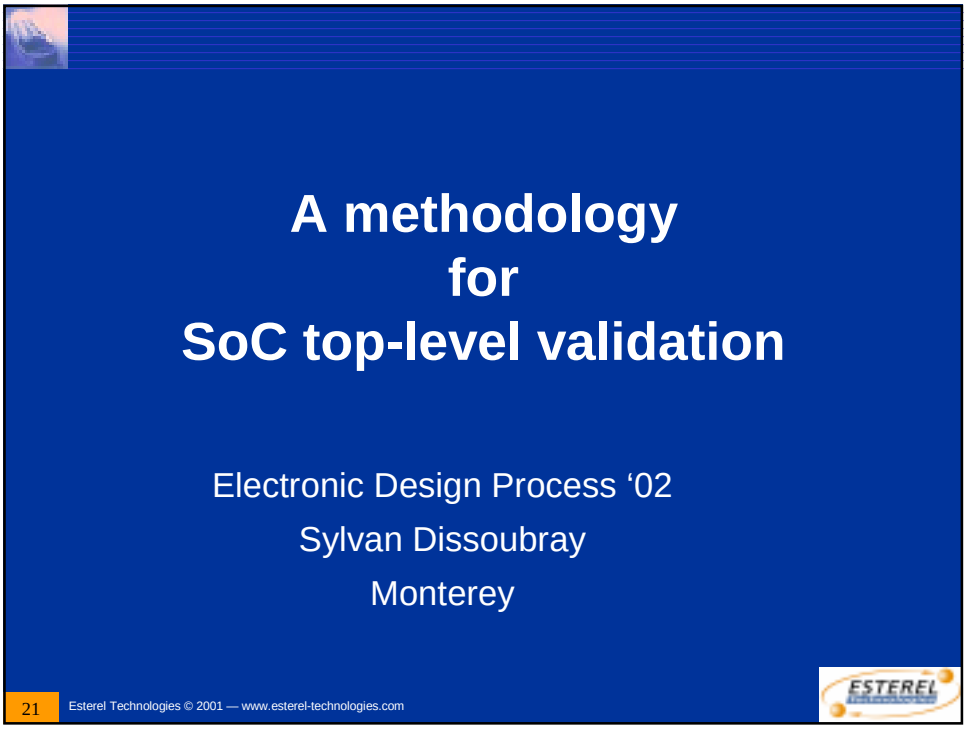
```
DWORD WINAPI Thread_A(LPVOID lpstart){
   for (;;)
    { if (R) { R = 0; ExitThread(1); }
    if (A) { A = 0; ExitThread(0); } }
}
DWORD WINAPI Thread_B(LPVOID lpstart){
    for (;;) {
        if (R) { R = 0; ExitThread(1); }
        if (B) { B = 0; ExitThread(0); } }
}
int APIENTRY WinMain(HINSTANCE hInstance, HINSTANCE
hPrevInstance,
    LPSTR lpszCmdLine, int nCmdShow) {
    DWORD x=0;
    DWORD ExitCode[2];
    HINST = hInstance;
    hthread[2] = CreateThread(NULL,0,
                (LPTHREAD_START_ROUTINE)
ControlPanel,
                (LPVOID) &x,0,&dwThreadId[2]);
    for (;;) {
        hthread[0] = CreateThread
              (NULL,0,(LPTHREAD_START_ROUTINE)
Thread_A,
               (LPVOID) &x,0,&dwThreadId[0]);
        hthread[1] = CreateThread
              (NULL,0,(LPTHREAD_START_ROUTINE)
Thread_B,
              (LPVOID) &x,0,&dwThreadId[1]);
        WaitForMultipleObjects(2,hthread,TRUE,INFINITE);
        GetExitCodeThread(hthread[0], &ExitCode[0]);
        GetExitCodeThread(hthread[1], &ExitCode[1]);
        if (!ExitCode[0] && !ExitCode[1]) {
            O();
          for (;;) if (R) break;  }   }   }
```




## State Notion






A methodology
for
SoC top-level validation
Electronic Design Process '02
Sylvan Dissoubray
Monterey
21
Esterel Technologies © 2001 — www.esterel-technologies.com
ESTEREL