



Smalltalk Debug Lives in the Matrix



Loïc Lagadec, Damien Picard
Loic.lagadec@univ-brest.fr



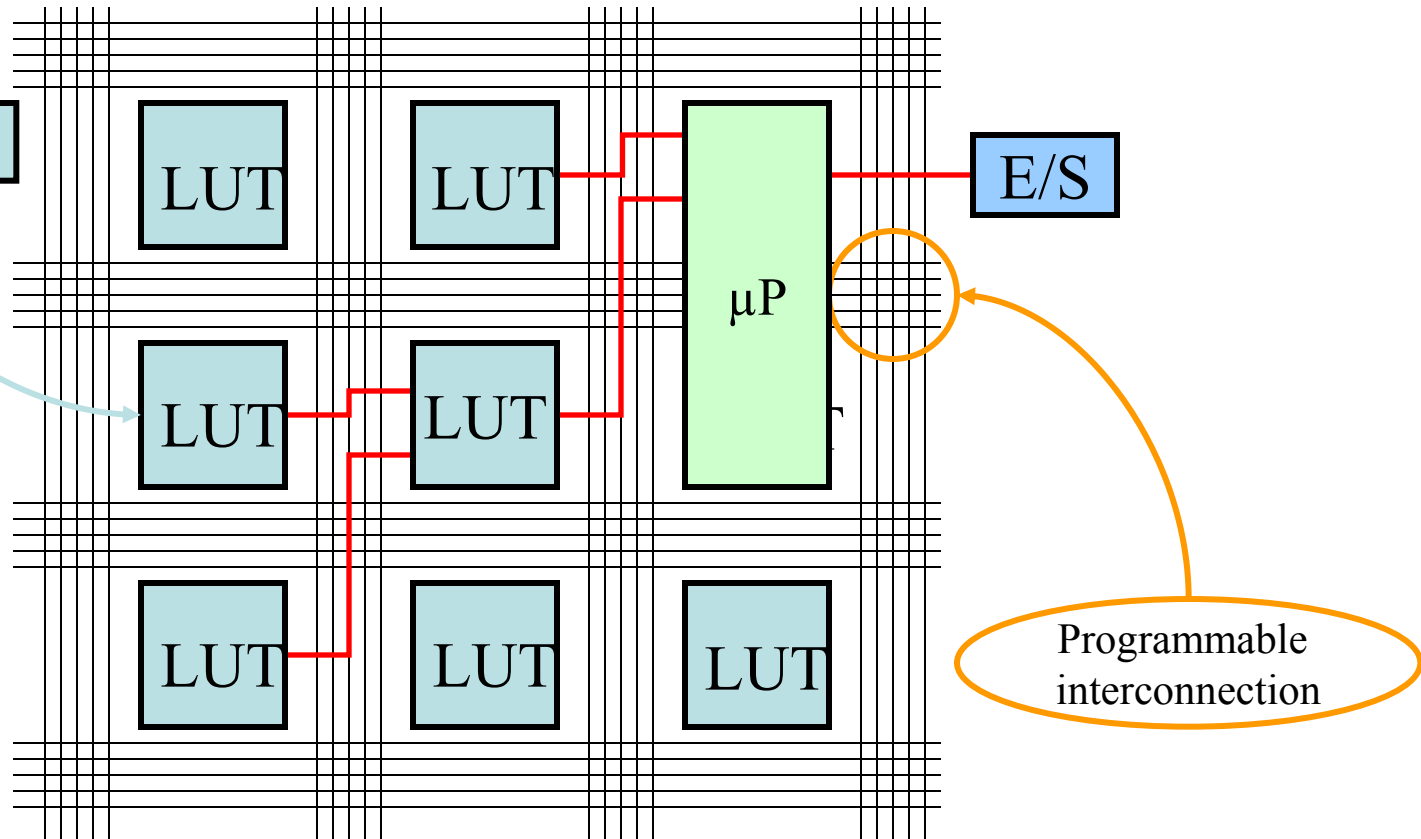
What Matrix?



“Flexible” hardware
Time to market



Hard to program
Hard to debug



What Matrix?



“Flexible” hardware
Time to market

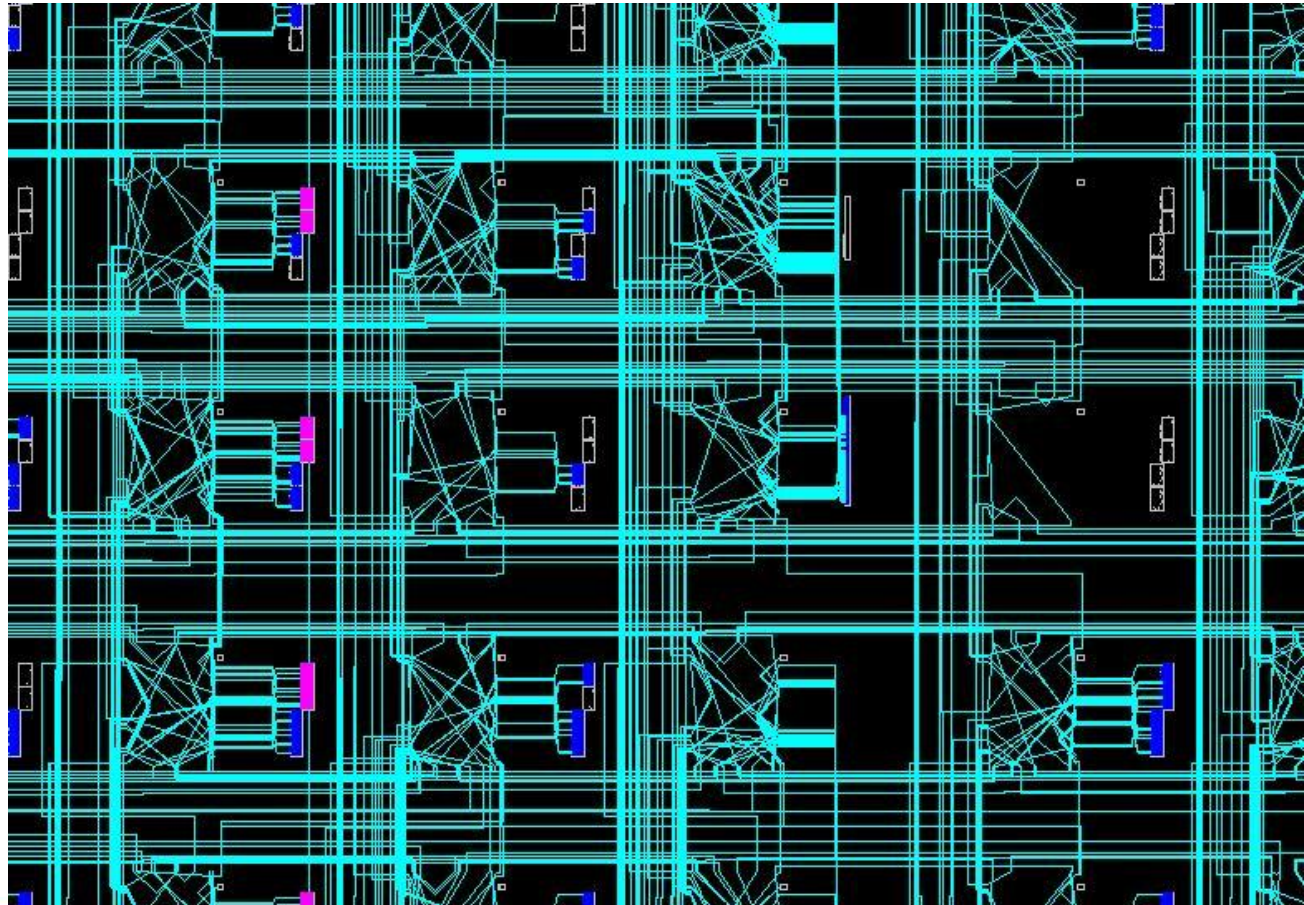


Hard to program
Hard to debug

Specific languages
Specific tools

Performances still
requires manual
tuning

EE skills required



State of the art debugging

Simplifying Xilinx and Altera FPGA Debug



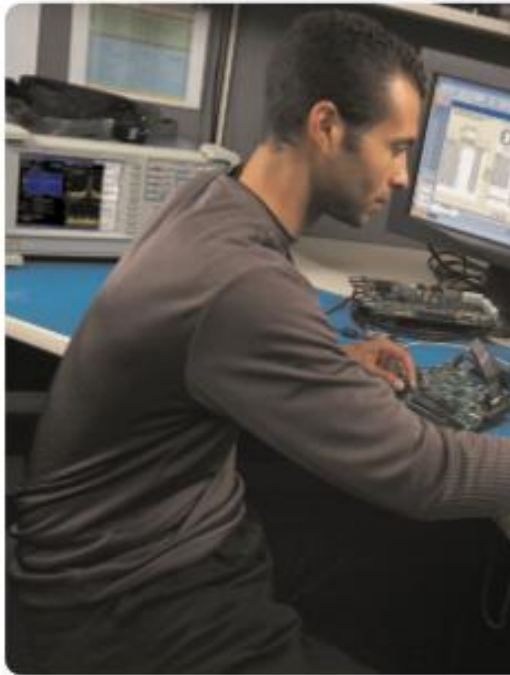
Huge challenge

Debug Your FPGA Design At Full Speed

Solutions such as FPGAView™ enable you to instantly move probe points within your Xilinx and Altera FPGAs without the need to recompile your design. Plus the ability to correlate internal FPGA signal activity to board-level signals can make the difference between hitting your schedule and missing your time-to-market window.

Touching the void

Simplifying Xilinx and Altera



Debug Your FPGA Design At Full Speed
Solutions such as FPGAView™ enable you to install and debug your FPGA designs without the need to recompile your design. This level of activity to board-level signals can make the difference in your time-to-market window.



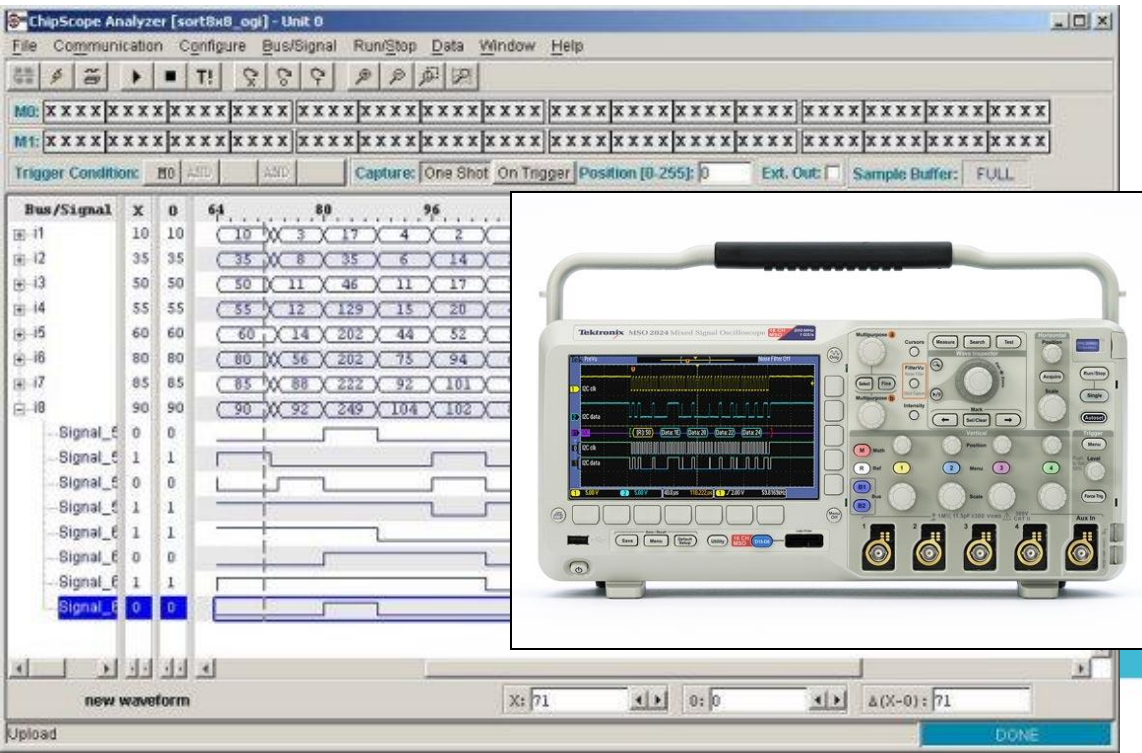
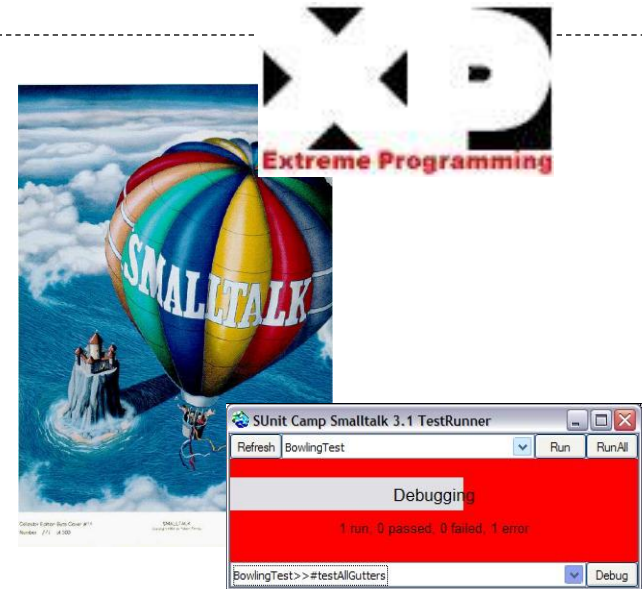
Huge challenge

Meet in the middle



Debug silver bullet

- Observability
- Controlability
- Abstract analysis
- Fast changes

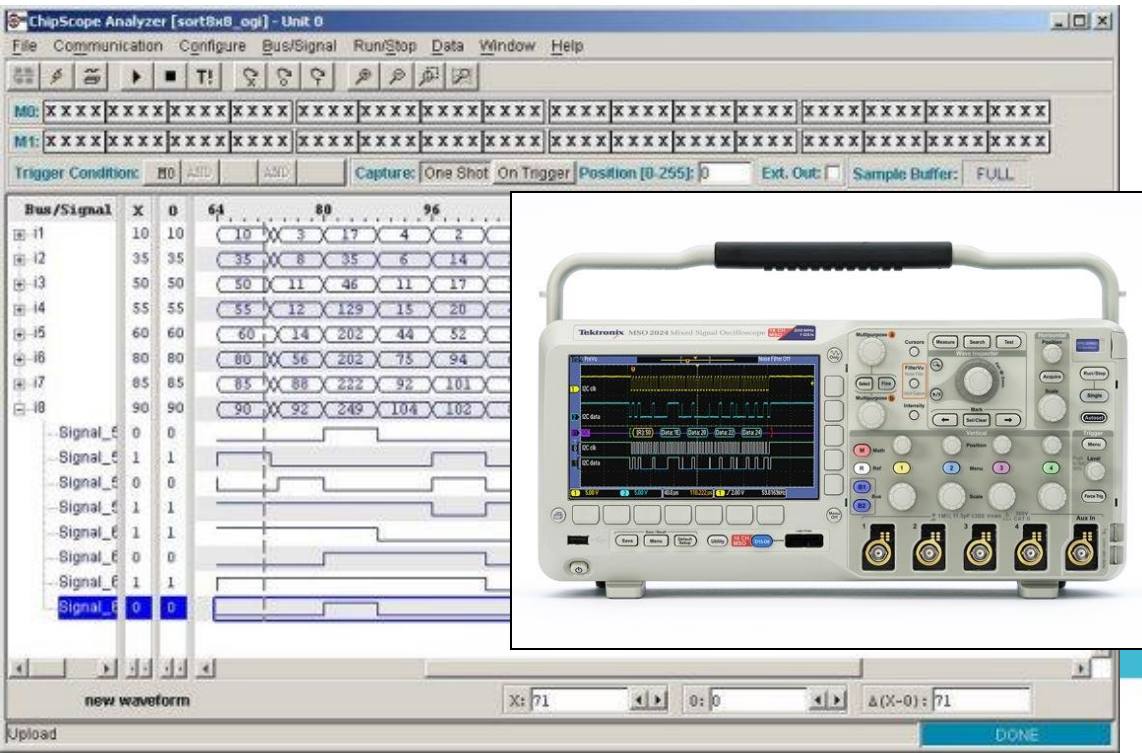
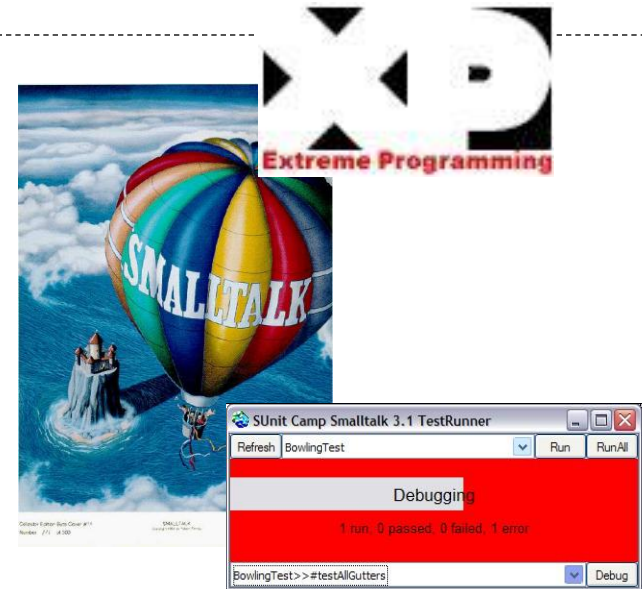


- Time to market
- Multiple runs
- Long cycles
- Simulation is not enough



Debug silver bullet

- Observability
- Controlability
- Abstract analysis
- Fast changes



- Time to market
 - Multiple runs
 - Long
 - Simplicity is not
- New challenges :
Threat or opportunity?**

Operate at-speed

Keep your speed-up alive



Multiple runs prohibit any over time penalty

Operate in-situ

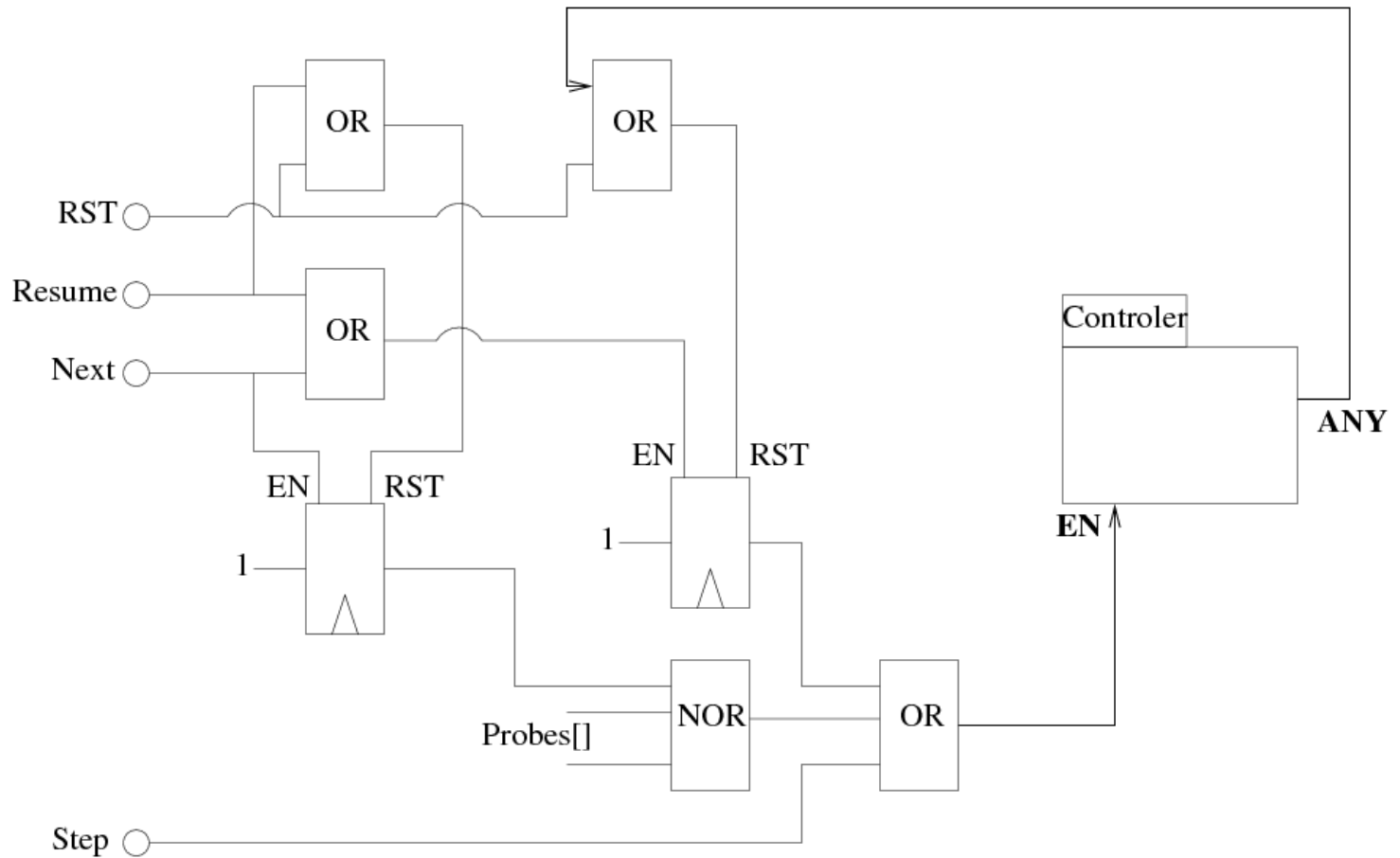
Because only HW brings speed-up



Debug requires extra circuitry



Debugging facilities are circuits



Debugging facilities are circuits

Break in UIBiniou>>openCDFG at 32

Stack Method Edit Execute Correct

UIBiniou>>openCDFG
MenuBarButtonController>>dispatchMenuSymbol:
MenuBarButtonController>>dispatchMenuSelection:
optimized [] in MenuBarButtonController>>menuBarActionFo
BlockClosure>>ensure:
MenuBarButtonController>>menuBarActionForMenu:
MenuBarButtonController>>menuBarAction
MenuBarButtonController>>redButtonPressedEvent:
RedButtonPressedEvent>>dispatchTo:
MenuBarButtonController(Controller)>>handleEvent:
EventDispatcher>>dispatch.to:
EventDispatcher>>dispatchEvent:

RST

Resume

Next

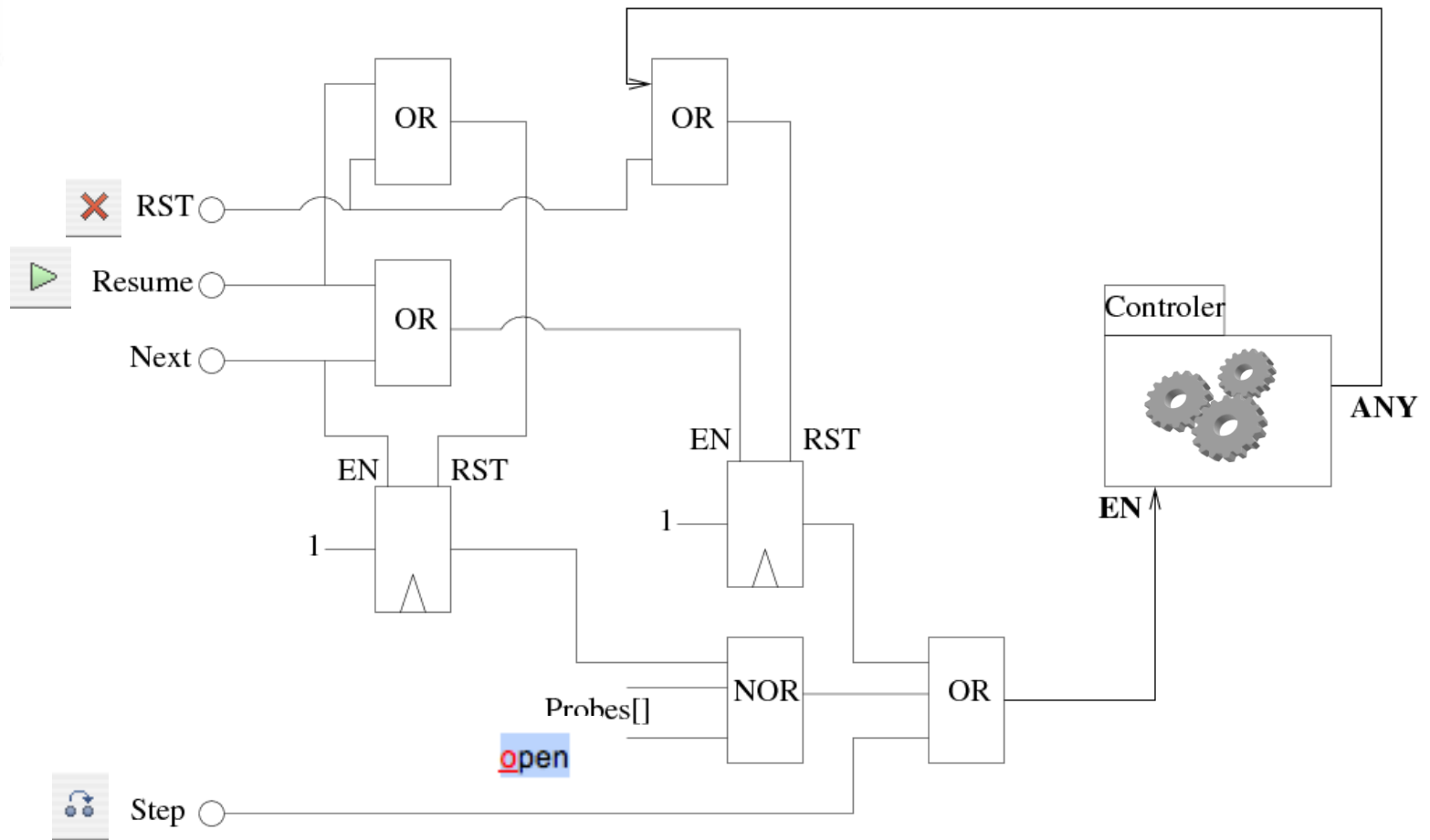
Step

self cdfgSynthesis open

self
depends
builder
uiSession
eventHandlers
breakPointsList
typeList

ANY

Debugging facilities are circuits



Observe and monitor



Take decision

You cannot watch everything

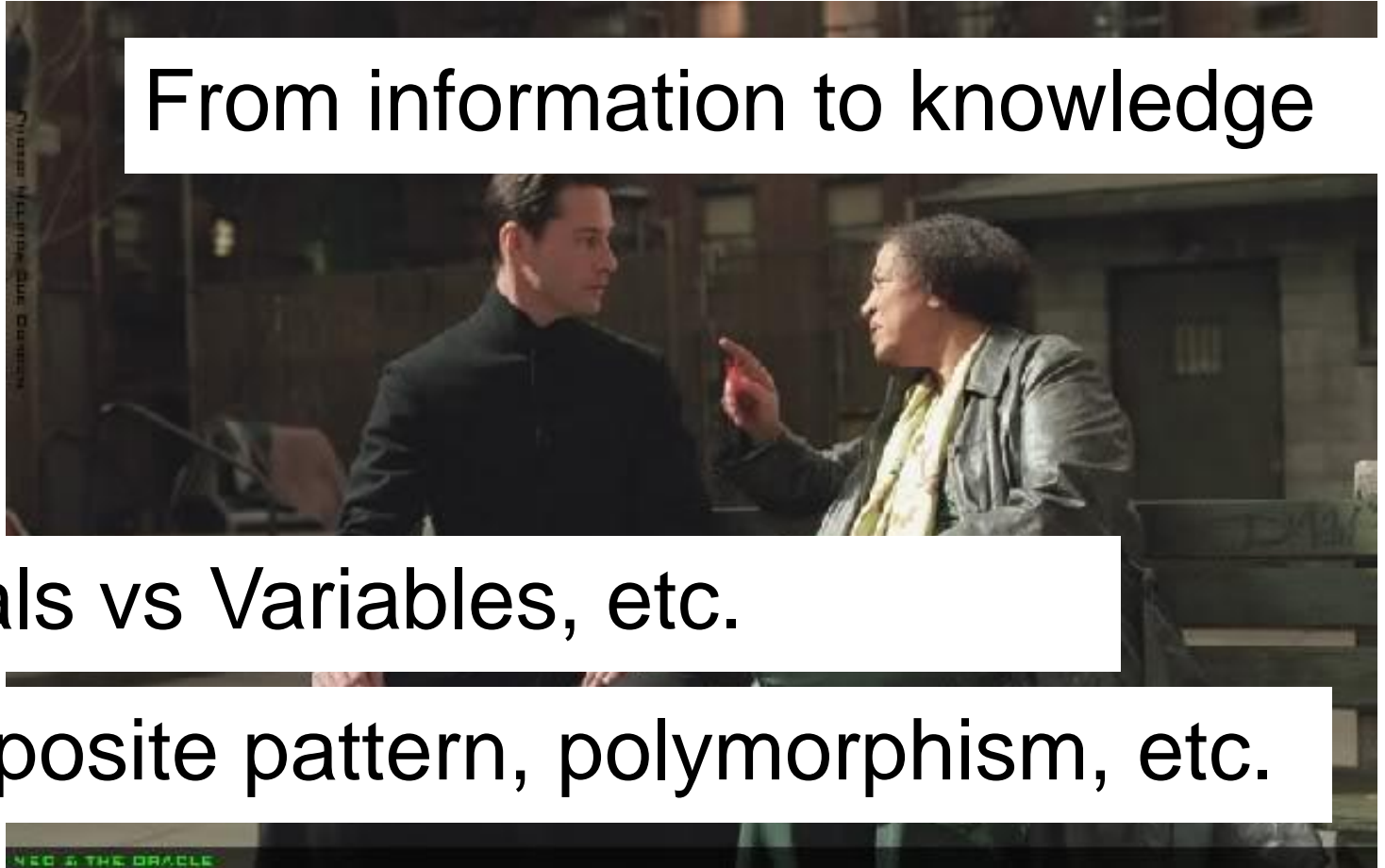


Abstract analysis: semantic needed

From information to knowledge

Signals vs Variables, etc.

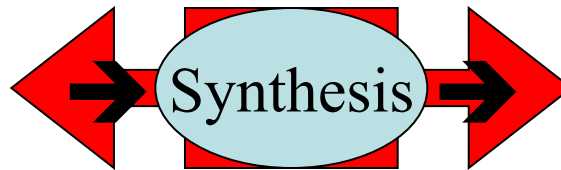
Composite pattern, polymorphism, etc.



From a technical point of view



Software abstraction



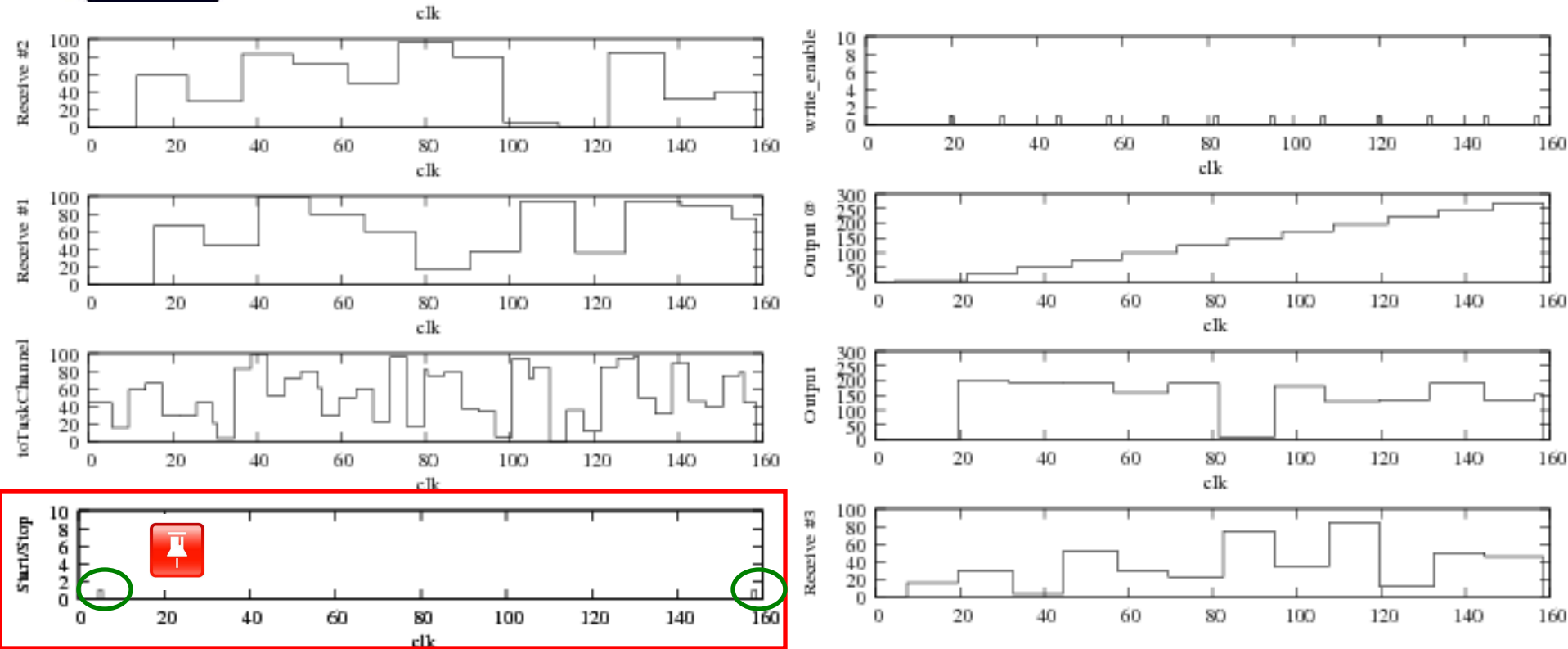
Implementation in hardware

From D. Picard's ESUG 2009 talk



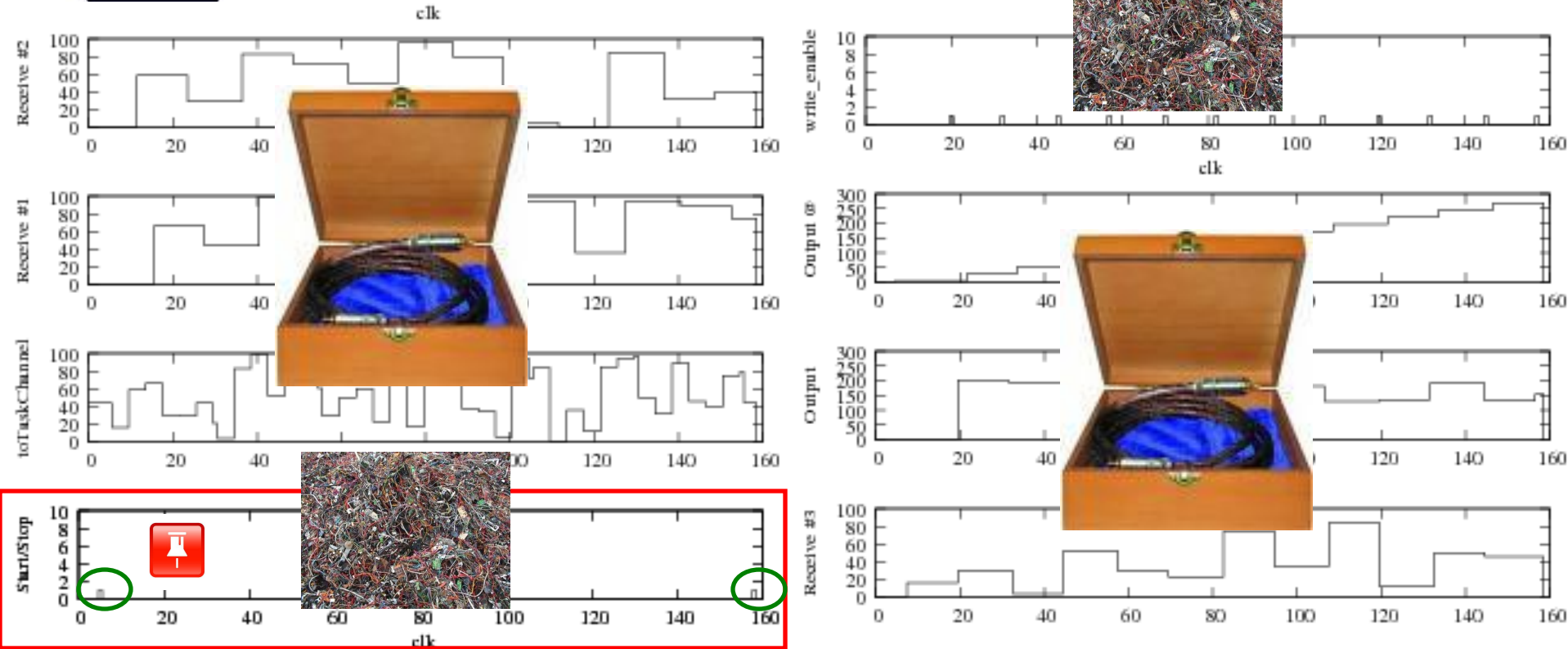
Full observability isn't scalable

Focus on POI

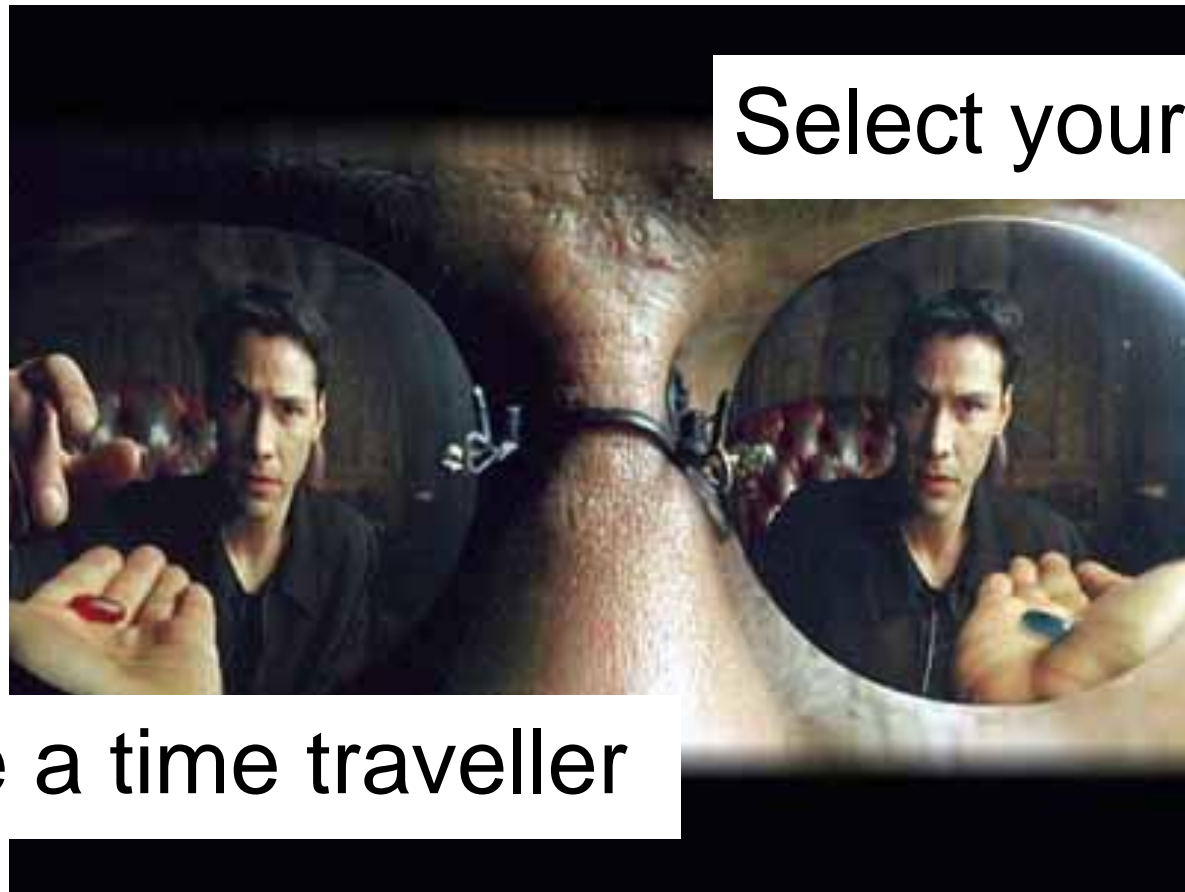


Full observability isn't scalable

Focus on POI



Take control



Become a time traveller

Smalltalk debugger

- Just fit approach
- Run code and catch exception
- Code hot replacement, variable update, etc.
- Step on or resume execution
- Possible rollback

- Multiple runs
- Breakpoints update, earlier exception
- Same conceptual behavior

Become a time traveller

Dodge bullets



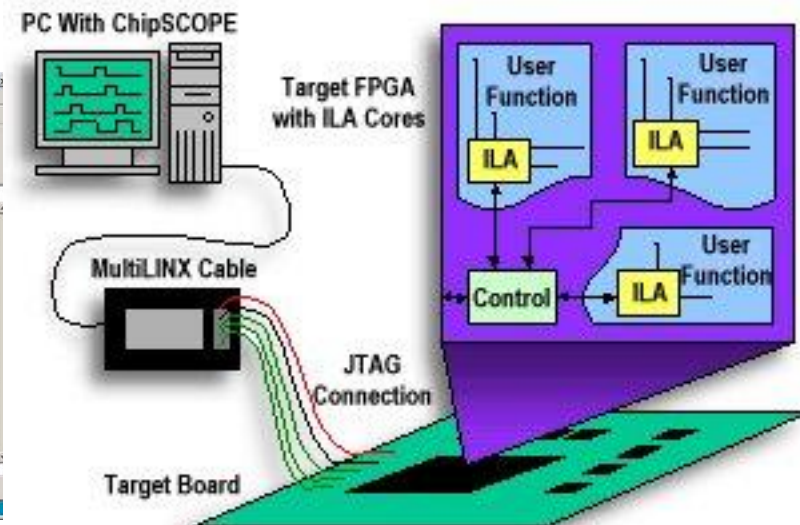
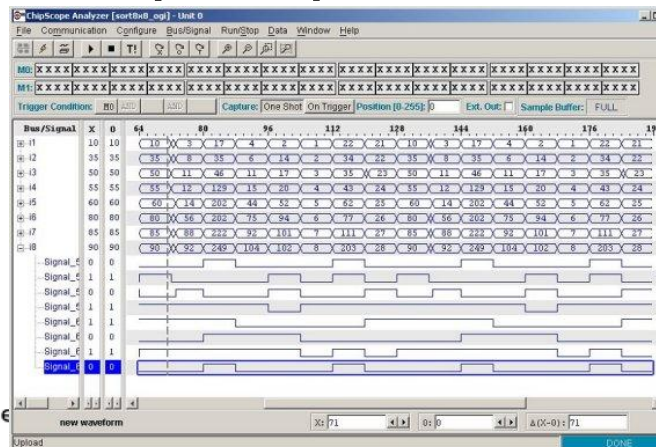
Once the time has stopped

... just operate !



Conditional probes

- Conditional probes offer the controlability that lacks in commercial tools
- Observability can be gained at the cost of adding some variable look-up wires
- ... But also using vendor's tool such as Chipscope



Red Pill

Decomposition

Probes

Application

Application
(Generated)

Probes
inspector

Inserted probe

Control panel

Active probes

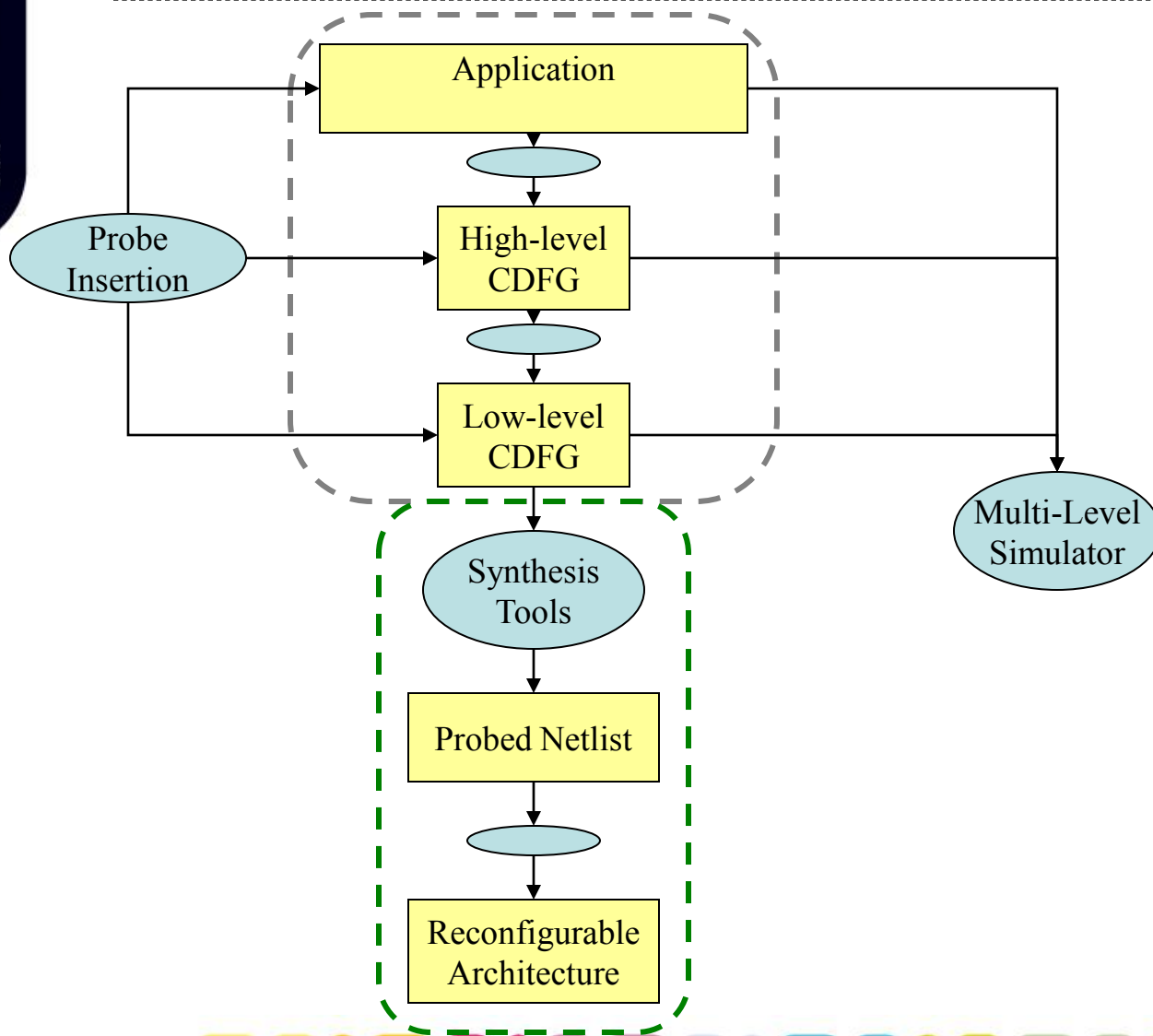
The screenshot shows the Red Pill GUI with the following components:

- Menu:** Compile, Synthesize, Simulate
- Tasks Panel:** Tasks, Connectors, Topology, UnrolledCode
- Code Editor:**

```
process fiab(chanin char cmf, chanout char fib)
{
  char t_1_6;
  char t_1_4;
  t_1_2;
  t_1_5;
  char t_1_1;

  cmf ? t_1_1;
  cmf ? t_1_2;
  cmf ? t_1_3;
}
```
- Probes Inspector:** t_1_1, t_1_2 (highlighted), t_1_3, t_1_4, t_1_5, t_1_6
- Control Panel:** Play, Stop, Add, Search, Refresh icons; Clock: 39
- Output:** test

RedPill Flow extends Biniou (HLS)



Biniou

Abstract analysis

Encapsulate circuit modules as smalltalk blocks

- Enables soft and hard objects to communicate
- Delivers the power of Sunit to hardware

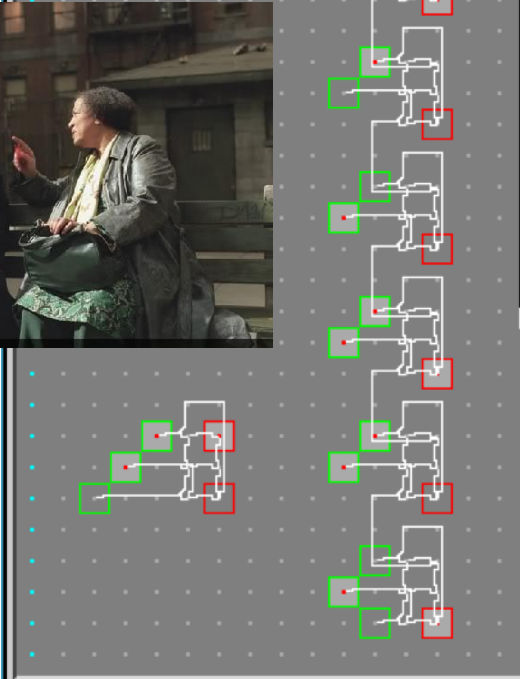
Abstract analysis (example)



File Modules Routing Device Expe

xc6216PC84-2 selection size : cell 1@10

Converting



Converting

a Hardware Context

```

block
module
position
-- status --
status
interactive
reactive
-- inputs --
in0
in1
-- outputs --
out3

```

" Hardware Execution"
self value: 15 value:22*2 ["#(37)"]

"Software Execution"

| block |
block := [a :b] Array with:(a + b \ 128).

block value: 15 value:22 "#(37)"

a Hardware Context

```

block
module
position
-- status --
status
interactive
reactive
-- inputs --
in0
in1
in2
-- outputs --
out4

```

" Hardware Execution"
self value: 0 value:1 value:1 "#(2)"]

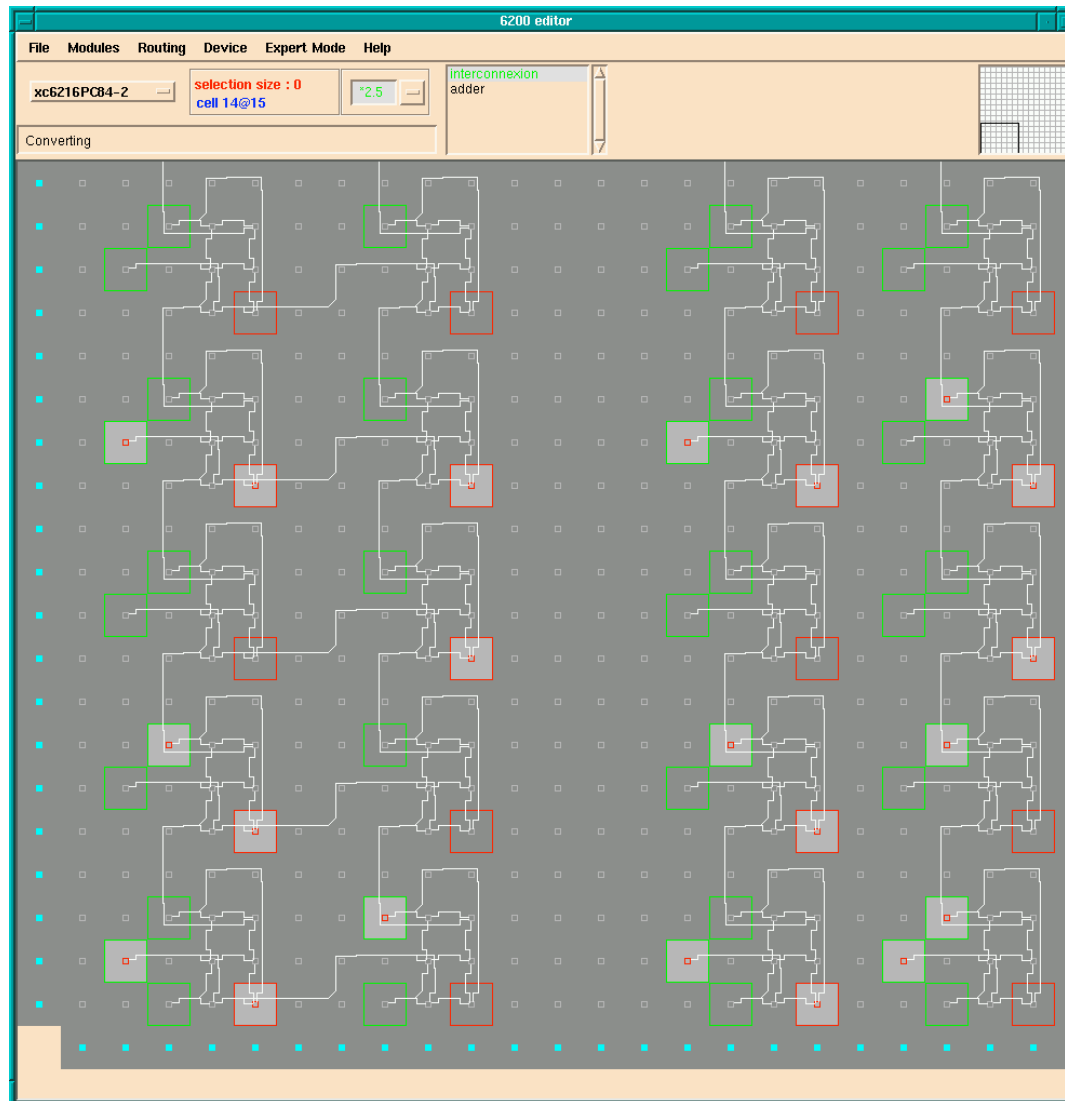
"Software Execution"

| block [
block := [x :y :z] [c s]
c := x + y + z > 1 ifTrue:[1]
iffFalse:[0].
s := x + y + z \ 2.
^ Array with:(c*2 +s)].

block value: 0 value:1 value:1 "#(2)"]



Characterization tests & SUnit



How Many matrix?



The screenshot displays the 'Architecture Designer' software interface. The main window has a menu bar with 'File', 'Operations', 'Options', and 'Help'. Below the menu is a 'Title' field containing 'LPPGACell2' and buttons for 'New' and 'Delete'. To the right is an 'Options' section with a 'Category' field set to 'lppga'.

The 'Contents' area shows a hierarchical tree structure with a context menu open over it. The menu items include: 'edition', '(', 'Architecture', 'Command', 'New Window', 'Done', 'copy', 'cut', 'paste', 'format', 'file', 'find', 'find and replace', 'again', 'undo', and 'hardCopy'. The tree structure includes nodes like '(ARRAY', '(DOMAIN 1 1 40 40) "END of DOMAIN"', '(INPUTS', '(OUTPUT', 'REPRE', and '(DEFAULTCOLOR very arbitrary)'. Below the tree is a 'Building' section with buttons for 'compile', 'internal', 'tree view', 'build', 'produce', 'UGI', and 'open'.

Three smaller windows are overlaid on the right side of the main window:

- The top window, titled 'LPPGArray Editor', shows a grid of colored lines (red, green, blue) representing a matrix structure.
- The middle window, also titled 'LPPGArray Editor', shows a grid of small square cells, some containing icons, representing a different view of the matrix.
- The bottom window, titled 'M2KTopLevel Editor', shows a grid of small square cells, similar to the middle window but with a different layout.

Reboot the Matrix



The collage features several key elements:

- Virtual layer / Physical Layer:** A diagram showing a grid of orange blocks (Virtual layer) and green blocks (Physical Layer) connected by lines.
- Architecture Designer:** A screenshot of a software interface with a menu containing:
 - New Window
 - Done
 - (INPUTS)
 - (OUTPUT)
 - REPRESE
 - (DEFAULTCOLOR vel...
- MicroBlaze and CGRA:** A large central image of a circuit board layout with two red boxes highlighting the 'MicroBlaze' and 'CGRA' components.
- Other Screenshots:** Various other screenshots showing circuit diagrams, a grid-based layout, and software windows.

Will you take the red pill?

- HLS (Biniou) offers a path from HL languages to circuits
- Vendors tools offer observability
- Red Pill offers controlability
- Object encapsulation offers abstract analysis and polymorphism.

Smalltalk debug definitively lives in the Matrix

Thank you for your attention

