

Real-Time Embedded Systems

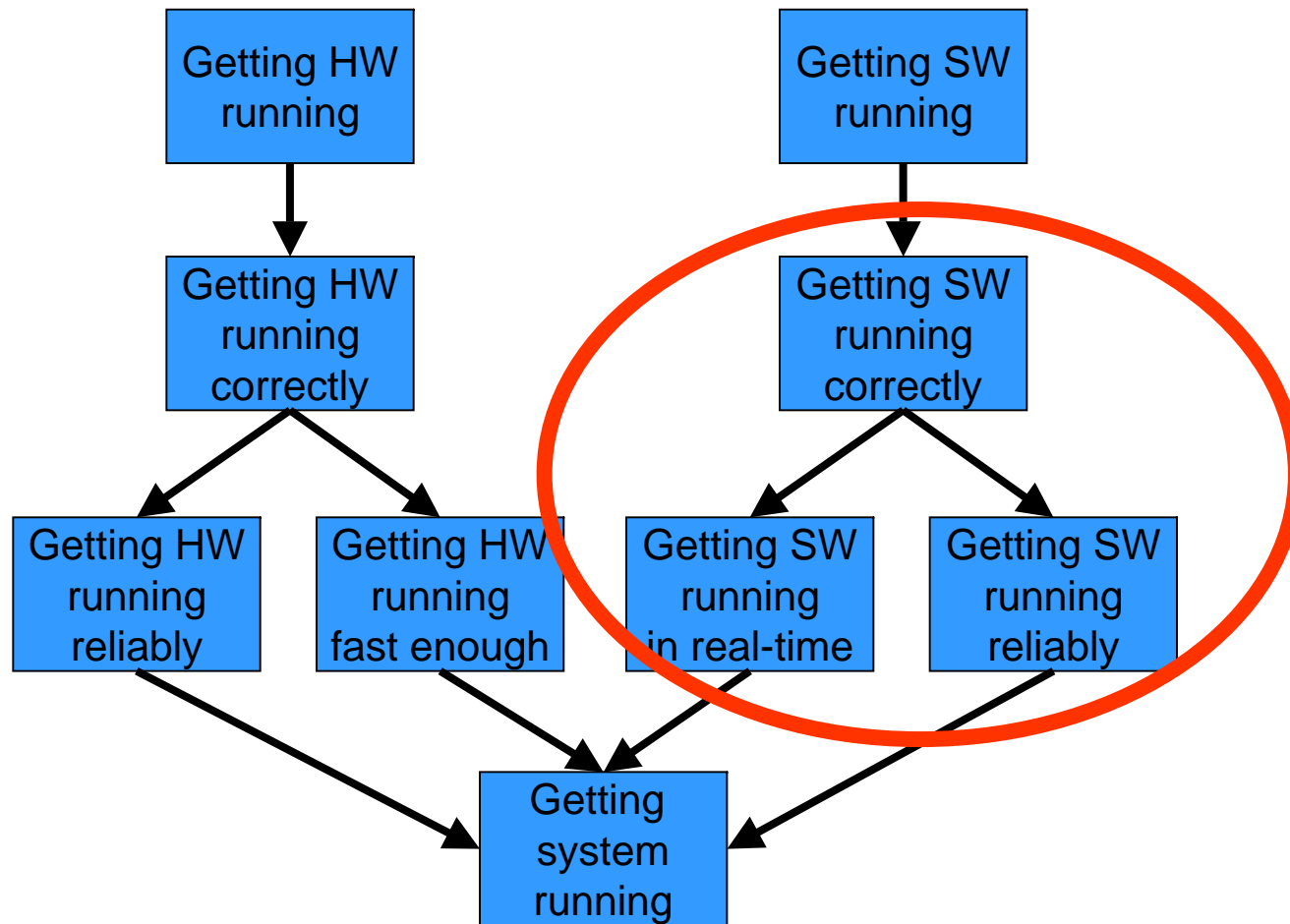
CpE-450 Spring 06

Class 9

Bruce McNair

bmcnair@stevens.edu

Correctness, Reliability, and Real-time Performance



Real-time and Correct Operation

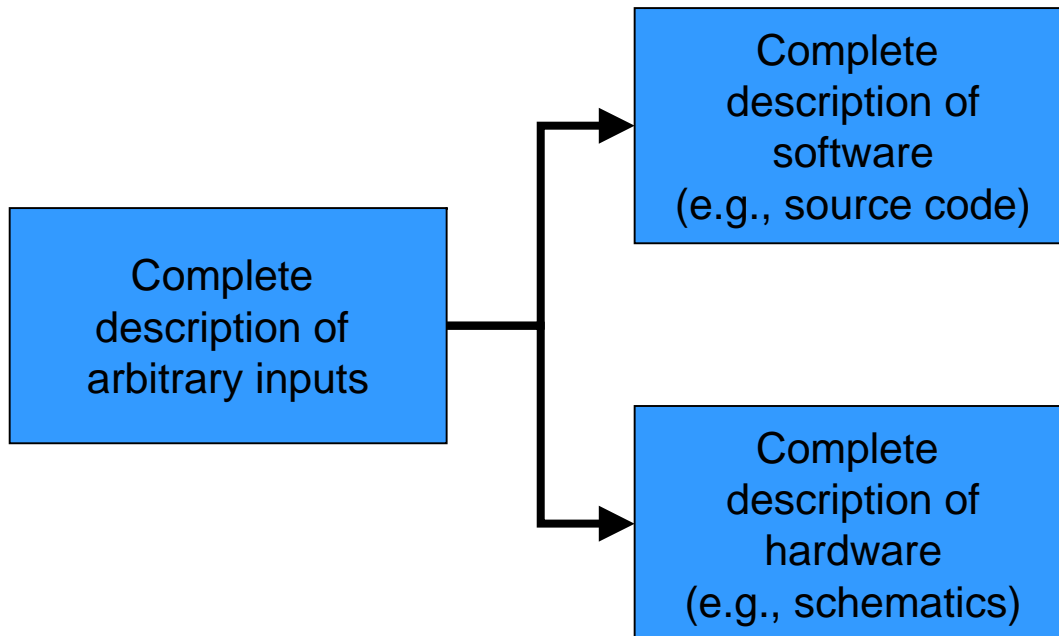
- Consider this system:

Complete
description of
software
(e.g., source code)

Complete
description of
hardware
(e.g., schematics)

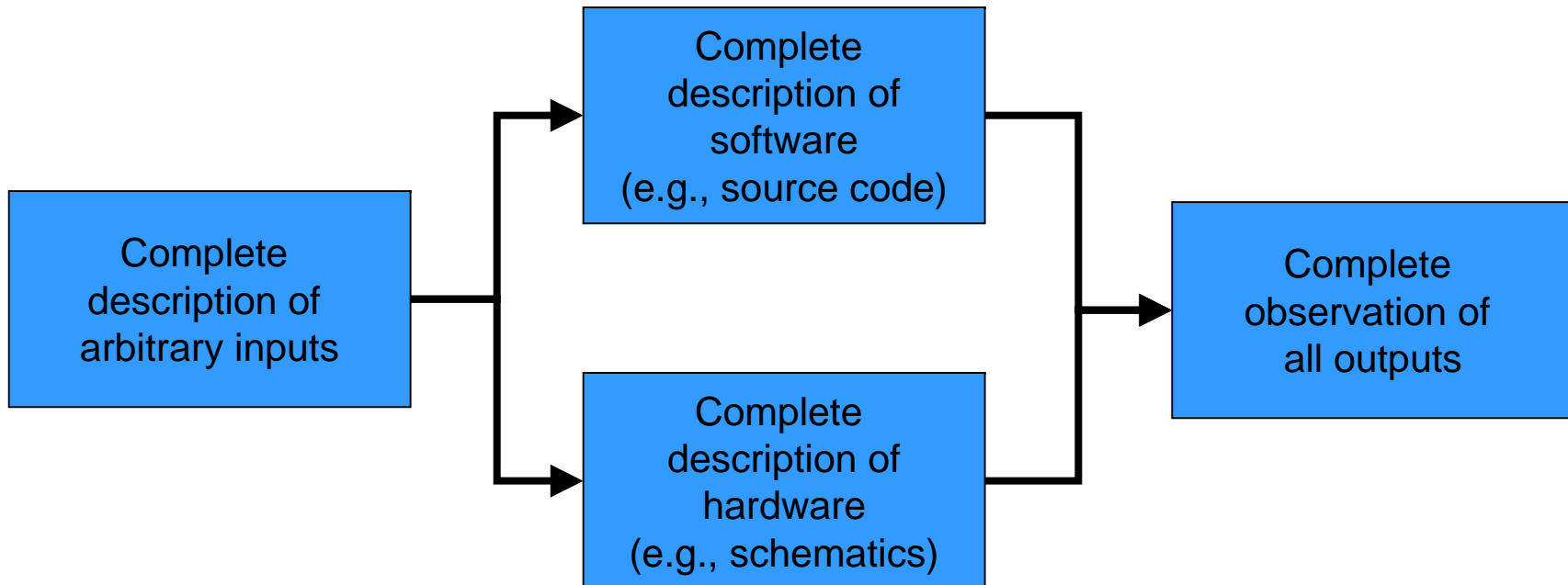
Real-time and Correct Operation

- Consider this system:



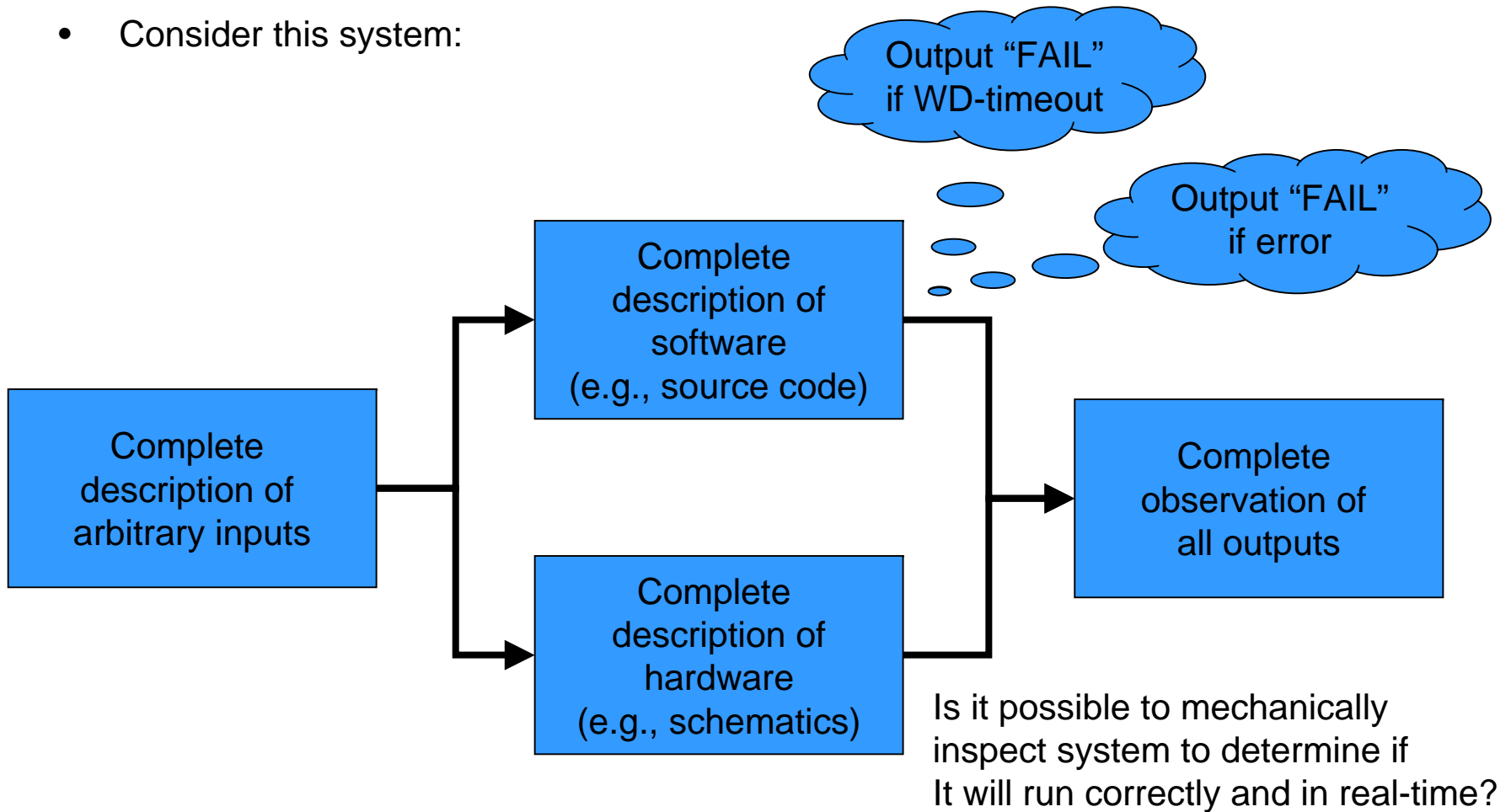
Real-time and Correct Operation

- Consider this system:



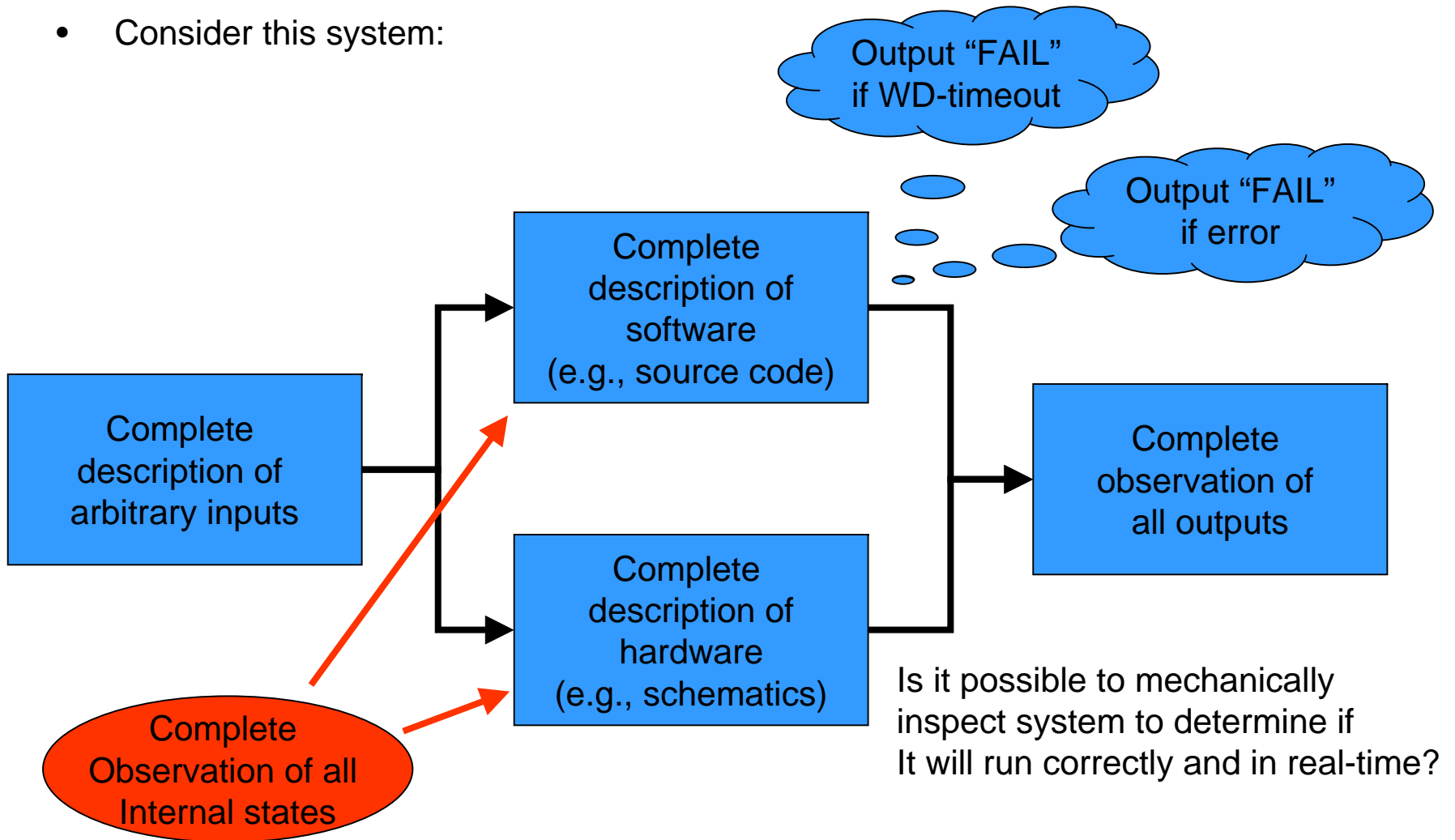
Real-time and Correct Operation

- Consider this system:



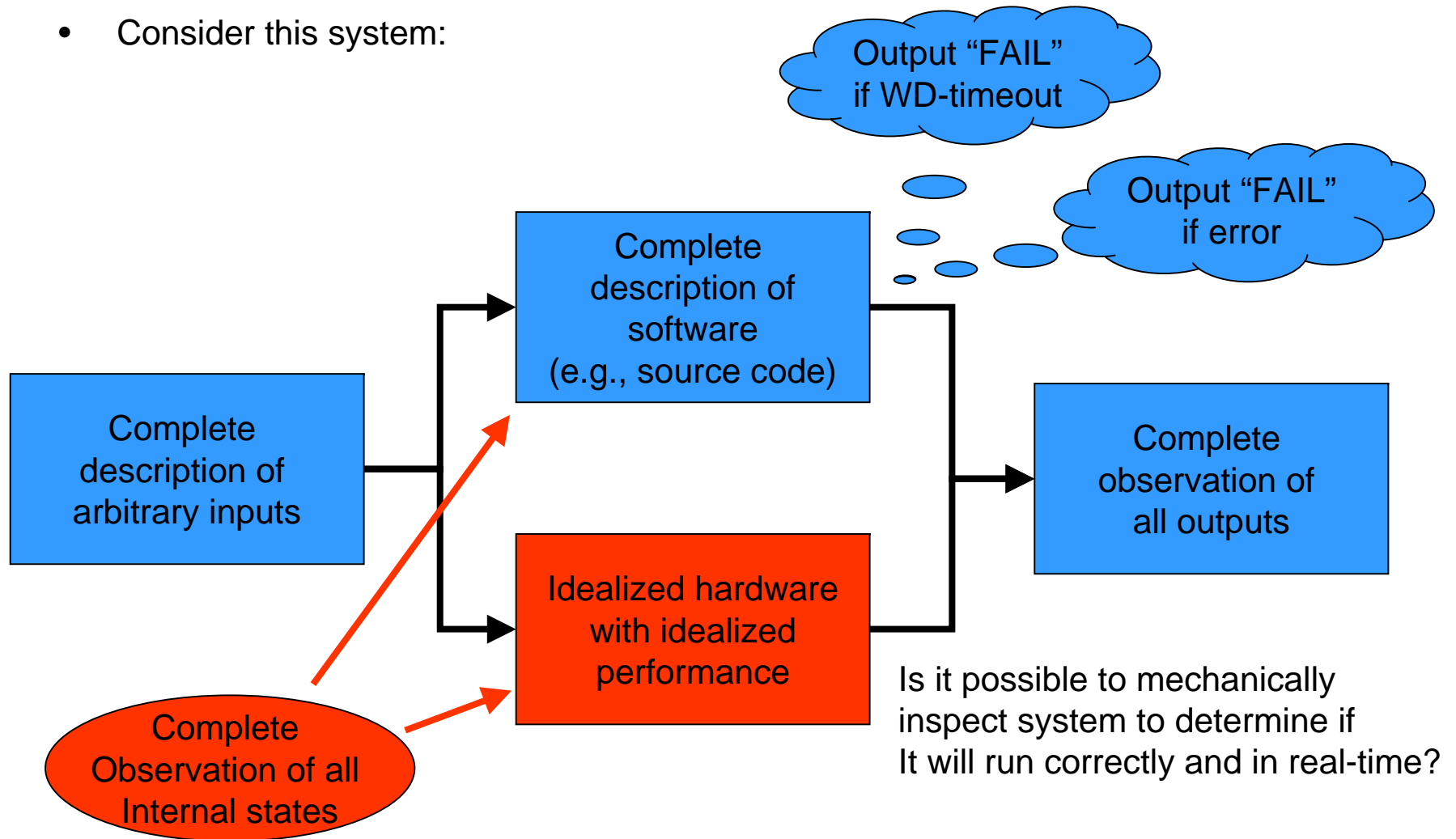
Real-time and Correct Operation

- Consider this system:



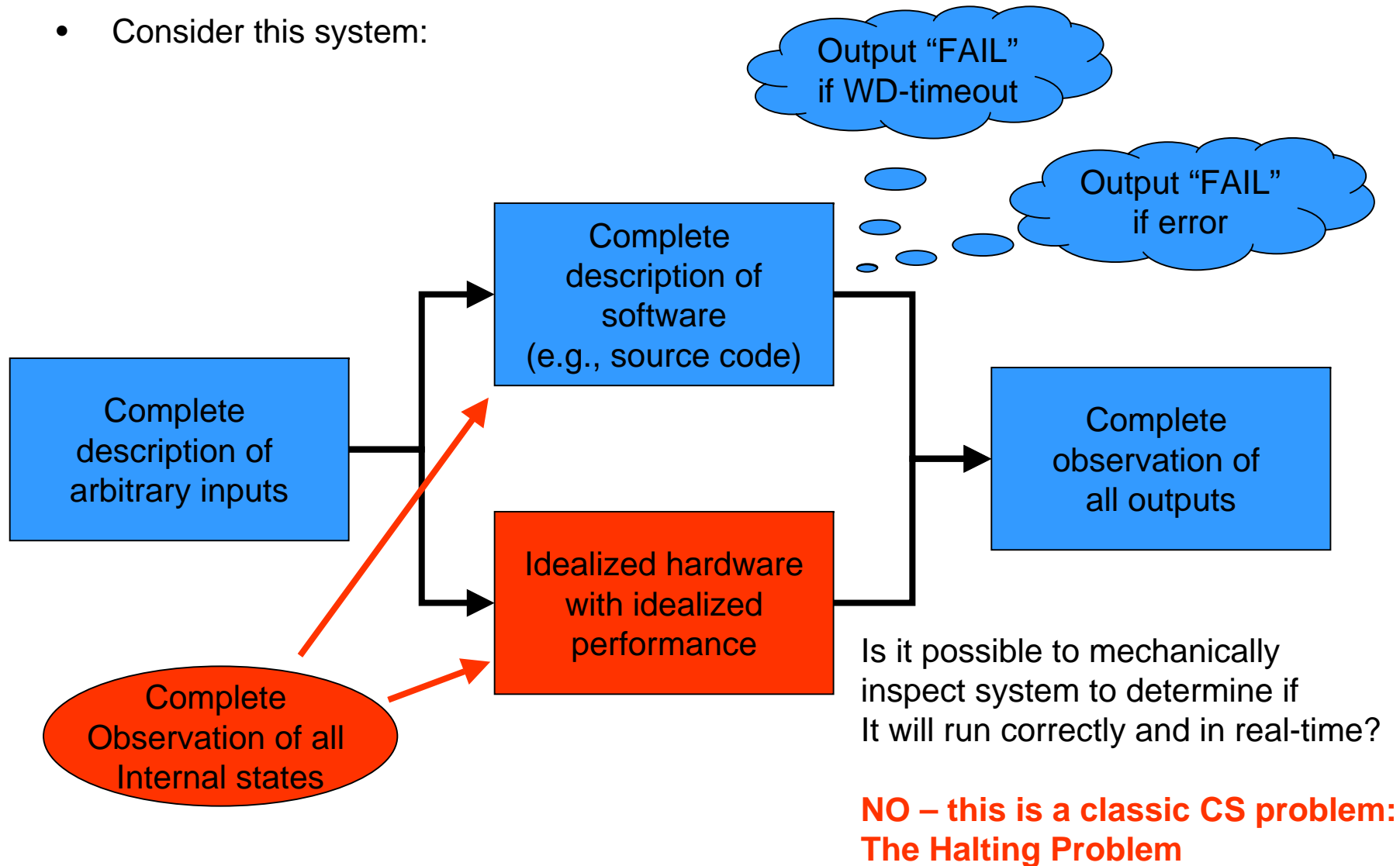
Real-time and Correct Operation

- Consider this system:

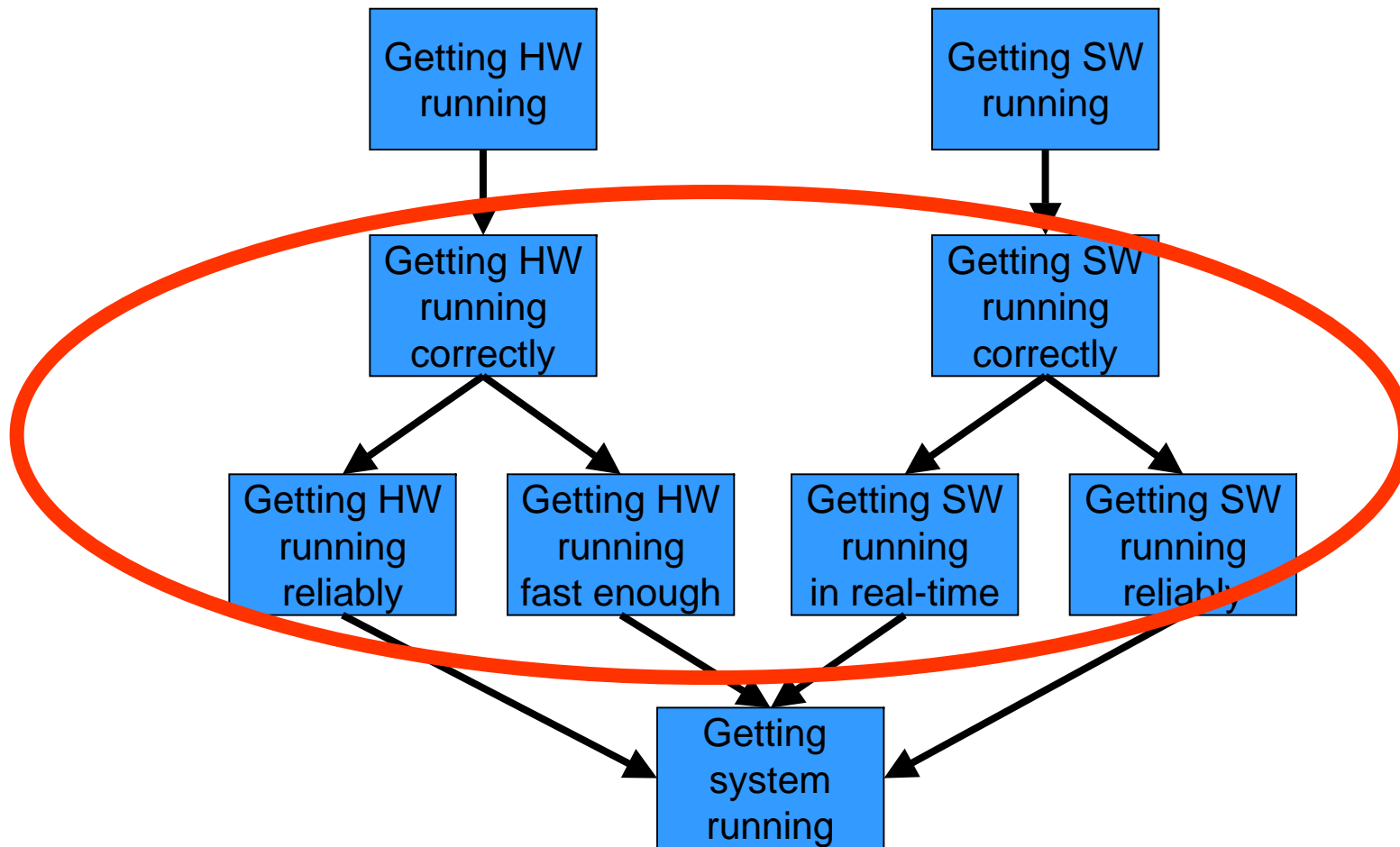


Real-time and Correct Operation

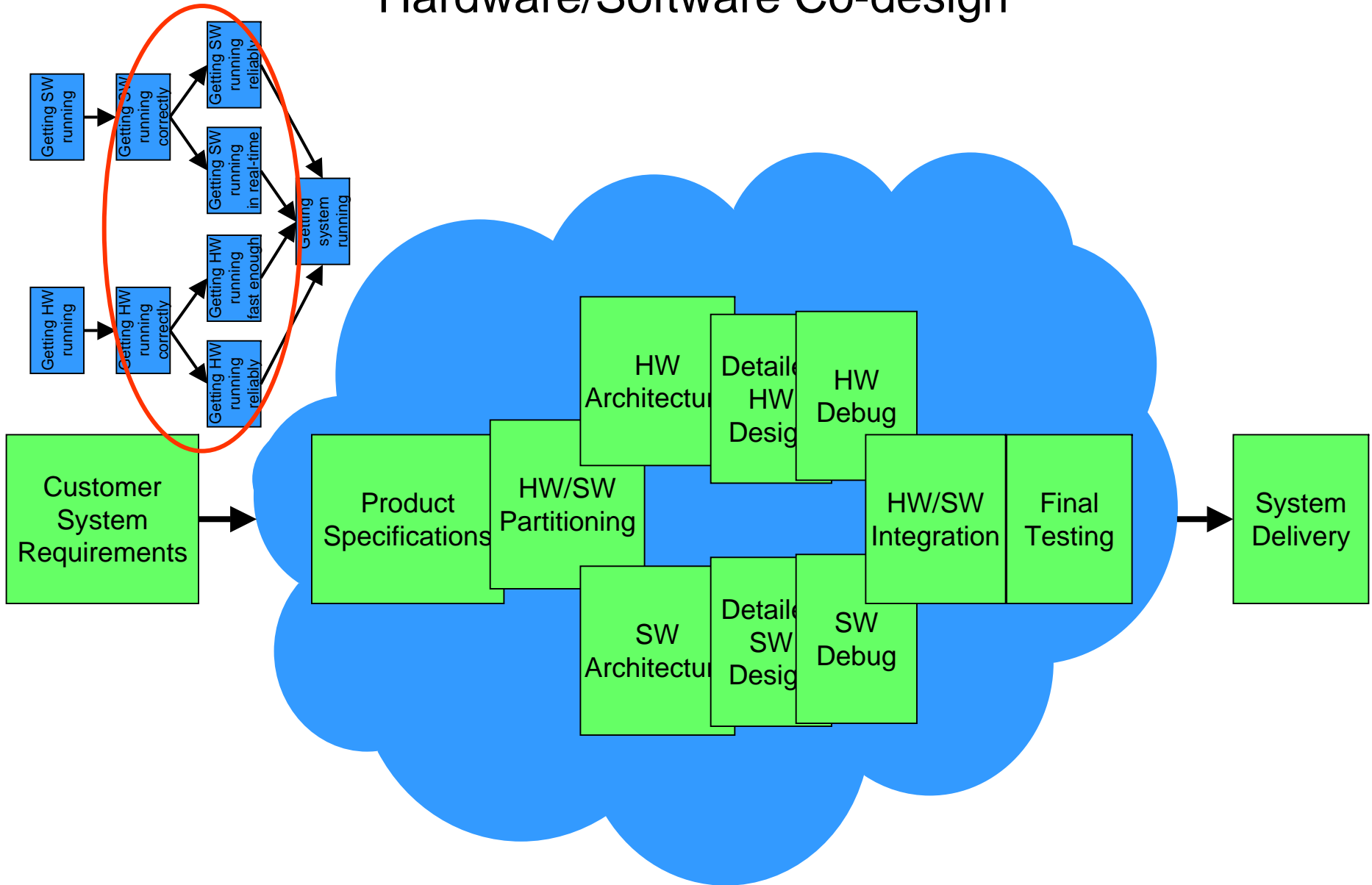
- Consider this system:



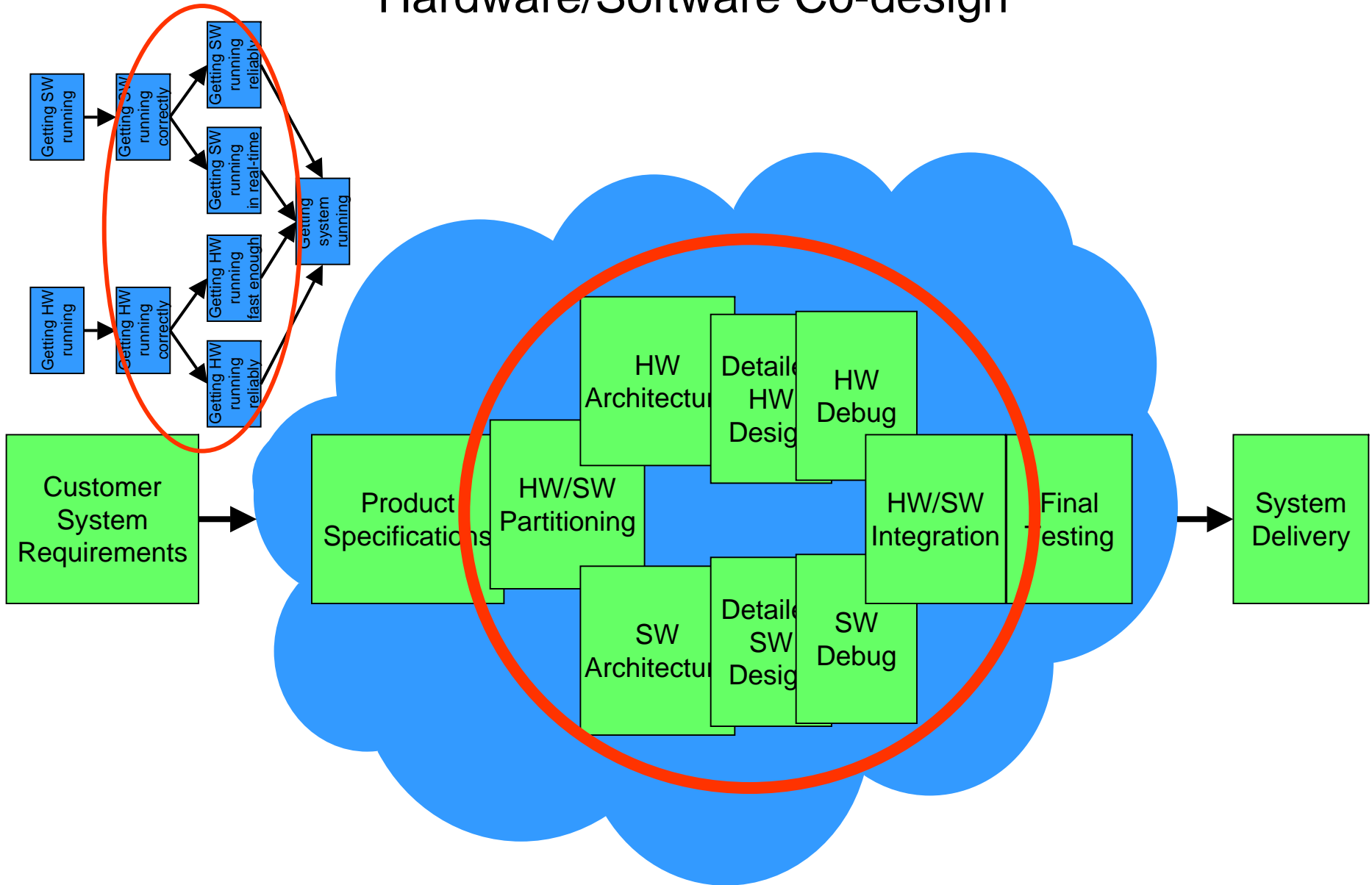
Hardware/Software Co-design



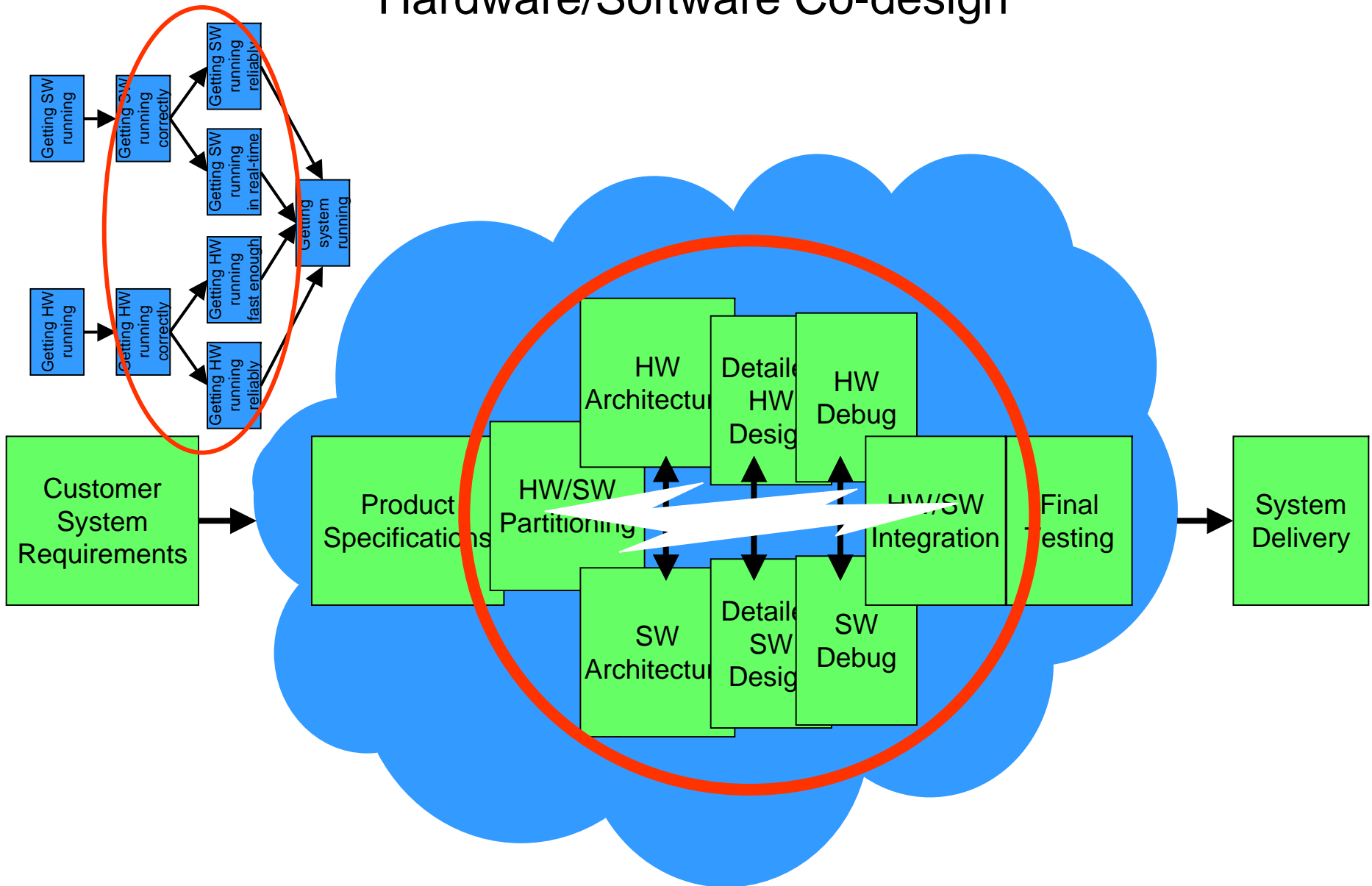
Hardware/Software Co-design



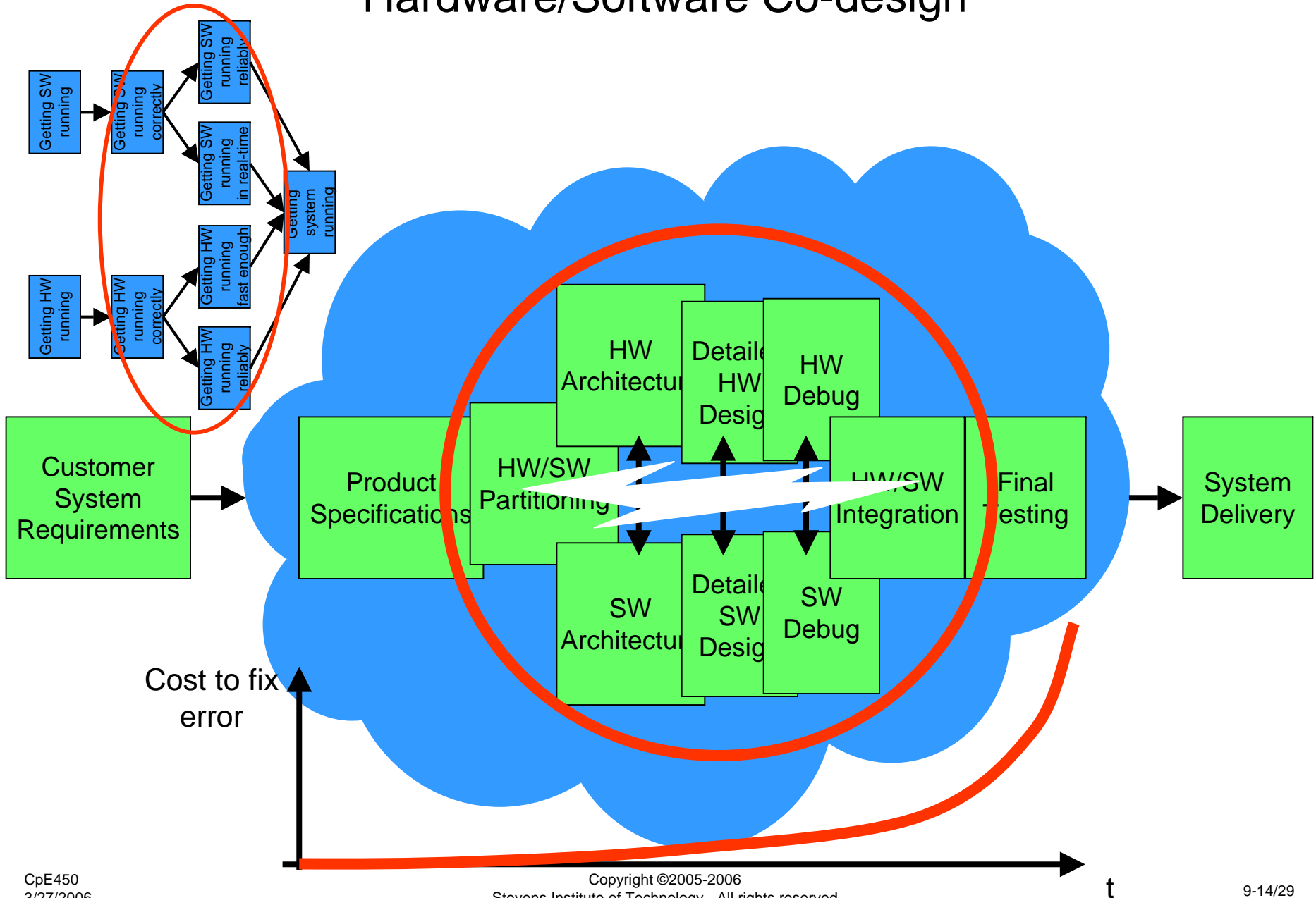
Hardware/Software Co-design



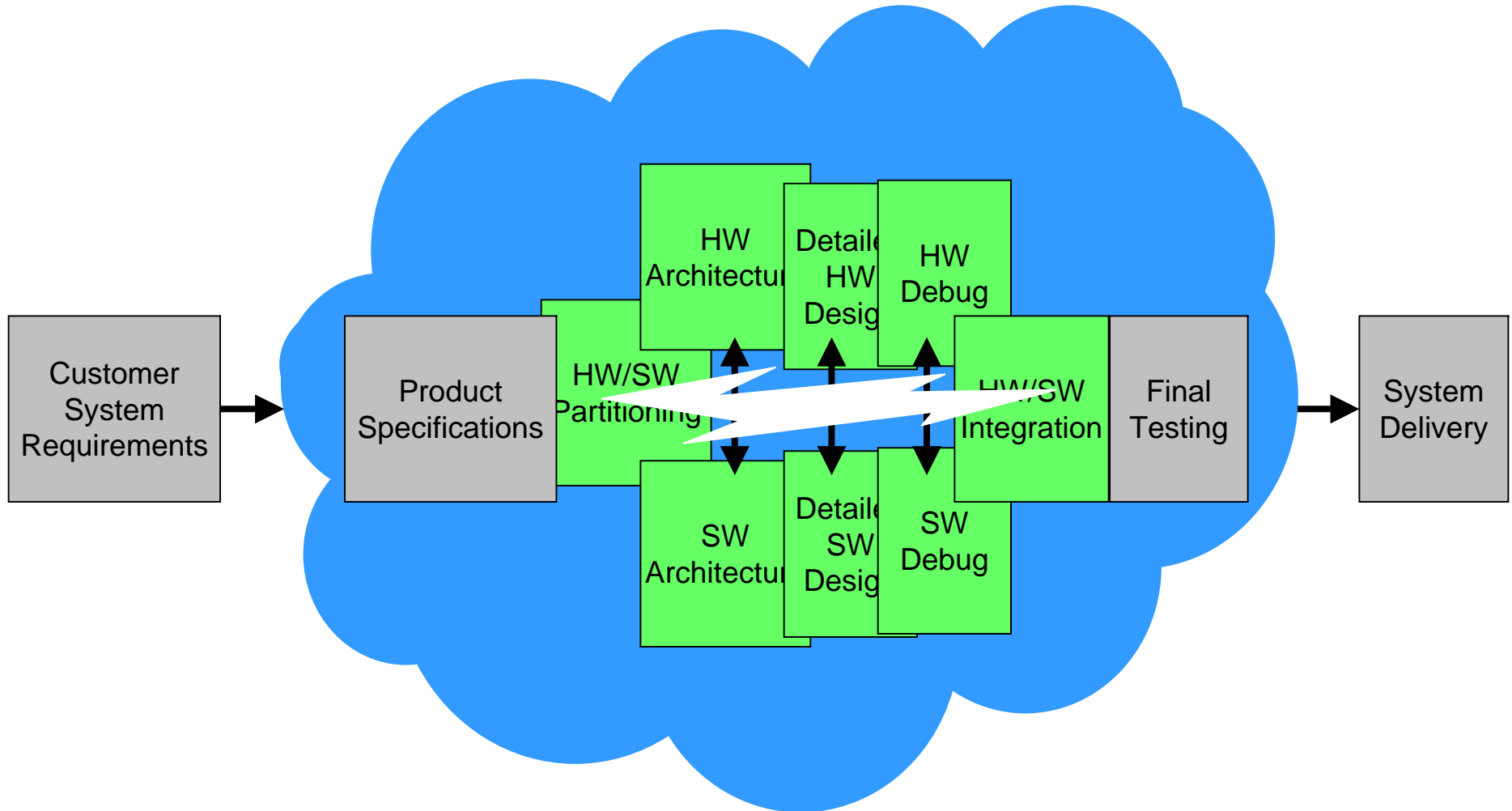
Hardware/Software Co-design



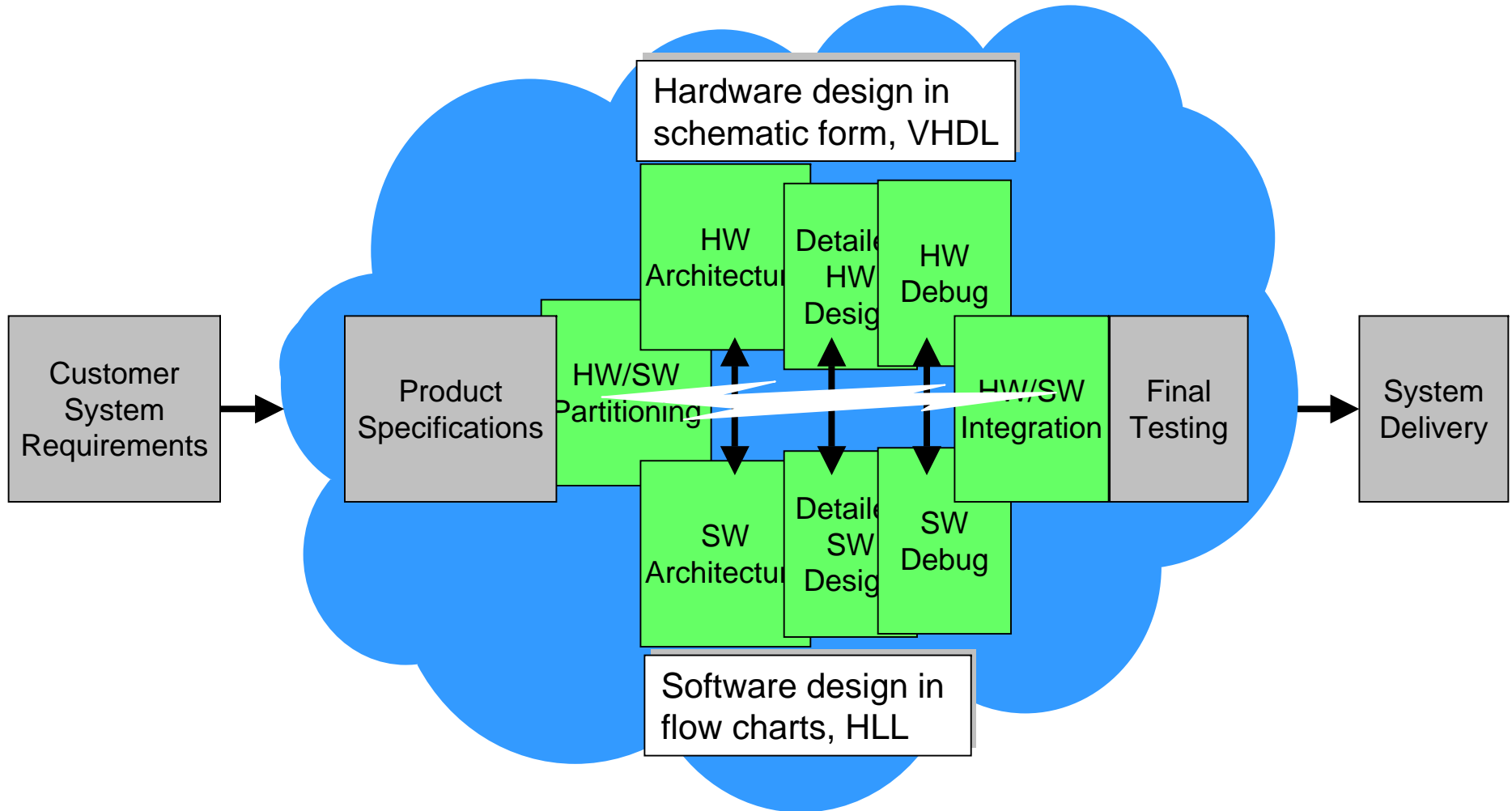
Hardware/Software Co-design



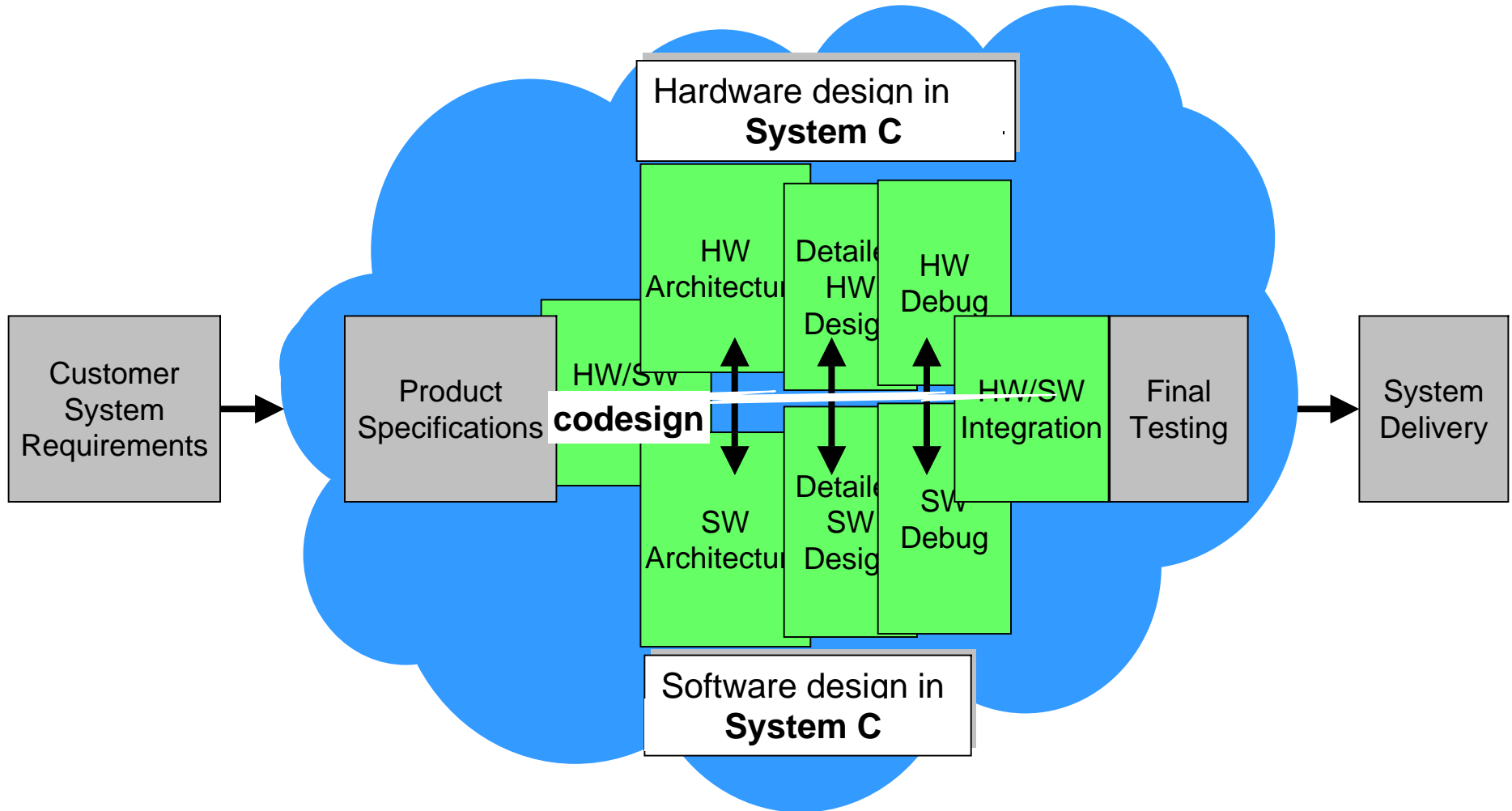
Addressing the Hardware/Software Co-design Issue



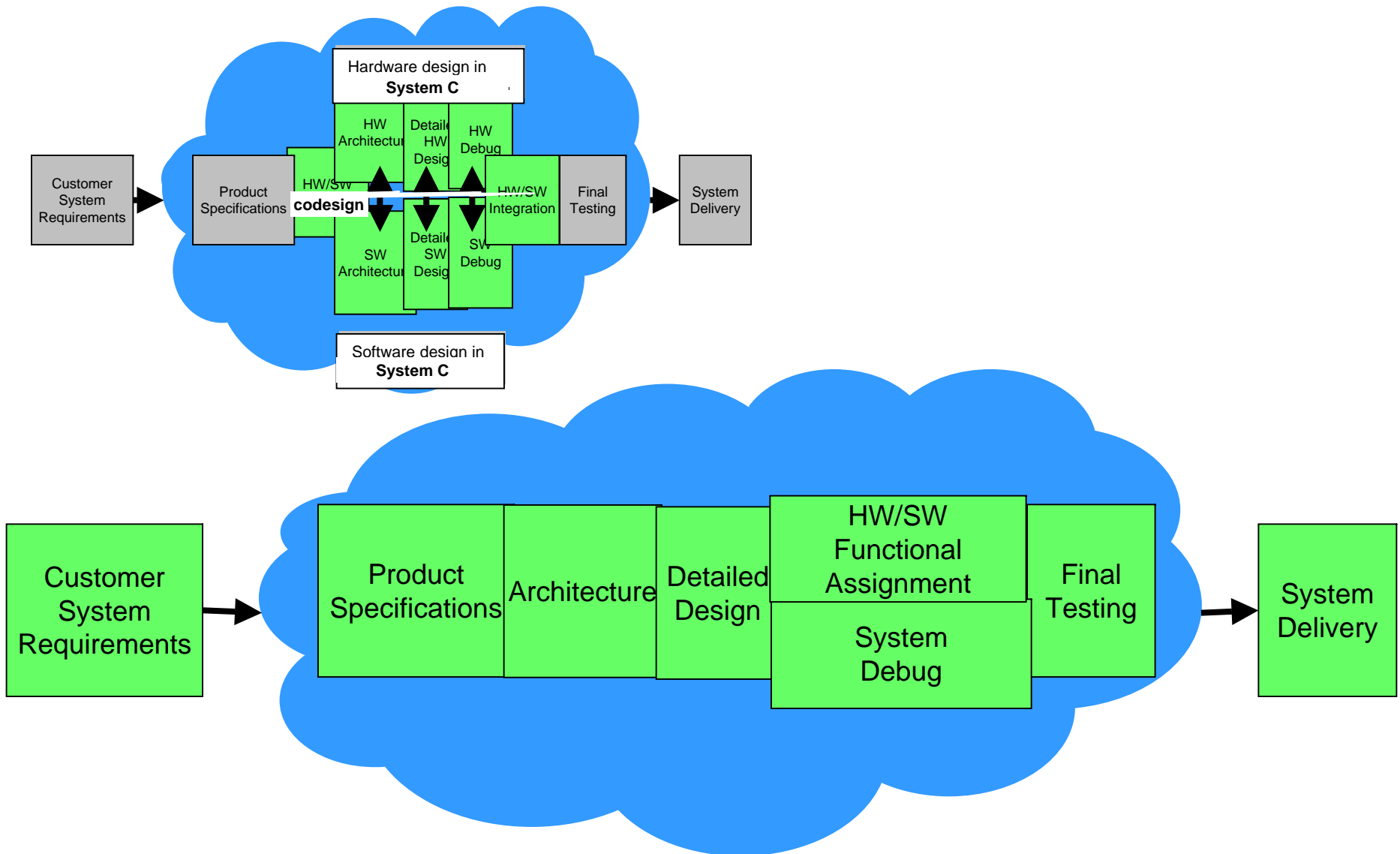
Addressing the Hardware/Software Co-design Issue



Addressing the Hardware/Software Co-design Issue

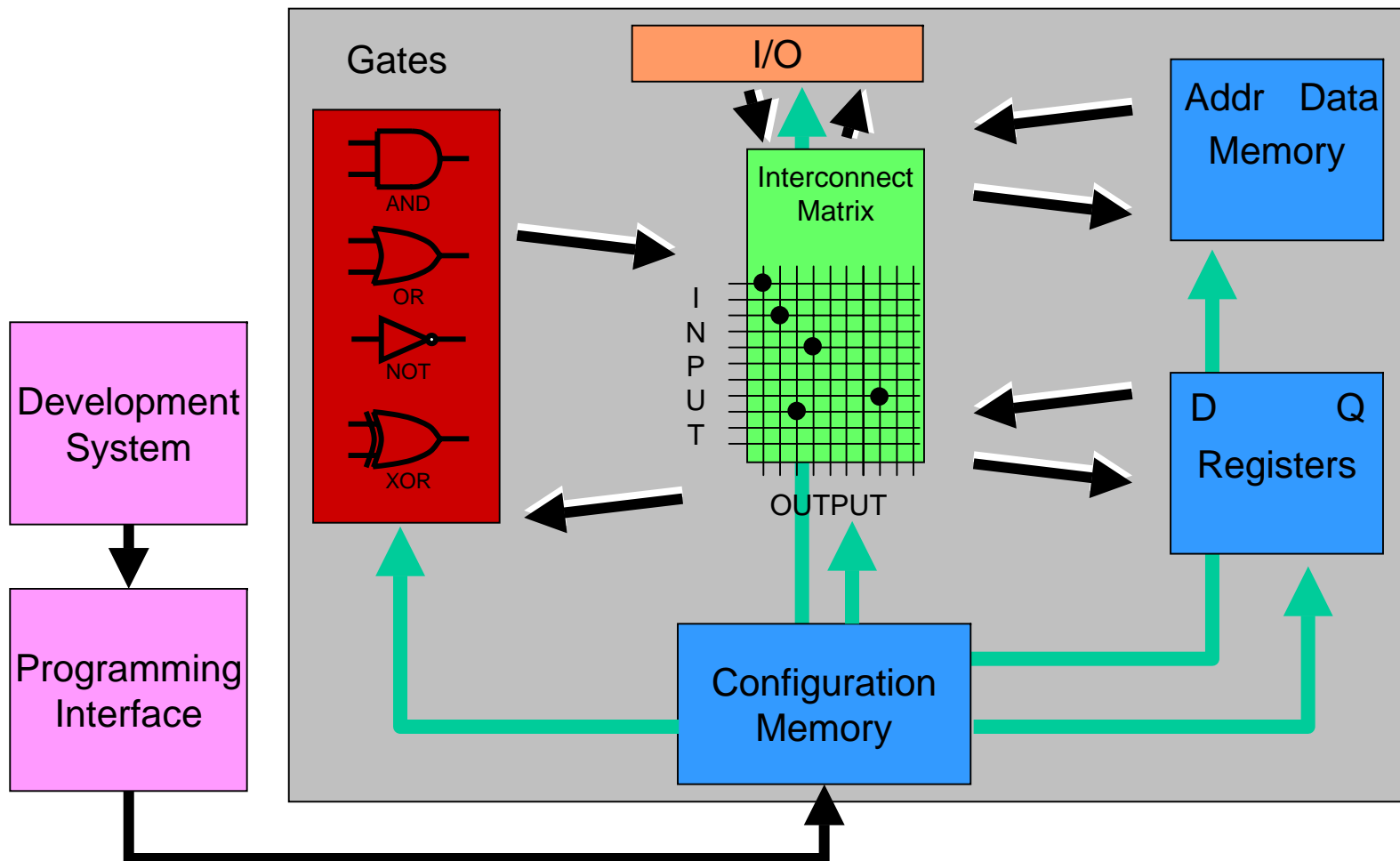


Addressing the Hardware/Software Co-design Issue



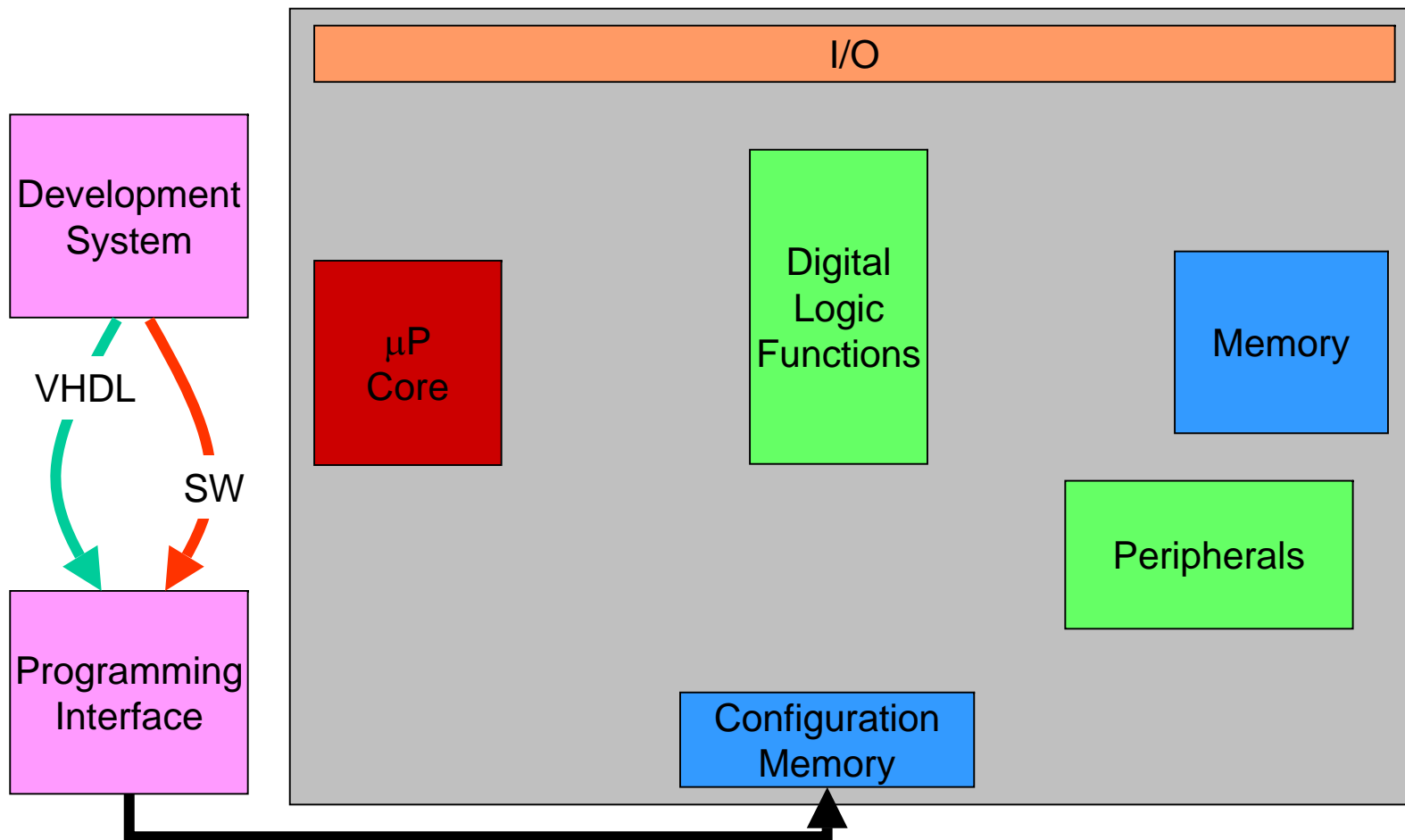
FPGAs with Processor Cores

- FPGA concept:



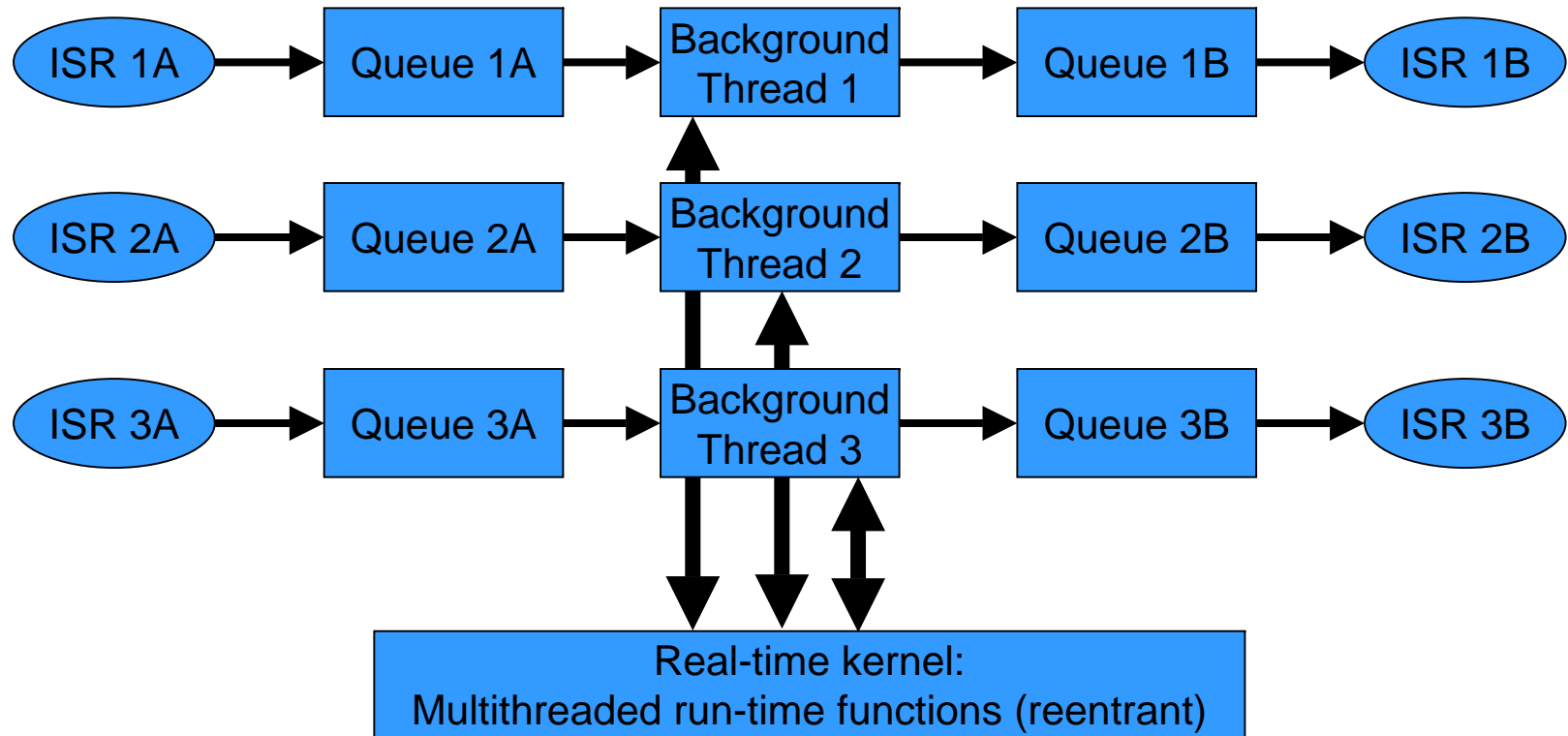
FPGAs with Processor Cores

- FPGA functional breakdown:



Multithreaded Programming

```
while(1) { . . . }
```



Context Switching

Thread 1



Thread 2



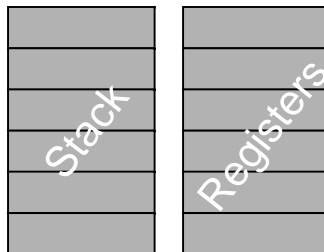
Thread 3



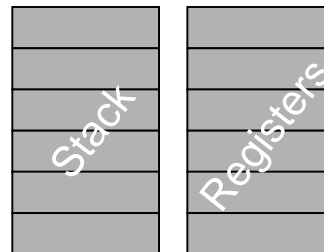
Context 1



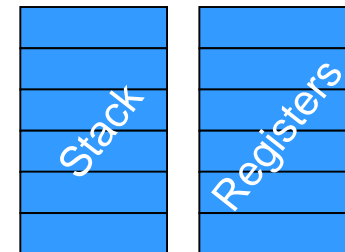
Context 2



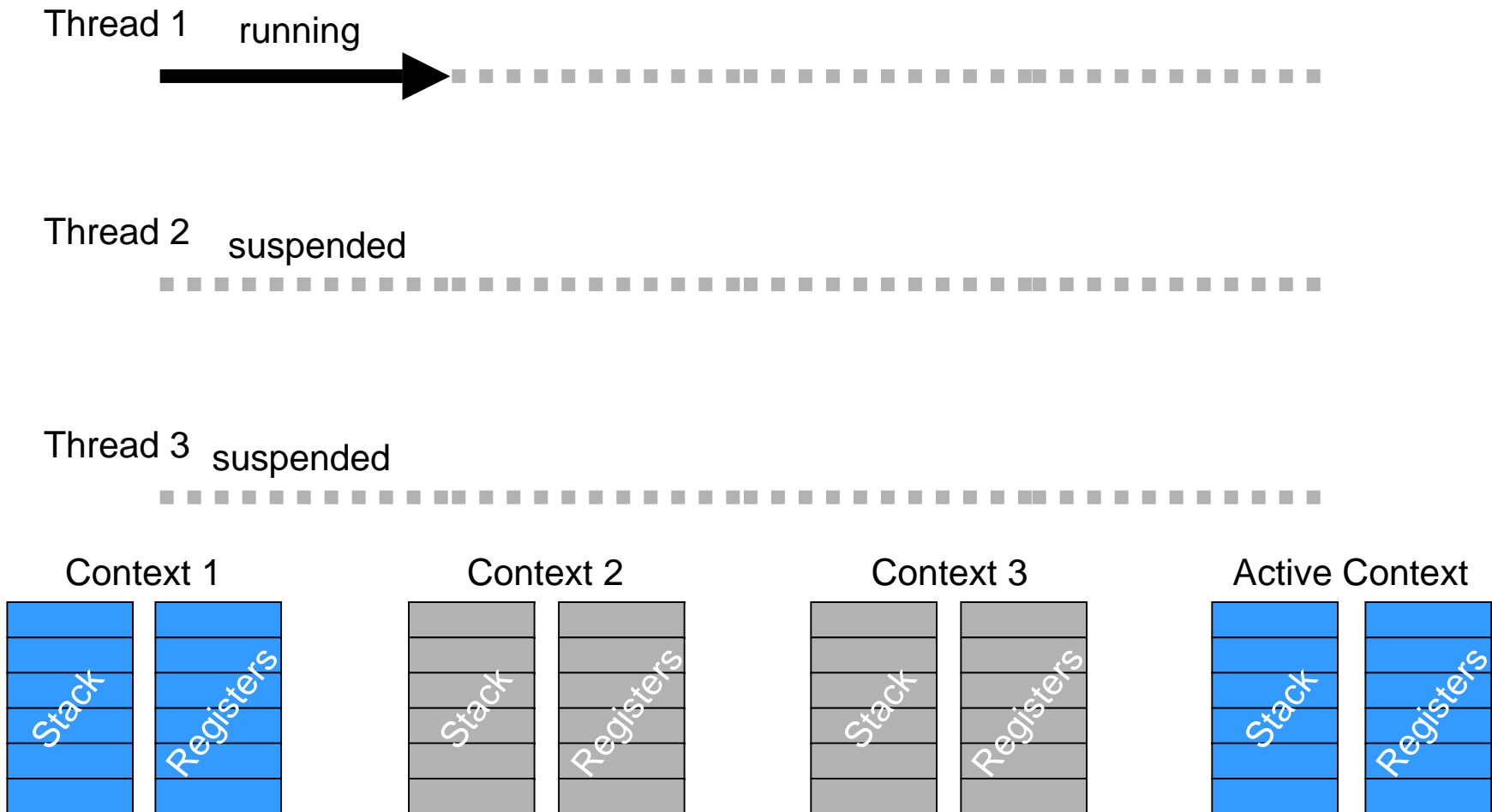
Context 3



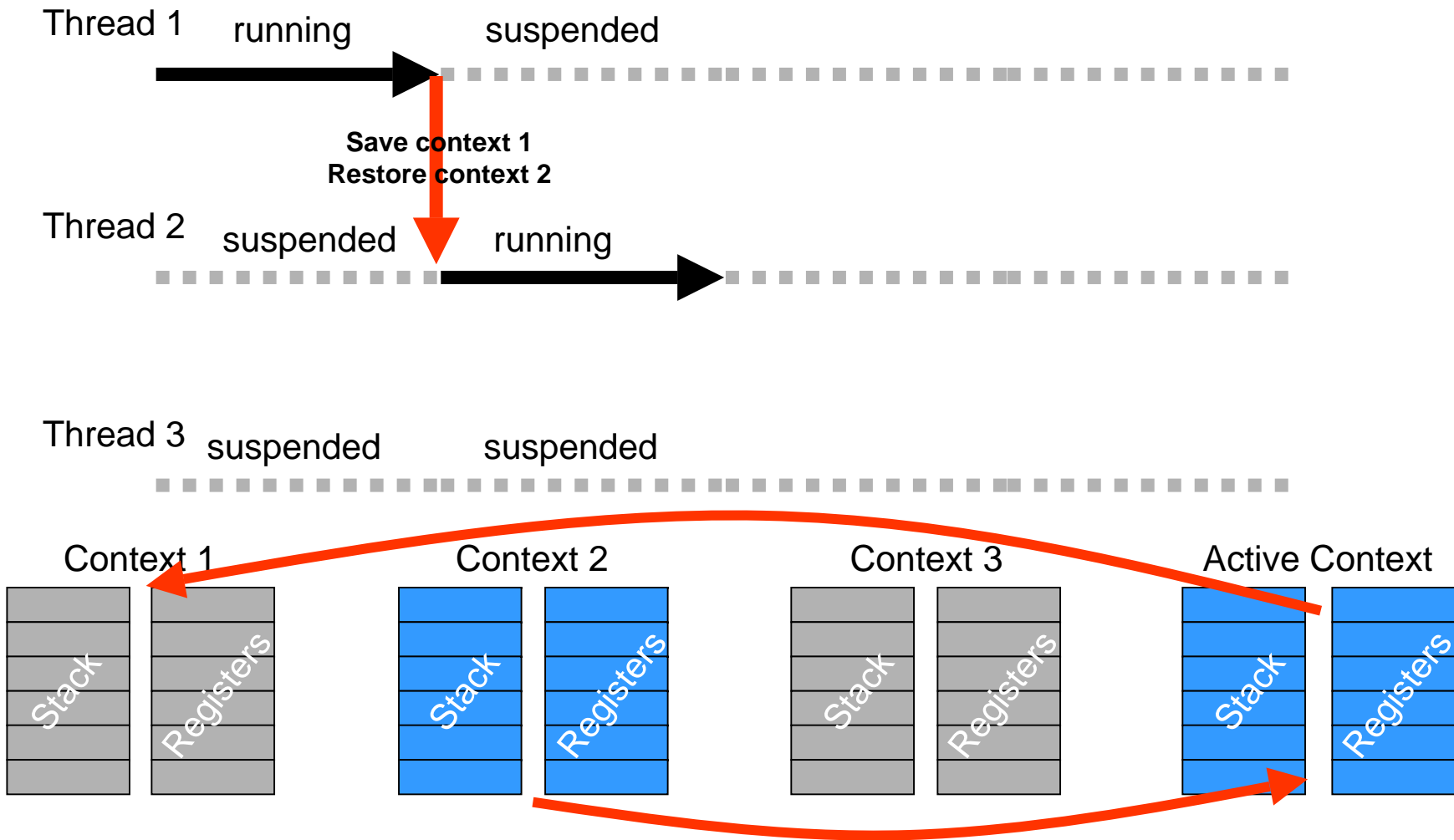
Active Context



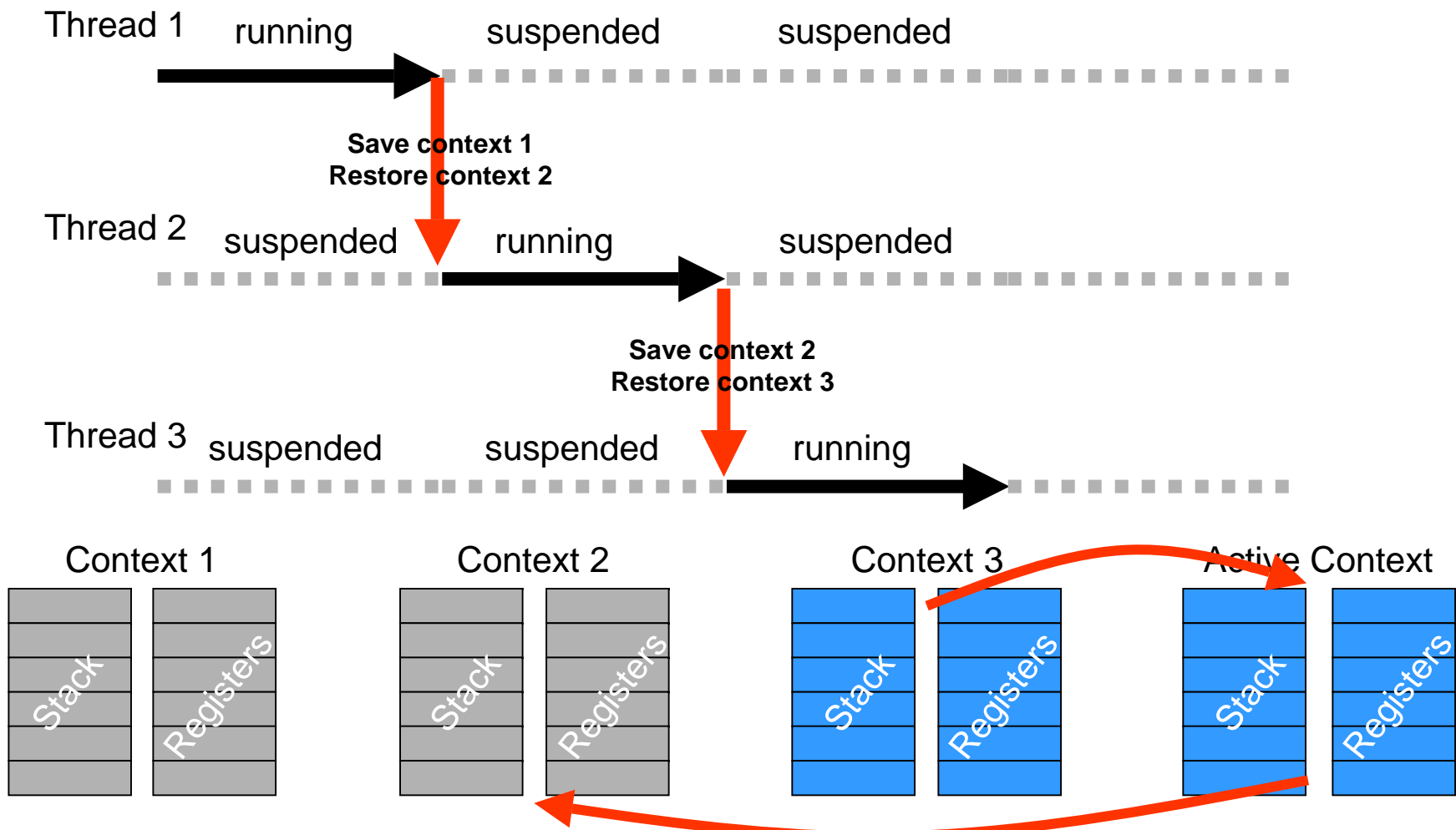
Context Switching



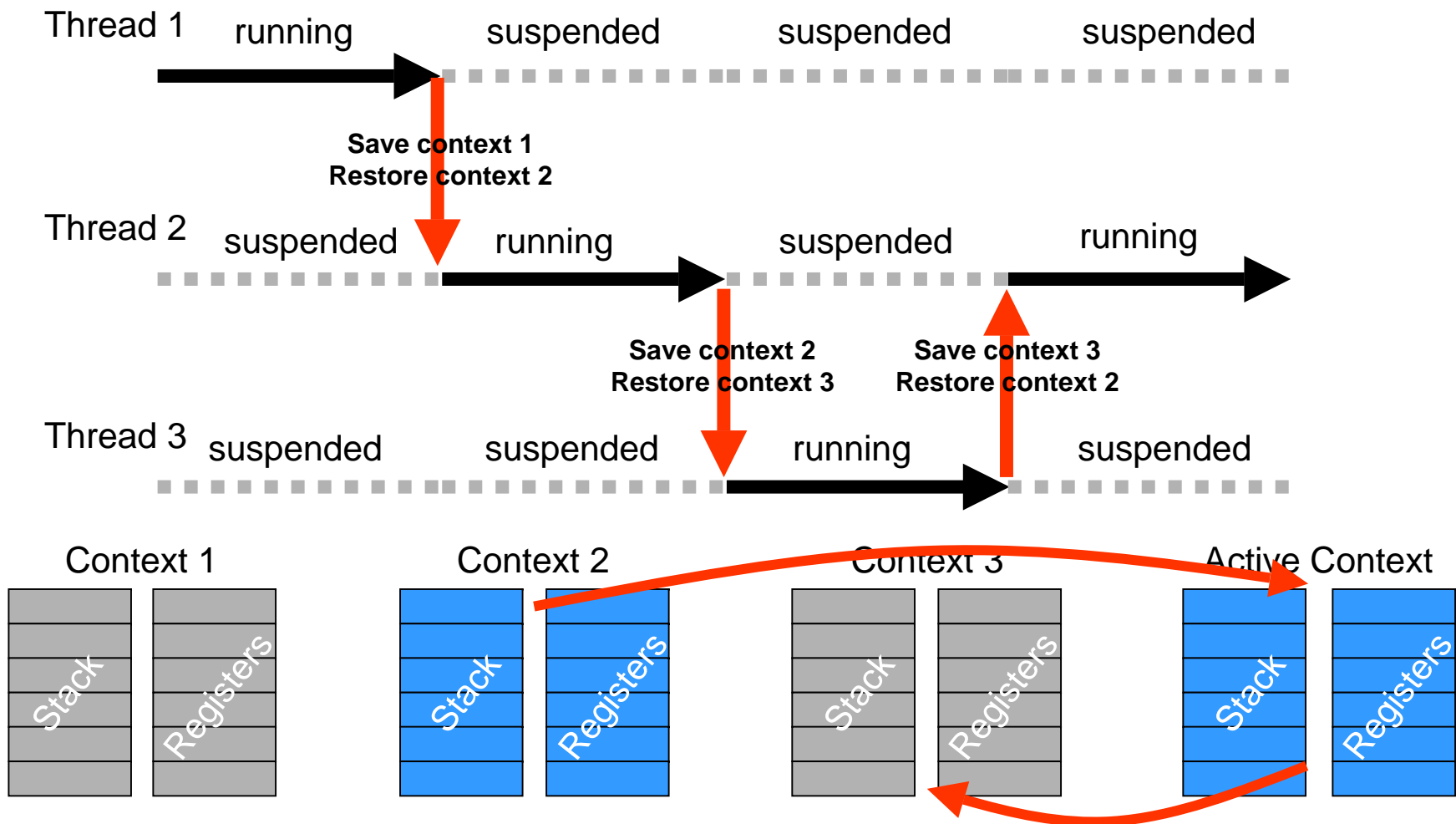
Context Switching



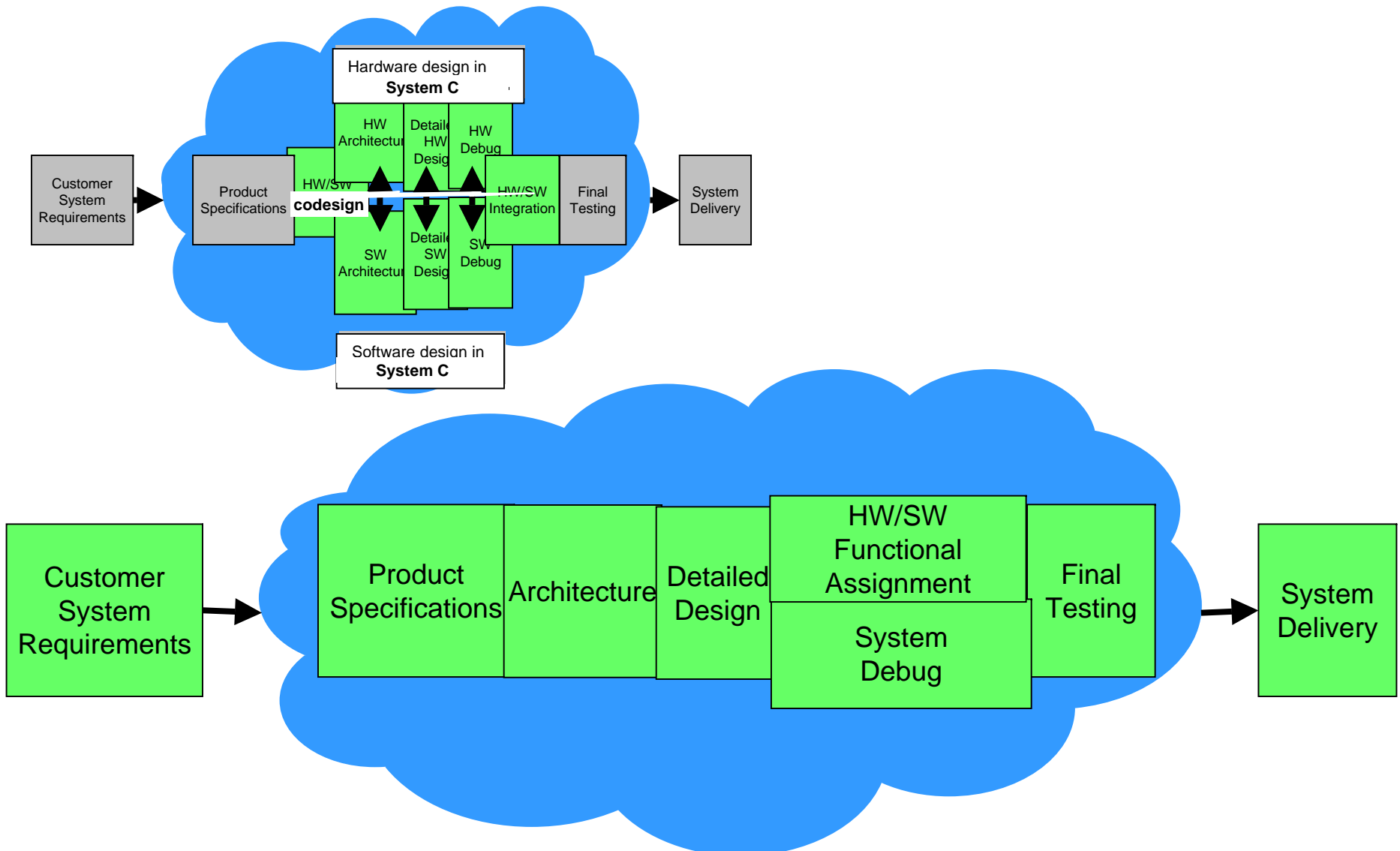
Context Switching



Context Switching

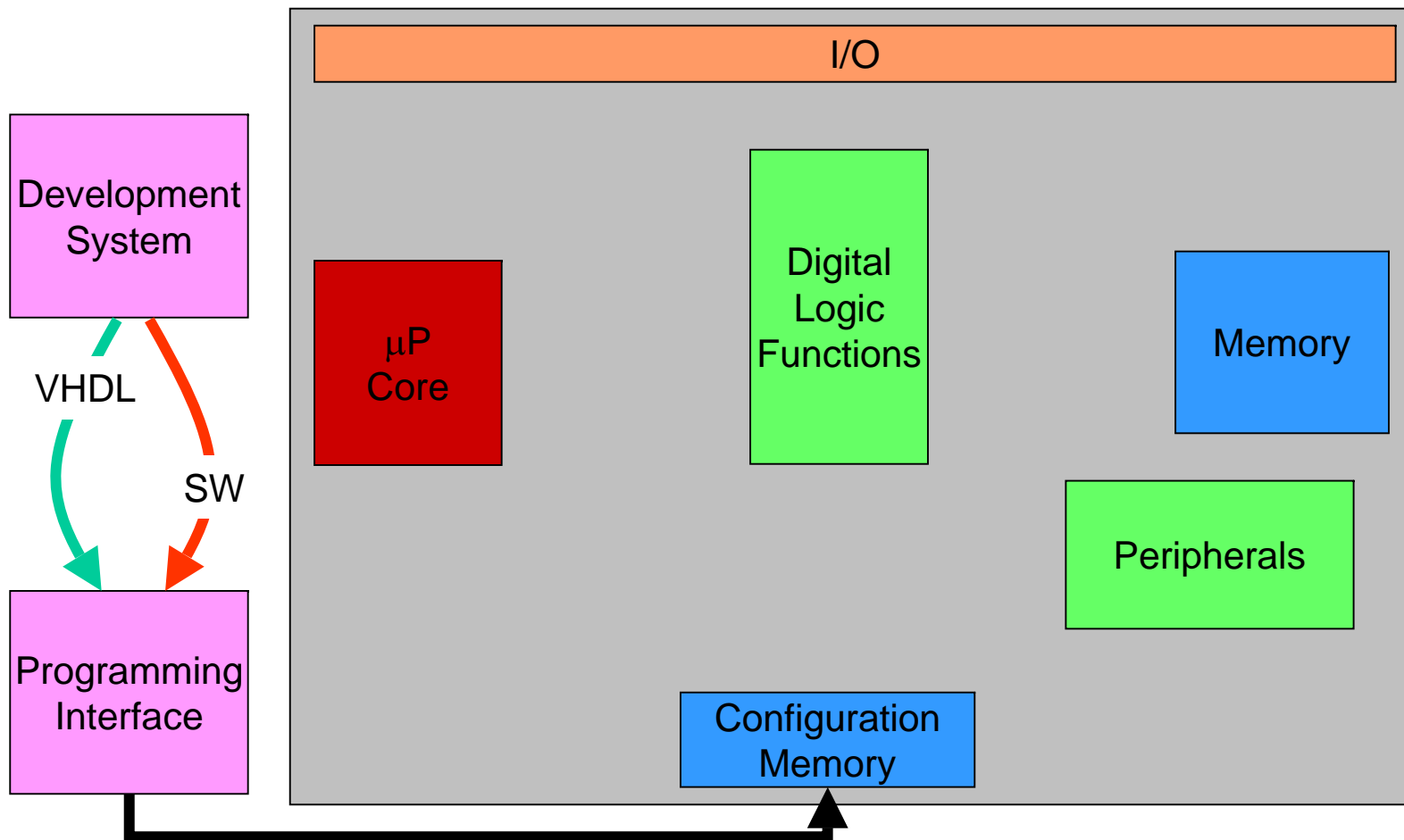


Addressing the Hardware/Software Co-design Issue



FPGAs with Processor Cores

- FPGA functional breakdown:



Open Research Questions

- If you are doing HW/SW co-design
 - on an FPGA-like device with microprocessor core **and** digital logic,
 - how can you support execution threads that cross the boundary between hardware and software?
 - This is part of the HW/SW partitioning issue – you might decide to realize a function in hardware or software, as you do the HW/SW functional assignment.
 - How do you support simple synchronization primitives, like `signal()` and `wait()`? E.g., what if a software process needs to wait on a hardware function?