Multicore start execution synchronization

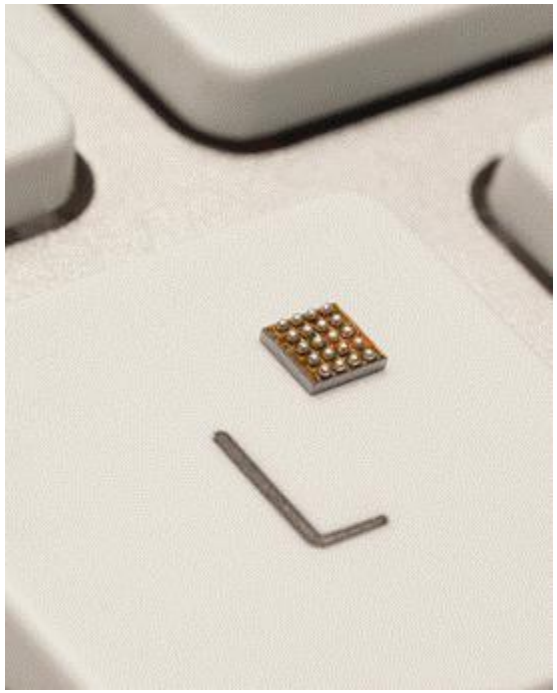
Razvan Ionescu, Radu Ivan, Ionut Vicovan

Oct.08.2014

Agenda

- Synchronize multicore applications
- Using Trace IP to synchronize multicore applications
- Synchronization mechanisms comparison
- Conclusion (debate)

Synchronize multicore applications

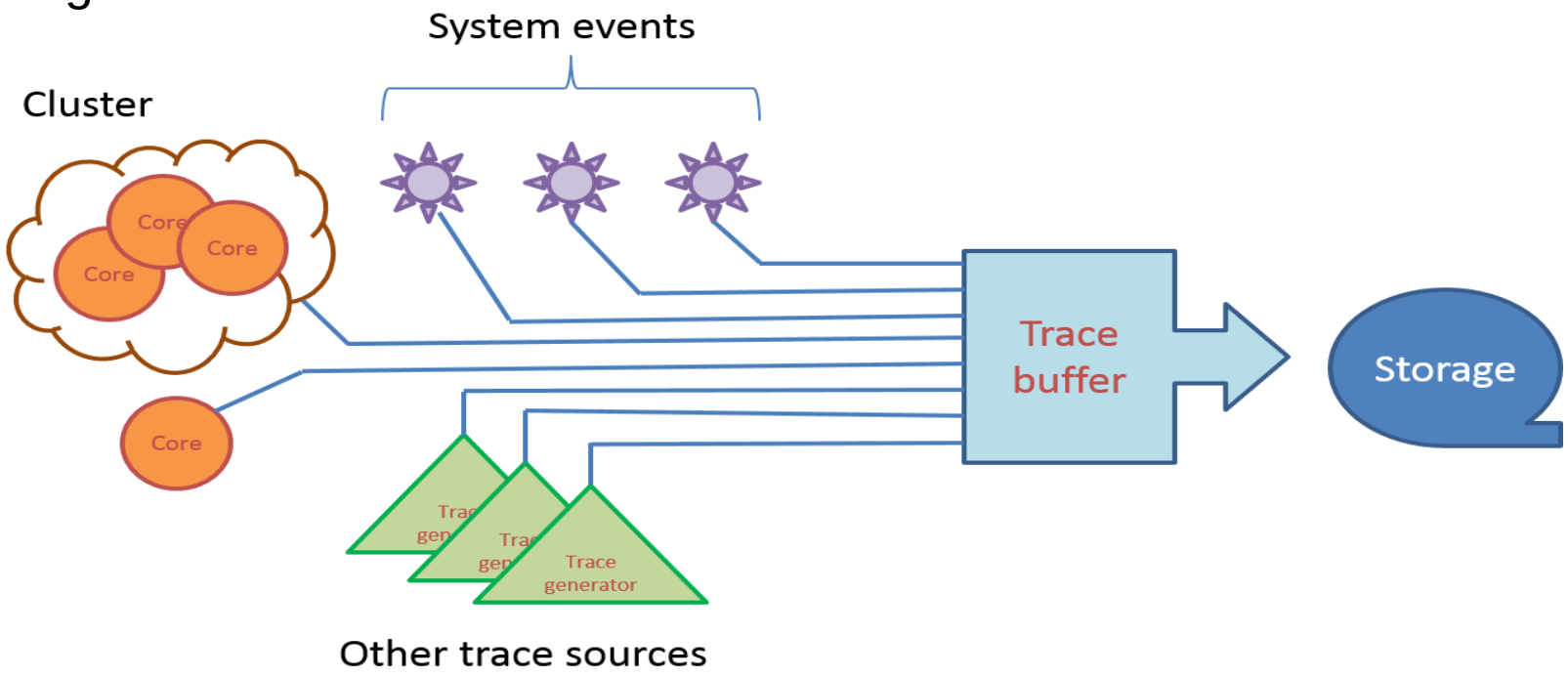


- Start multiple cores in the (approximatively) same time
- Barrier concept
  - Embedded systems
  - Platform level
  - Software accessible, hardware implemented

- **Challenge:** minimize the skew time between start execution of the cores at an affordable price and in an easy to use manner.

Using Trace IP to synchronize multicore applications

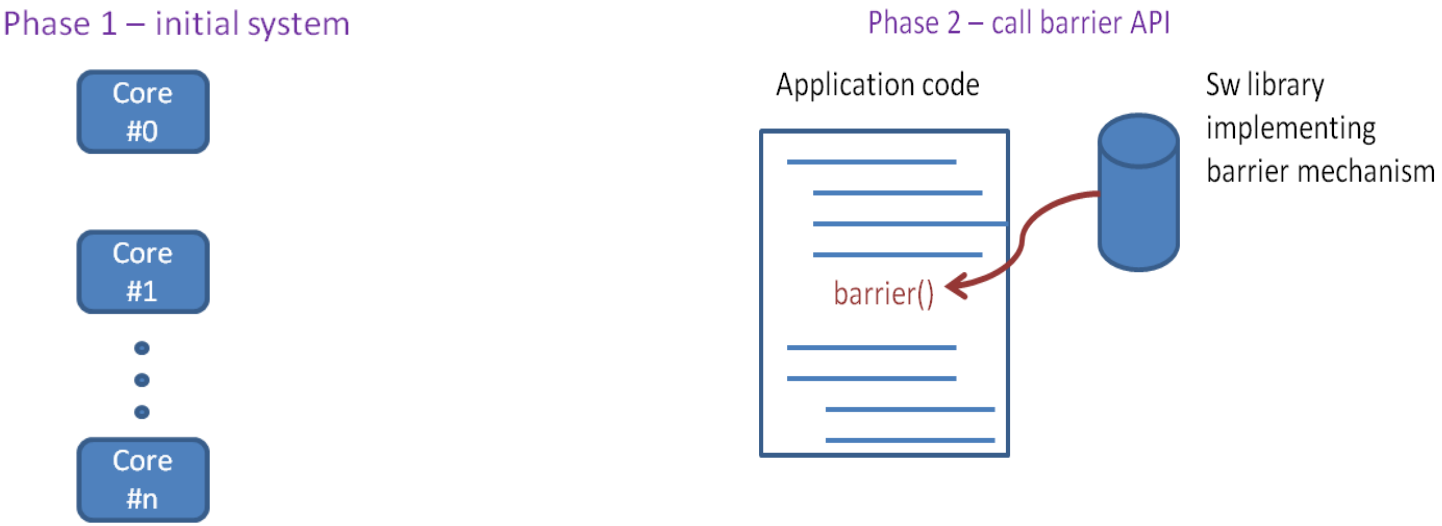
- Complex embedded systems processors have Trace IP blocks
  - Trace IP is a must have for multicore processors
- Leverage Trace IP in deployed embedded products
  - Little to no use of Trace IP infrastructure in deployed systems – normal usage



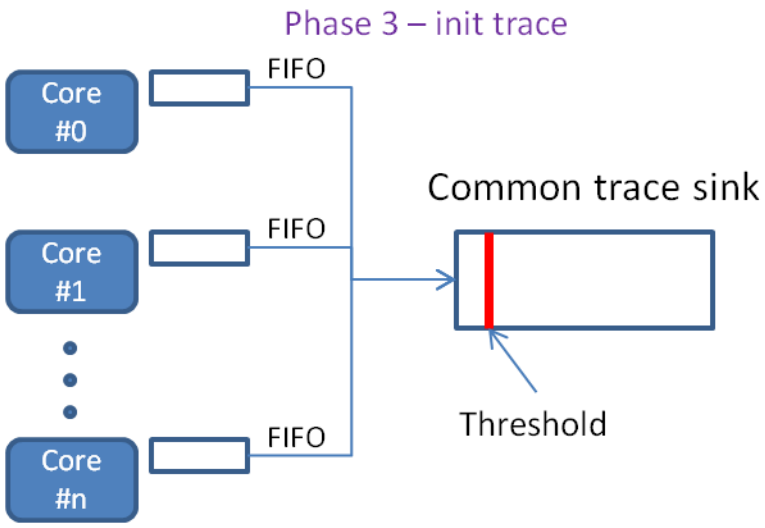
Trace IP features

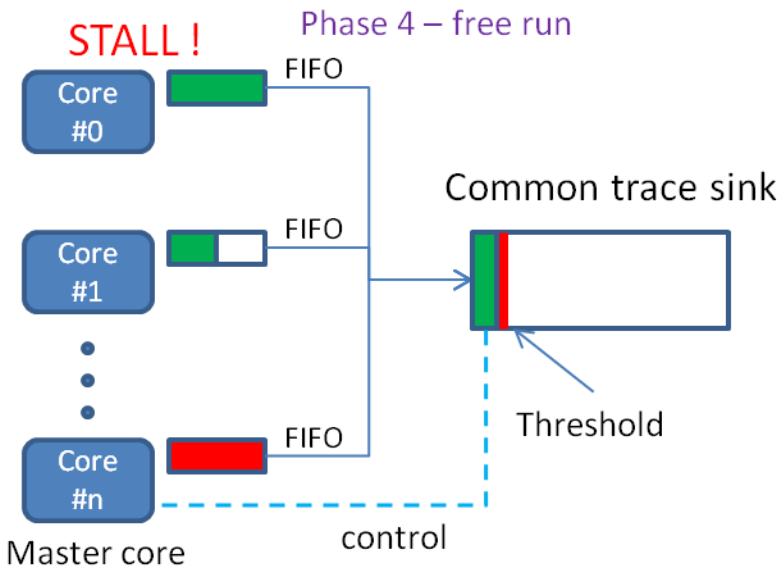
- Ability of trace buffer to stall core execution when it is full
- Central trace buffer (platform level) can control cores execution
- Topology of Trace IP allows to propagate signals from central trace buffer to cores
- **Caution:** usually cannot stall other trace generators than cores

Synchronization mechanism

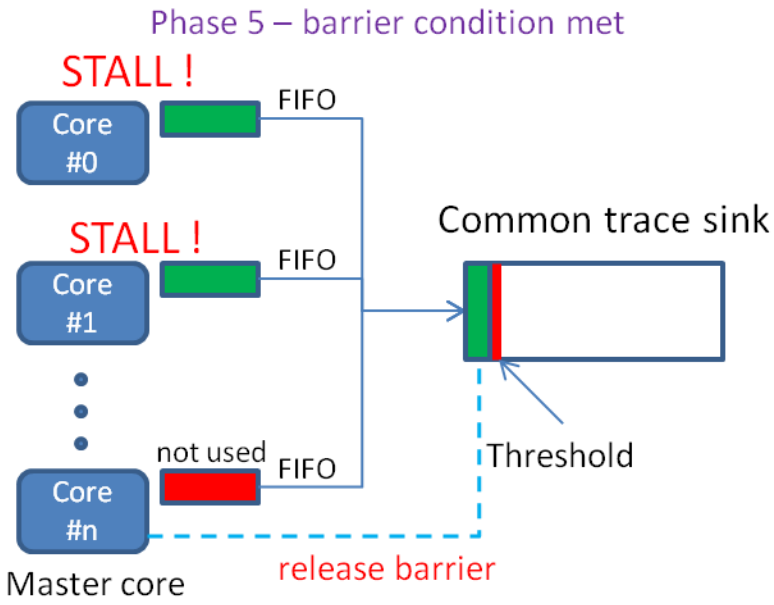


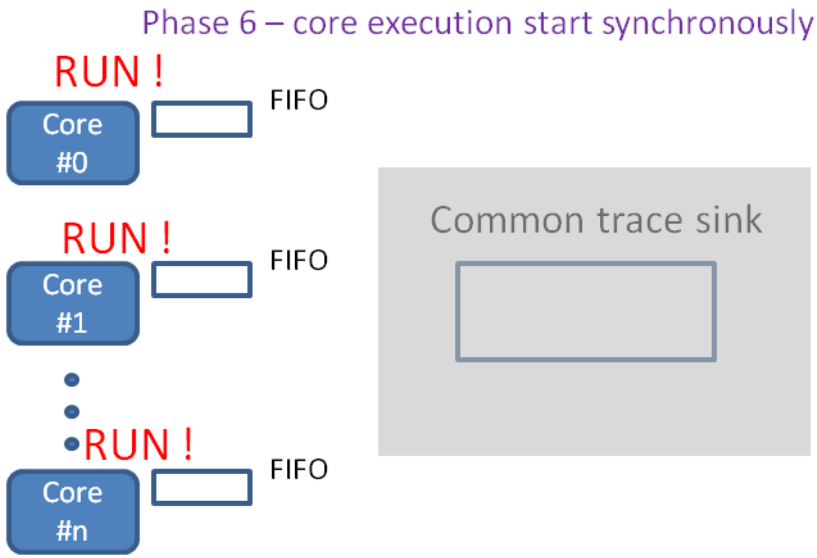
Synchronization mechanism (cont.)





Synchronization mechanism (cont.)





Software library to use synchronization mechanism

Core#0:	Core#1:	…..	Core#5(master)	Comments
code	*code*		*code*	Application code
barrier()	barrier()		barrier_master(6)	Call for barrier API
synchrone_code	*synchrone_code*		*synchrone_code*	This code will start run synchronously

- Easy to use
- Hide hardware configuration complexity
- Can build complex scenarios
- Make customer/user code portable

Synchronization mechanisms comparison

Feature(s)	Hw barrier register implemented at SoC level, with bits allocated for each core	Using a shared memory location	Cross triggering hw mechanism	Broadcasting messaging	Synchronization mechanism using hw trace infrastructure
Precise	Yes	No	Yes	probably No	Yes
Fast	Yes	No	Yes	No	Yes
Special designed hw circuit	Yes	No	probably Yes	maybe Yes	probably No
Requires sw modification of original application	Yes	Yes	No	Yes	Yes
Ease of use	Easy	Easy	Complex	Easy	Easy
Portability of code	No	probably Easy	No	probably Easy	probably Easy

Conclusion (debate)

By re-using Trace IP infrastructure in deployed systems, customers will benefit of a hardware precise mechanism, fast and reliable to start multiple cores in the same time. It will use a low overhead mechanism to achieve higher degree of synchronization for multicore applications.

- **Hardware precise mechanism**
- **Low cost**
- **Easy to use**

www.Freescale.com