What is it about?

- Communication compatibility among different components of an embedded system.
- Is it that important?
  - Pieces of a complex embedded system may be taken from off-the-shelf components.
  - Communication protocol in each component may be different.
  - Need a protocol converter to allow communication
  - YES!

Notion: Interface

- Two views of interface
  - Passive entity: provides pin-mappings!
  - Active entity: a separate process in charge of communication on the component's behalf.
    - Control flow of its own.

Why converter and interfaces?

- Bus-based communication - Each component hooked to the bus, has its own bus interface.
  - Common in system-on-chip bus protocols such as AMBA (in ARM).
  - Bus controller or arbiter acts as the central converter enabling communication among the bus interfaces of the components.

Organization

- Task of converter/interfaces
  - Enable communication.
  - Resolve protocol incompatibilities.
- What are the common incompatibilities?
- Converter synthesis
1. Signal ordering mixed up

Sender
- start
- addr
- data
- stop

Receiver

Sender
- start
- addr
- data
- stop

Receiver

Sender
- start
- addr
- data
- stop

Receiver

S-Interface

R-Interface

2. Different Signal Alphabet

Sender
- request
- data
- ack

Receiver

Sender
- ready
- data
- stop
- ack

Receiver

Sender
- request
- data
- ack

Receiver

S-Interface

R-Interface

3. Mismatch in data format

Sender
- request
- data
- ack

Receiver

S-Interface

R-Interface

Converter

Shift Register
(shift every cycle)

Buffer
(group and send every 8 cycles)

Bits Sent

Receiver

00000001
00000010
00000101
00001011
00010110
01100100
10110001
01100110
01101000
10110001
01100110
01101000

OR

10110001

Copyright (c) 2009, Abhik Roychoudhury
4. Mismatch in data rates

- Smoothing out differences in data rates.

Organization
- Task of converter/intrafaces
  - Enable communication.
  - Resolve protocol incompatibilities.
- What are the common incompatibilities?
- Converter synthesis
  - Represent native protocols and converter as FSM.

Example: Incompatible protocols

Converter Synthesis

Converter under construction
Converter

- Product of the incompatible protocol FSMs.
  - Sending and receiving of each signal de-linked.
  - Sending and receiving are separate events.
- What about shared signals?
  - Signals in the common alphabet.
  - Typically include “data” signals.
  - If the sending and receiving happen together, we can avoid storing these signals in converter.
    - This restricts behaviors of converter - fewer exec. traces.

Converter

- Protocol FSMs may contain infinite length traces.
  - Protocol FSM = Interface of a component.
  - Ongoing interaction with other components.
  - FSM with cycles.
- In that case,
  - Converter also needs to be a cyclic FSM.
  - “Sessions” of the protocol interactions across components need to be synchronized.
    - Explicitly (adding new messages), or
    - Implicitly (same msg. at session end in each component).

Synchronizing the “sessions”

Summary: Converter synthesis

Message \( m \) between component1 and component2 flows through converter.
No direct interaction between components.