From clock constraints to GALS executives/shells/wrappers

Jacky Potop
Yves Sorel
Robert de Simone

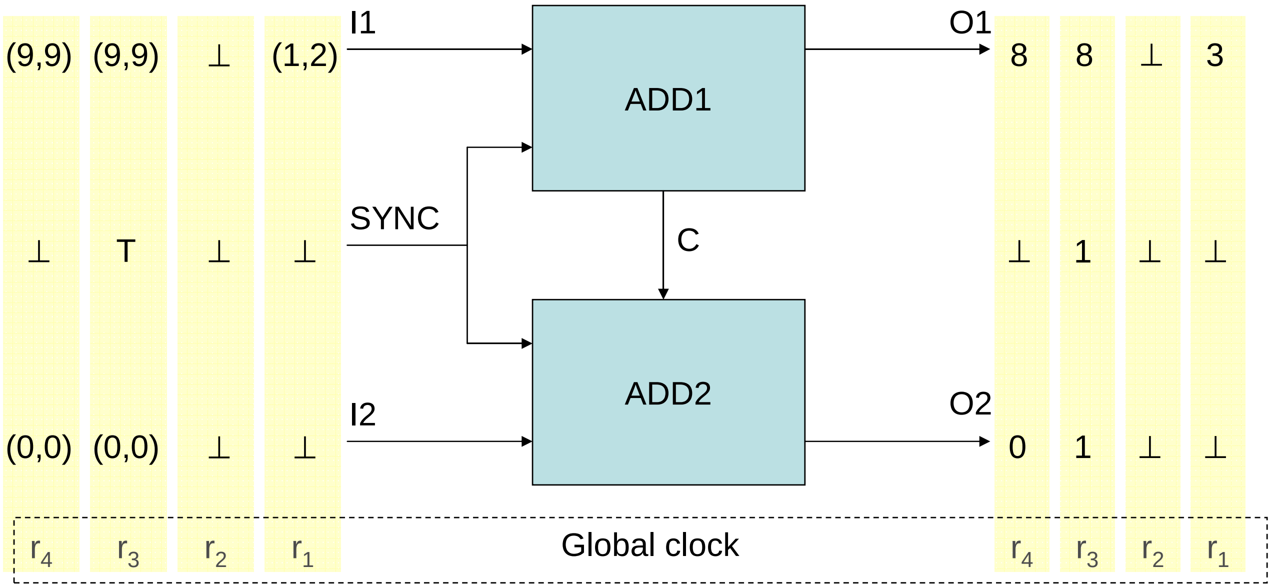
INRIA, projet AOSTE
(Rocq & Sop)

Outline

- The problem
- Weak endochrony
  - Theory
  - Shell generation
  - Checking weak endochrony
- Composition issues
- Future work

Modular synchronous system

- Uniform length computations



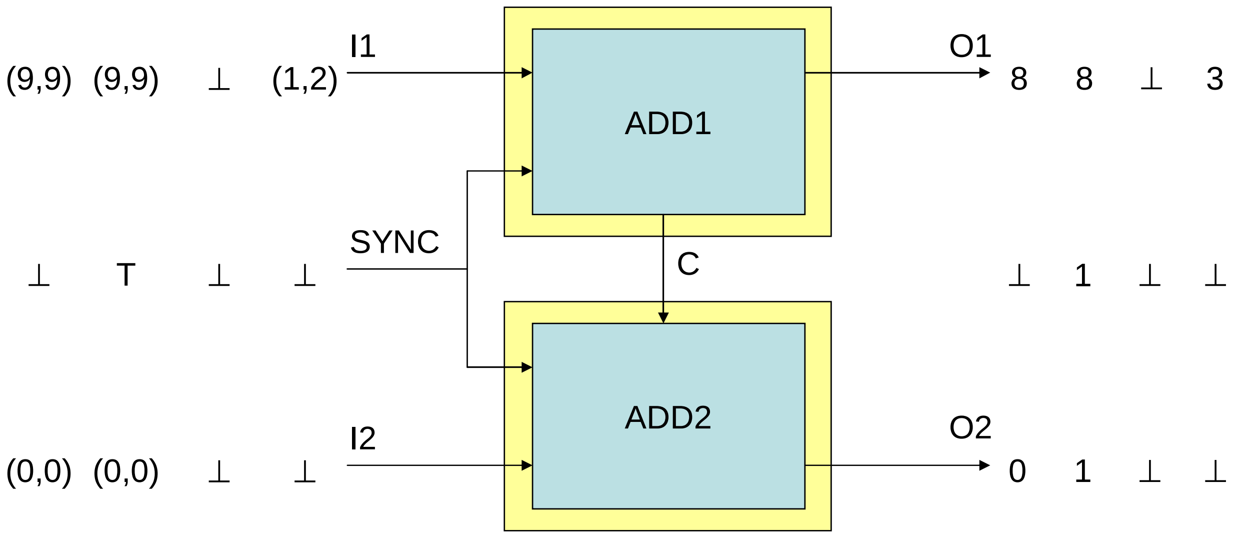
From synchronous to GALS

- Modular synchronous system
  - Absence as an explicit value (⊥)
  - Reactions are fired by consuming one value on all inputs (and producing on all outputs)

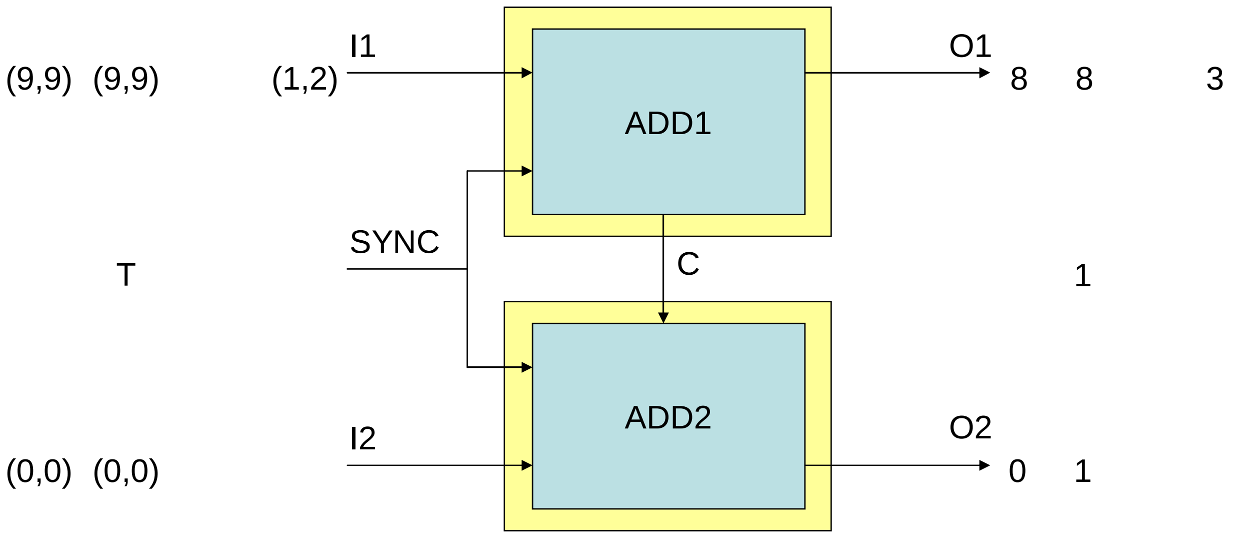
→ GALS
  - No timing information
  - Reaction firing: wrappers/shells
  - Our approach: erase ⊥ values

When is it feasible, while preserving the semantics?
  - Same I/O sequences, without the ⊥ values
  - Different timing

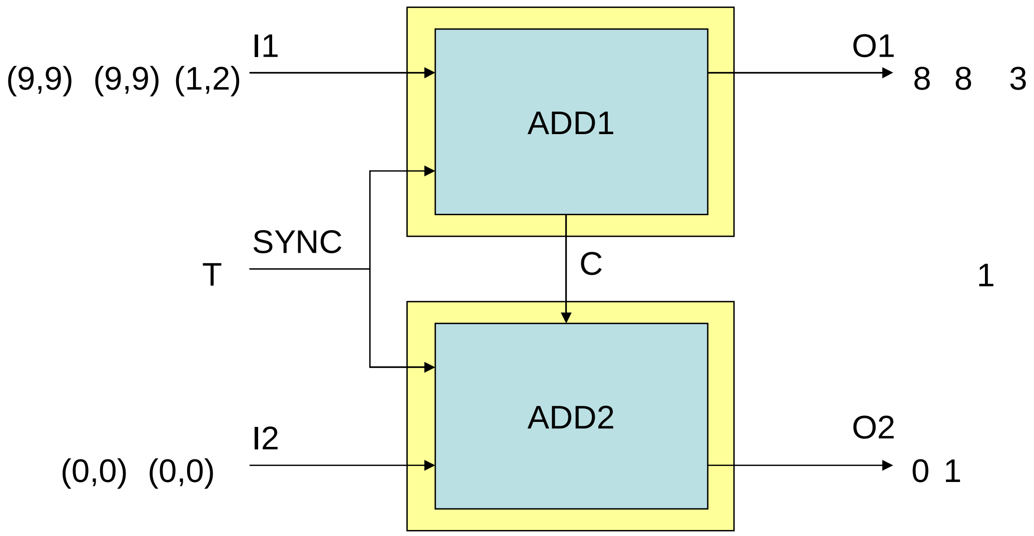
From synchronous to GALS



From synchronous to GALS



Shells and pearls

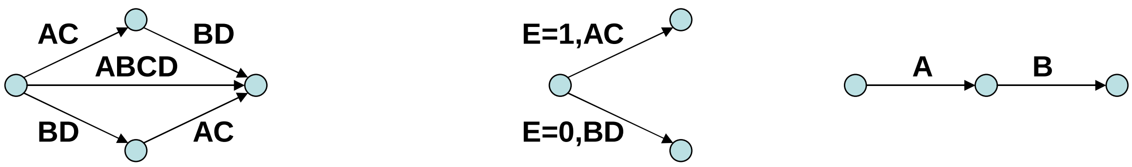


Wrapper: intended specification

- Reads values on input channels
- When enough input is available:
  - Add the missing ⊥ values to complete the input vector
  - Activates its pearl(s) (clock gating)
  - Remove the absent values from the output
  - Propagate the results on the output channels when space is available
  - Mark used inputs as read (new ones can arrive)
- In general, does not preserve semantics (or does so by reintroducing explicit absence)
  - Need correctness criteria: **weak endochrony**

Weak endochrony (Potop, Caillaud, Benveniste 2004)

- Property to be checked on the synchronous module
  - No reaction to signal absence
  - Potential asynchrony is contained in the synchronous module already
- In every state, if I, J sets of inputs that can trigger reactions:
  - If $I \cap J = \emptyset$ then the reactions of I and J commute



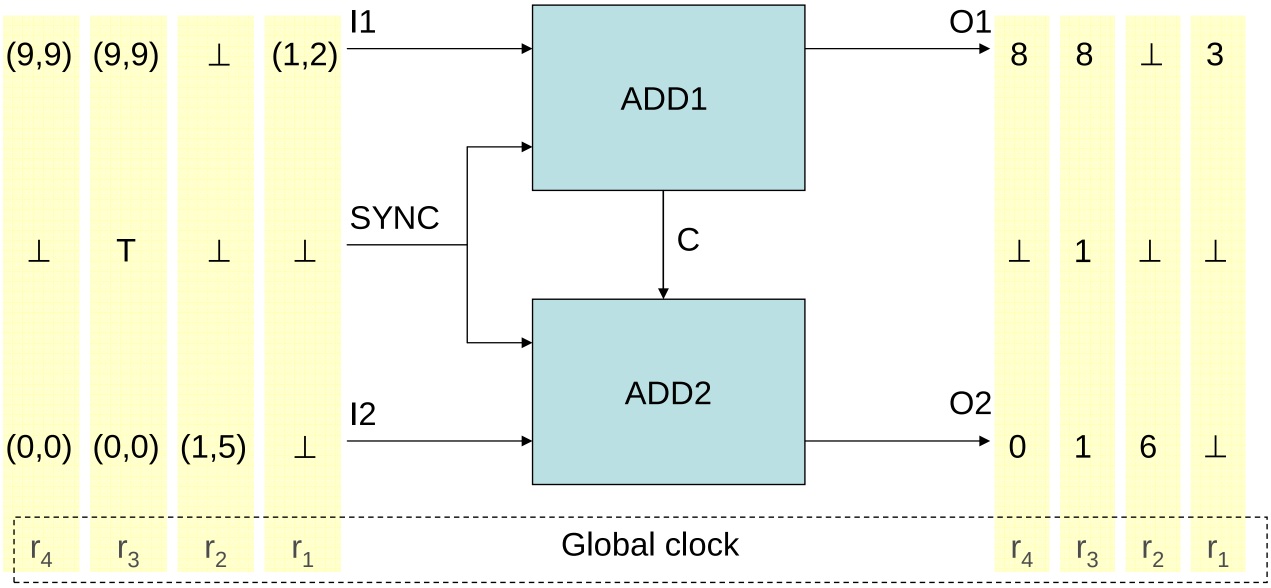
  - If no signal has different values in I and J, then $I \cap J$, $I \cup J$, and $I \setminus J$ can trigger reactions.

Weak endochrony

- Ensures that constructing synchronous input is
  - Deterministic, up to commutation of independent reactions
  - Possible using single-place buffers
- In this paper, stateless weak endochrony:
  - If I, J sets of inputs that can trigger reactions, if no signal has different values in I and J, then $I \cap J$, $I \cup J$, and $I \setminus J$ can trigger reactions.
- Two issues :
  - Checking/enforcing WE
  - Synthesizing the shells

Weak endochrony

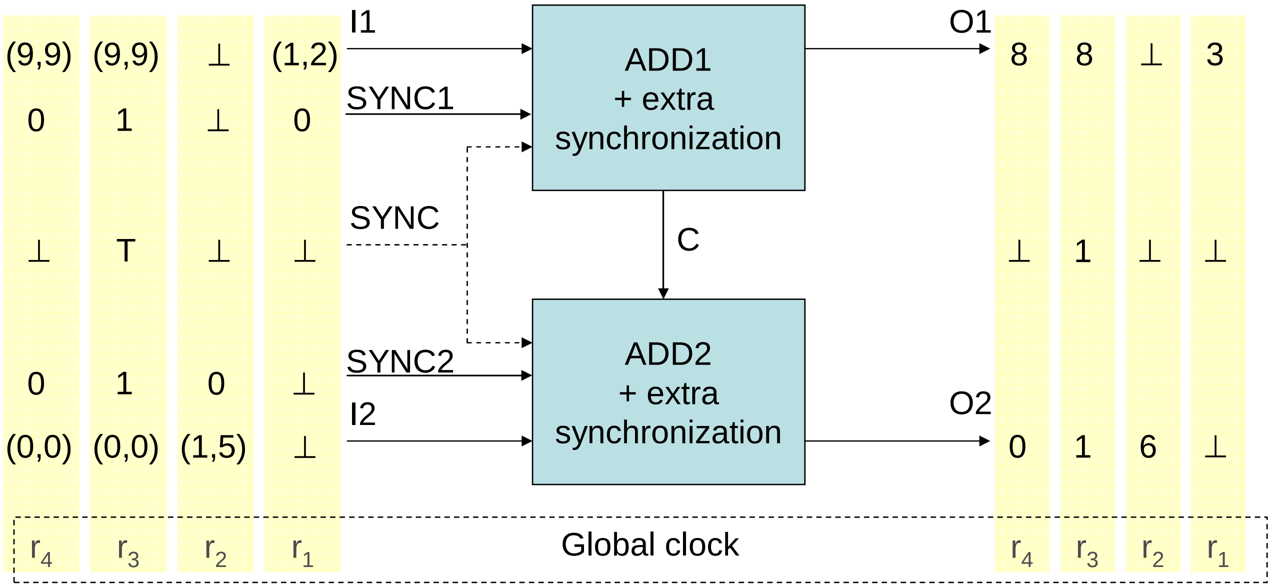
- Counter-example



$r_3 \backslash r_4$ not a reaction (r3,r4 cannot be distinguished in an asynch environment)

Weak endochrony

- Example



New signals make signal absence not necessary

Weak wndochrony

- Atoms = minimal reactions
  - Generators of all reactions.
  - Two different atoms that share a variable have contradictory inputs
- Example
  - r_1, r_2, r_3, r_5, r_6 atoms
  - $r_4 = r_5 \cup r_6$ not an atom

I1	⊥	(9,9)	(9,9)	(9,9)	⊥	(1,2)
O1	⊥	8	8	8	⊥	3
SYNC1	⊥	0	0	1	⊥	0
SYNC	⊥	⊥	⊥	T	⊥	⊥
C	⊥	⊥	⊥	1	⊥	⊥
I2	(0,0)	⊥	(0,0)	(0,0)	(1,5)	⊥
O2	0	⊥	0	1	0	⊥
SYNC2	0	⊥	0	1	0	⊥
	r_6	r_5	r_4	r_3	r_2	r_1

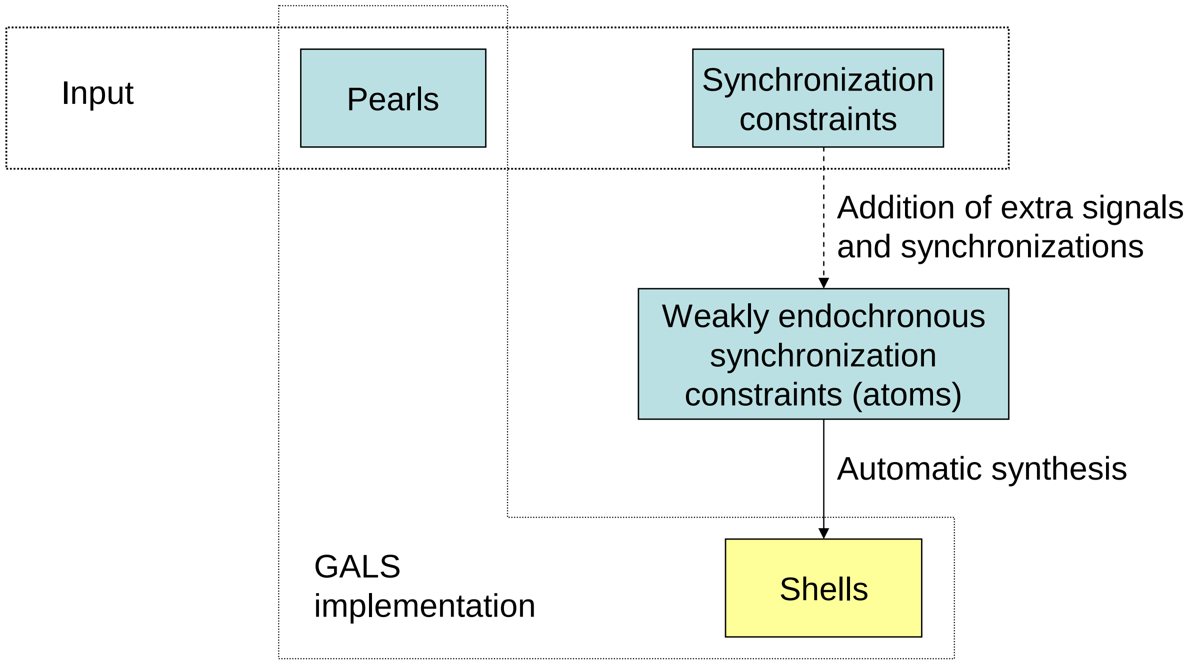
Determining weak endochrony

- Compute a smallest set of reactions that generate all the other by union
- The generator set has the properties of an atom set *iff* the system is WE.

Shell generation for WE

- Component = Shell +pearls
- Shell = concurrent triggers (1 per atom)
  - Atom trigger:
    - await atom input
    - acquire needed pearls (mutual exclusion zone)
    - set the inputs of the pearls
    - enable clock (in soft, call the reaction function)
    - disable clock upon completion
    - send the outputs
    - release the needed pearls

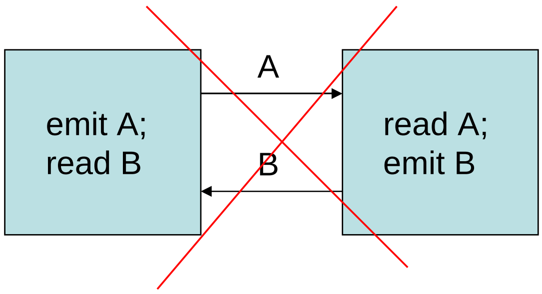
Proposed implementation flow



Simple case: state is not taken into account

Composition issues

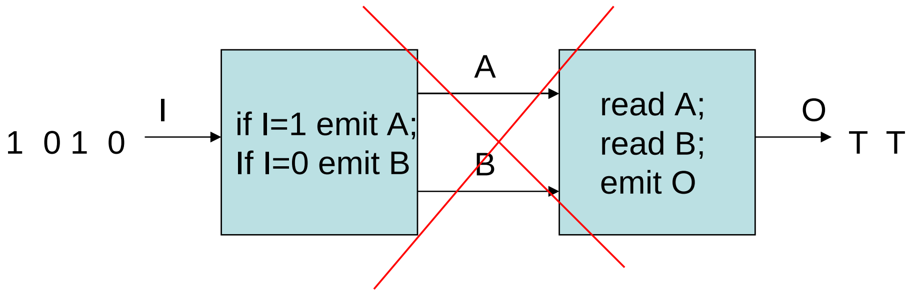
- Instantaneous feedback



  - An atom must have all input before any output is produced
  - Solution: require that all the atoms of all the components form an acyclic dependency system

Composition issues

- Weak isochrony



  - Weak endochrony does not guarantee correct resynchronization of signals
  - Weak isochrony of the set of components guarantees no incorrect resynchronization

Related work

- Latency-insensitive systems & SynDEx explicitly transmit all absence symbols
- Endochronous systems & generalized latency-insensitive add more synchronization (no independent computation of ADD1 and ADD2)

Future work

- Extend the techniques of SynDEx to complex multi-clock systems
  - Enrich the formalism
  - Extend the scheduling techniques to produced optimized executives
- Optimize shell generation
  - No need for fully separated atom triggers (can use forests of choices, generalizing the clock trees of Signal)
  - Possible pipelining of atoms in the pearls
  - In synchronous implementations of GALS systems (e.g. Latency-insensitive), can execute several atoms at the same time.