Modeling multi-periodic Simulink systems by Synchronous Dataflow Graphs

Enagnon C. Klikpo (IRT SystemX)

Jad Kathib (CEA)

Alix Munier-Kordon (UPMC)

Context

Need of computing power and multicore issues

- **Advanced embedded technologies in modern cars**
  - Standards (emission, safety),
  - ADAS, connected or autonomous car

- **Need of computing power**

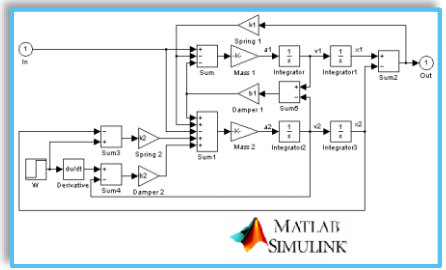
- **AUTomotive Open System Architecture : AUTOSAR**
  - Standard for the design and development of automotive E/E architecture
  - AUTOSAR 4.x introduced multicore platforms

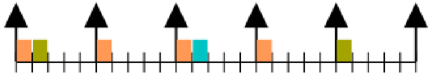
- **Multicore for critical automotive application raises some issues**
  - The mastery of the dataflow (and timing) among functionalities over cores
  - Missing a dataflow can lead to fatal scenario: e.g crash detection and inflator (ACU)

Mastering of the dataflow

Need of a dataflow formalism

- **Simulink models**
  - Synchronous sequential execution
  - dataflow communication patterns



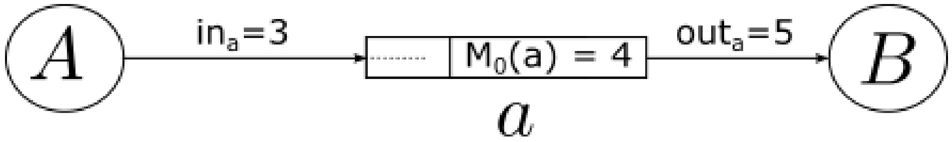


- **Need of predictability**

- **Understand and model the communication**
  - Synchronous dataflow graph (SDFG)

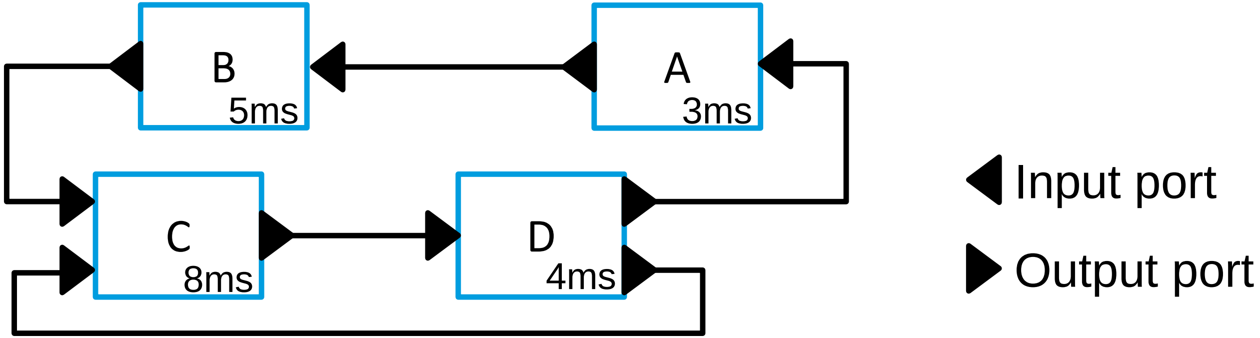
- Reminder on SDFG
- Description of communication in Simulink
- Identification of Simulink communication patterns
- Correspondence between Simulink and SDFG
- Example of a Fuel Cell Control System
- Conclusion

- **SDFG: Directed graph**
  - Lee et Messerschmidtt
  - Modeling communications in data flow applications



- **Static description:** Each process has the same behavior during execution
  - Low expressivity
  - Completely predictive

- **Matlab/Simulink:** specification and simulation tool used in Industry
- **A Simulink system is a set of communicating blocks**
- **Blocks are executed at they sample time (their period)**



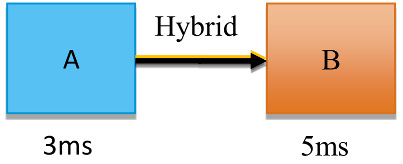
- **Block execution consists in:**
  - Input update and outputs computation (depend on the state and/or the inputs)
  - Updating the block state

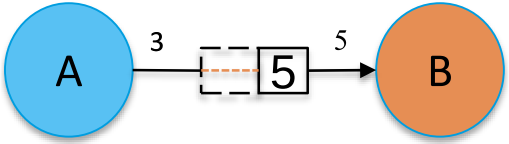
- **Several communication mechanisms in Simulink:**
  - The order in which blocks are executed
  - The input data that each execution of a block uses

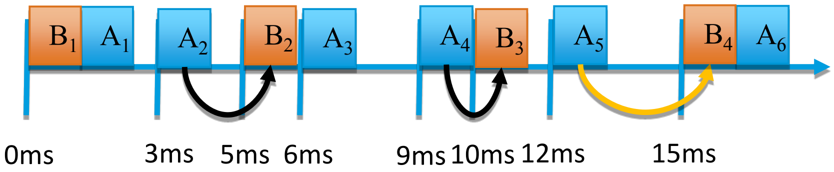
- **We have extracted three main communication patterns**
  - « Direct » communication
  - « Delayed » communication
  - **« Hybrid » communication**

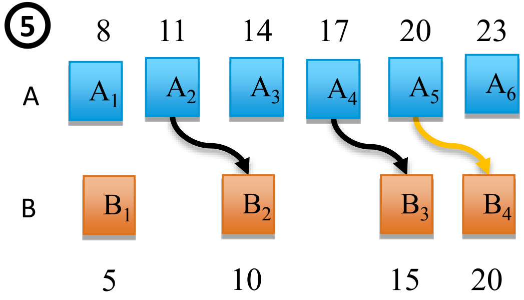
SDFG model of Simulink multi-periodic systems

Hybrid communication



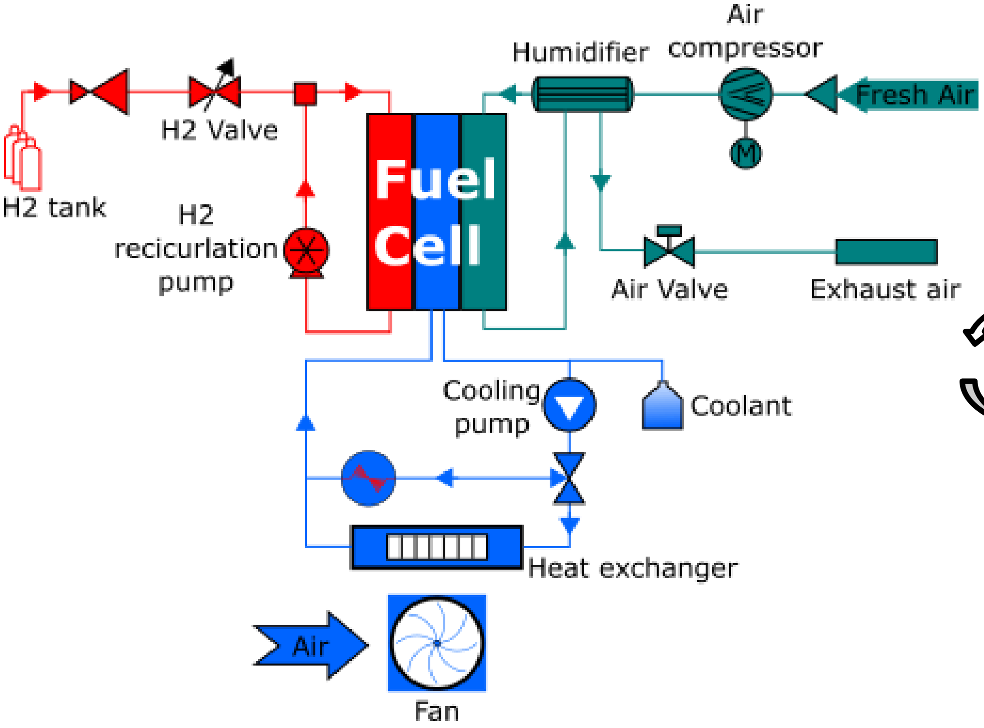


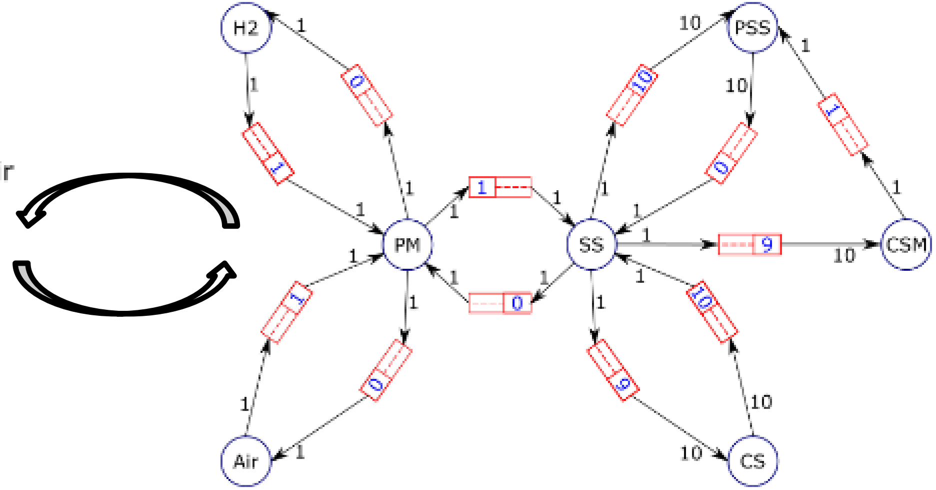




$$\omega_a > M_0 + \omega_a \cdot k_a - \omega_b \cdot k_b \geq \max(\omega_b - \omega_a, 0)$$

$$M_0(hybrid) = \omega_b$$


Air
compressor
Humidifier
Fresh Air
H2 Valve
H2 tank
H2
recicurlation
pump
Fuel
Cell
Air Valve
Exhaust air
Cooling
pump
Coolant
Heat exchanger
Air
Fan


H2
PSS
PM
SS
CSM
Air
CS

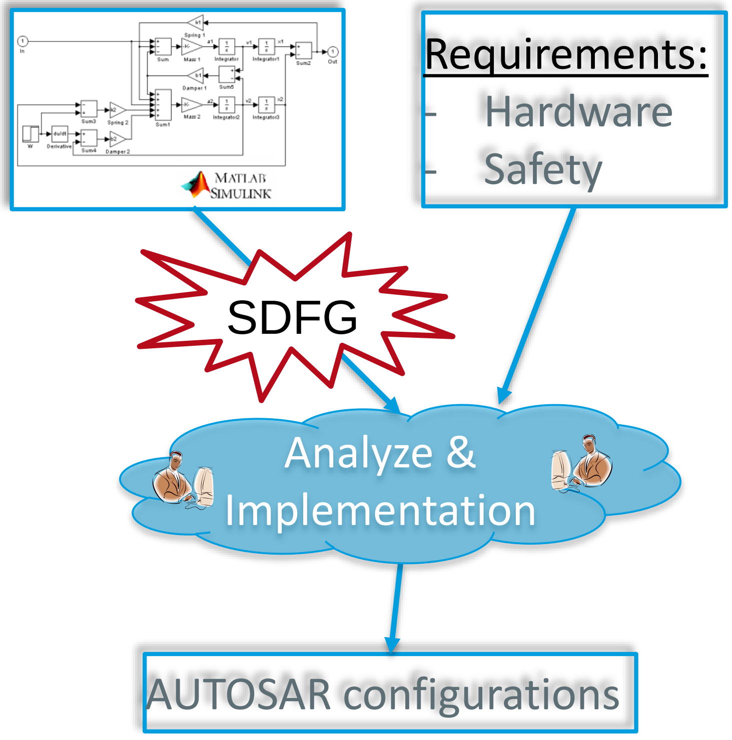
- **Formal equivalence between Simulink and SDFG**
  - SDFG results for Simulink systems implementation

- **SDFG is widely used:**
  - Initially design to for dataflow application (signal processing)
  - Compilation on multi-core (with several variants: CSDF, HSDF) Special case of petri nets (basic)
  - It has proven effective for modeling application flow

- **SDFG has existing results on**
  - Scheduling and mapping
  - Resources optimization

Perspectives

- **SDFG rather than Simulink models**
- **Preemptive Real-time implementation**
  - Use of mathematical tools of SDF
- **Other approaches and constraints**
  - Language bases approaches
    - PRELUDE
- **We are constrained by Simulink**



Arxml, source codes, …

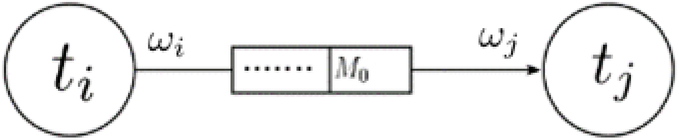
Modeling multi-periodic Simulink systems by Synchronous Dataflow Graphs

Enagnon C. Klikpo

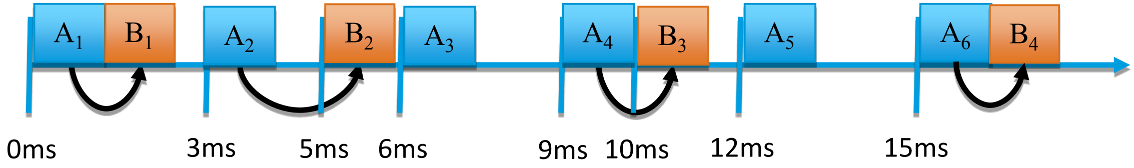
- **Modeling principle (equivalence)**
  - precedence constraints
  - data dependencies

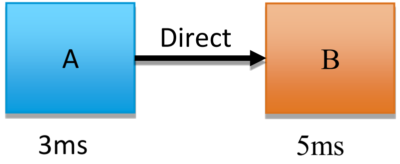
$$z_i > M_0(a) + n_i \cdot z_i - n_j \cdot z_j \geq \max(z_i - zj, 0)$$

- **The obtained data dependencies equation**
  - $\omega_i > M_0 + \omega_i \cdot n_i - \omega_j \cdot n_j \geq \max(\omega_j - \omega_i, 0)$
  - ω : periods
  - $M_0(direct) = \omega_j - gcd(\omega_i, \omega_j)$
  - $M_0(delayed) = \omega_j + \omega_i - gcd(\omega_i, \omega_j)$
  - $M_0(hybrid) = \omega_j$
  - gcd : greatest common divisor

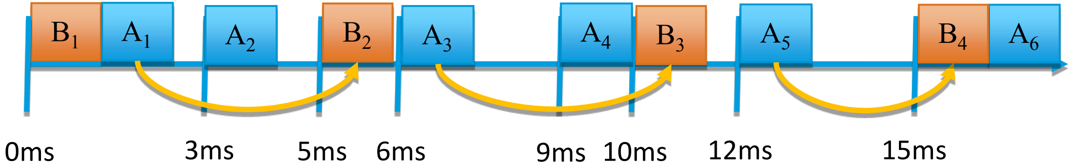


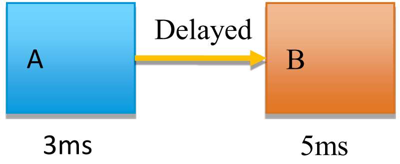
- **Direct communication**





- **Delayed communication**





- **Hybrid communication**