# A Further Step in the Incremental Design process : Incorporation of an Increment Specification

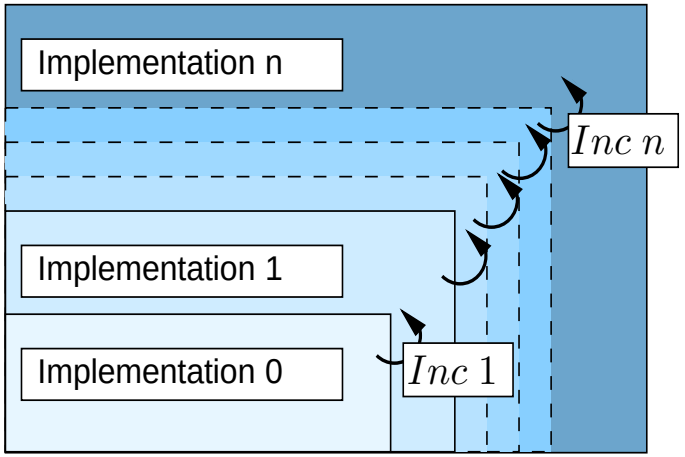
Cécile Braunstein[1] and Emmanuelle Encrenaz[2]

[1] University of Paris 6 (UPMC/LIP6/SOC)

[2] Laboratoire Spécification et Vérification (LSV)

# Context

- Incremental design process of hardware components : successive addition of new behaviours
- Verifi cation by model checking




- Writing relevant properties
- Alleviating the verifi cation process

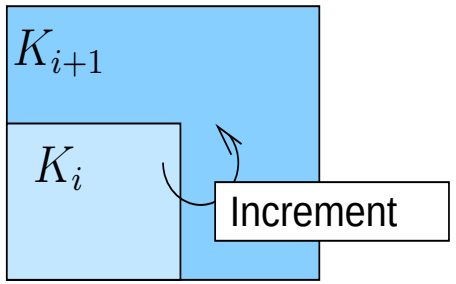
# Outline

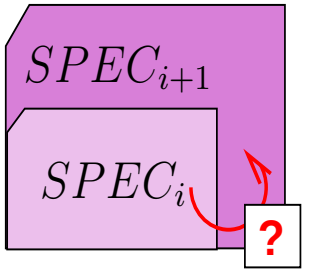
- The incremental design process
- Increment defi nition
- CTL properties transformation
- Concluding Remarks

# The incremental design process

A design framework inspired by hardware designers :

- Successive additions of new behaviours
- Conservation of existing behaviors : non-regression guaranty




$K_i$ : Kripke structure

$SPEC_i$: Conjunction of CTL formulas

In a general case :

- ACTL Property preservation $C_{i+1} \Rightarrow C_i$ (Grumberg/Long 91)
- ECTL Property preservation $C_i \Rightarrow C_{i+1}$ (Loiseau and al. 95)

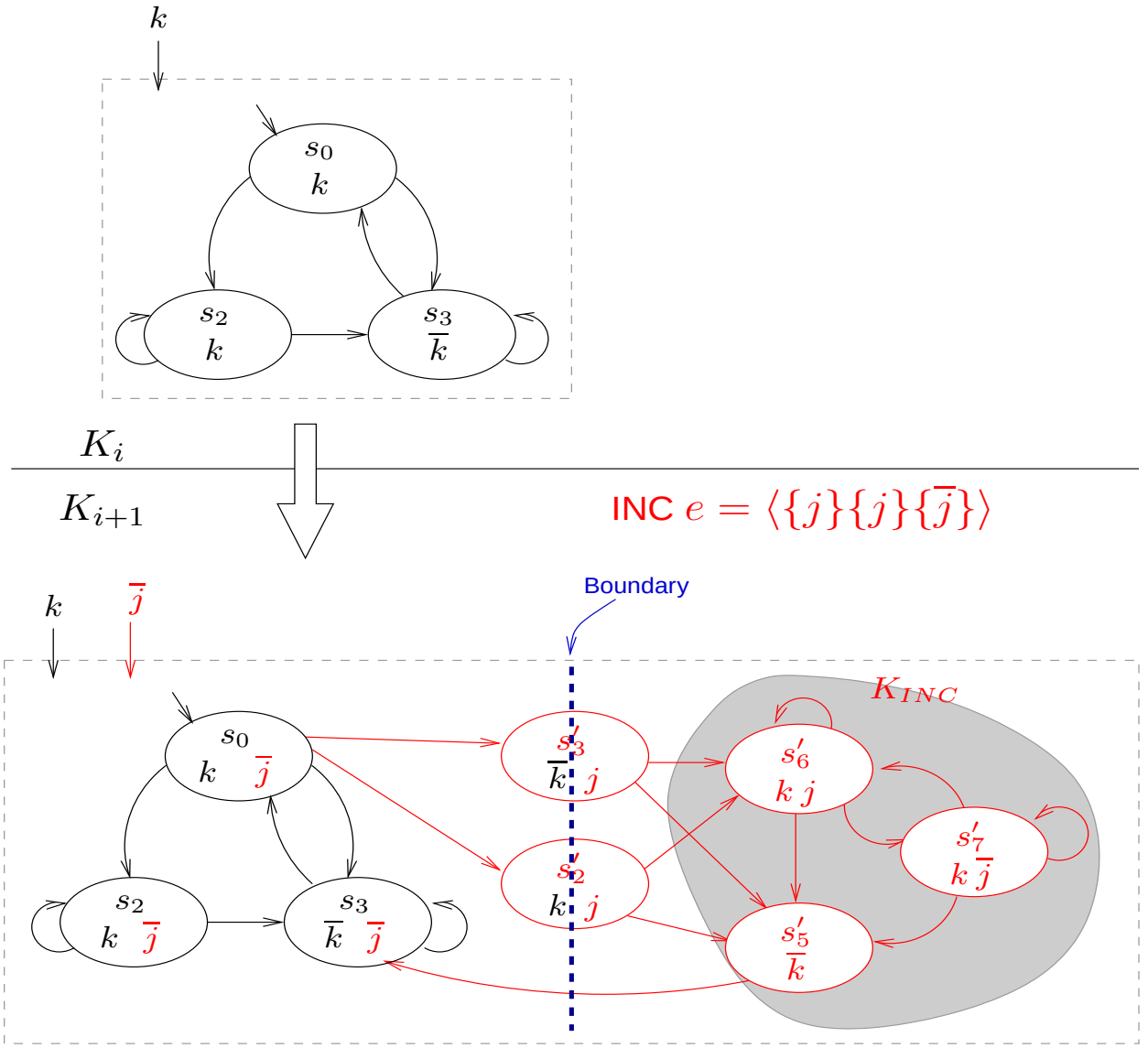
Incremental design :

- CTL Property transformation $C_i \Leftrightarrow C_{i+1}$ (Braunstein/Encrenaz 06)

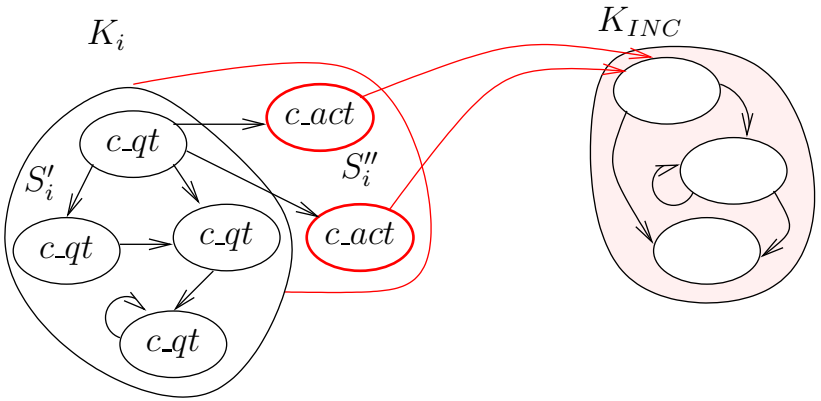
# Increment Definition

- Increment INC is a set of new events at the interface
  - Each event has quiet values and active values
  - No new initial state, No behaviour overriding
  - $K_{i+1}$ simulates $K_i$
- Increment $INC = \langle K_{INC}, R_{i \to INC}, R_{inc \to i} \rangle$
  - $K_{INC}$: increment's Kripke structure
  - $R_{i \to INC}$: connection between $K_i$ and $K_{INC}$
  - $R_{INC \to i}$: connection between $K_{INC}$ and $K_i$

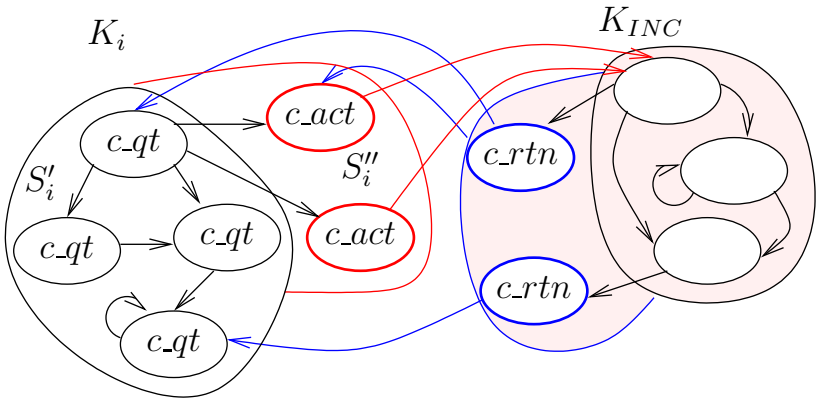
# Incremented structure $K_{i+1}$



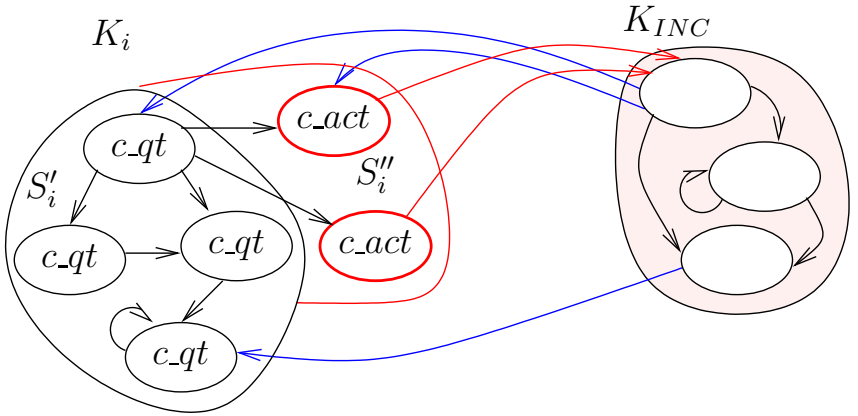
# Incorporation of the increment's specification




(a) Without return




(b) With a special return value




(c) Without special return value

# CTL-property transformations

(a) The specifi cation of $K_{INC}$ holds in $K_{i+1}$ as soon as the active value holds

$$K_{INC} \models \varphi \Rightarrow K_{i+1} \models \mathbf{A}(e\_qt\mathbf{W}(e\_act \wedge \mathbf{AX}\varphi))$$

(b) The specifi cation of $K_{INC}$ holds in $K_{i+1}$as soon as the active value holds and until the occurrence of a return value

$$K_{INC} \models \varphi \Rightarrow K_{i+1} \models \mathbf{A}(e\_qt\mathbf{W}(e\_act \wedge \mathbf{AX}\boxed{\varphi'}))$$

(c) Not enough characterisation of the return value but the "non-regression" rules still hold.

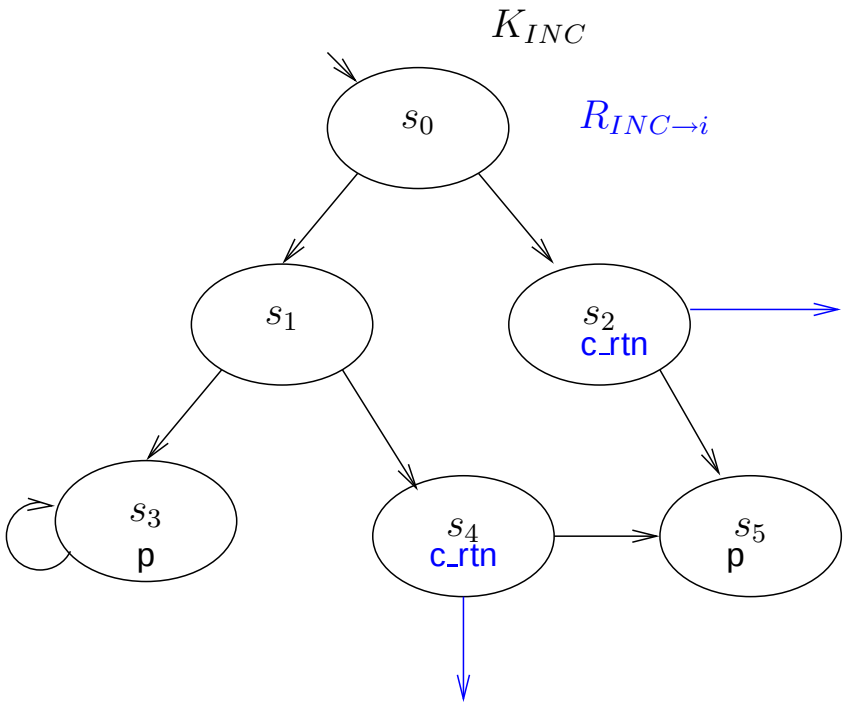
# Transformations of $\varphi'$

Inspired by the CTL-property transformations of $K_i$ (STTT06)

Principle : Reduction of the computational tree explored.

Example : $K_{INC} \models AFp$ ; Return value $c\_rtn$

$K'_{INC} \models AF(p \vee c\_rtn)$



# Concluding remarks

## Conclusion

- Extension of the incremental design process
- Automatic transformations of increment specifi cation
- Specifi cation of $K_{i+1}$ guarantied by construction
- Application to a concrete component design (VCI-PI protocol converter)

## Ongoing work

- Tool for automatic integration of increment
- Use of specifi cation as component abstraction