Journée MeFoSyLoMa – CosyVerif

Université Paris 13, Sorbonne Paris Cité

15th June 2012

IMITATOR

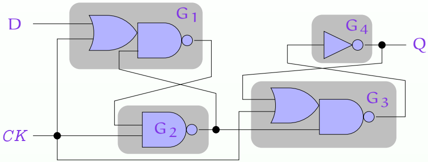
Inverse Method for Inferring Time AbstracT behaviOR

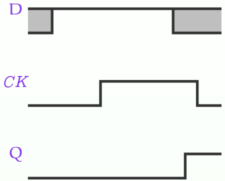
É. André[1], L. Fribourg[2], U. Kühne[3] and R. Soulat[2]

[1] Université Paris 13, Sorbonne Paris Cité, LIPN, CNRS, F-93430, Villetaneuse, France
[2] LSV, ENS de Cachan & CNRS, France
[3] Group for Computer Architecture, University of Bremen, Germany

An Example of Flip-Flop Circuit

- An asynchronous circuit [Clarisó and Cortadella, 2007]

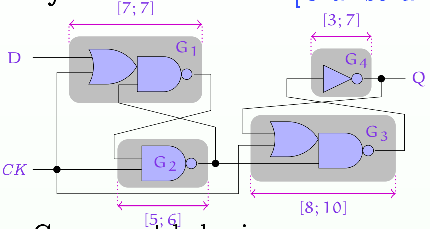


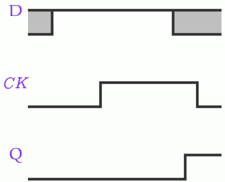


- Concurrent behavior
  - 4 elements: G_1, G_2, G_3, G_4
  - 2 input signals (D and CK), 1 output signal (Q)

An Example of Flip-Flop Circuit

- An asynchronous circuit [Clarisó and Cortadella, 2007]

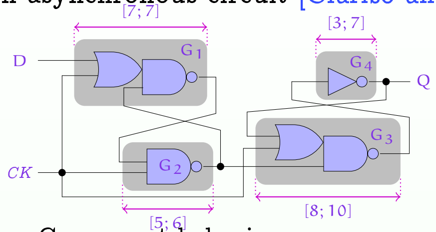


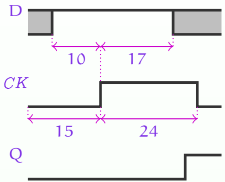


- Concurrent behavior
  - 4 elements: G_1, G_2, G_3, G_4
  - 2 input signals (D and CK), 1 output signal (Q)
- Timing delays
  - Traversal delays of the gates: one interval per gate

An Example of Flip-Flop Circuit

- An asynchronous circuit [Clarisó and Cortadella, 2007]

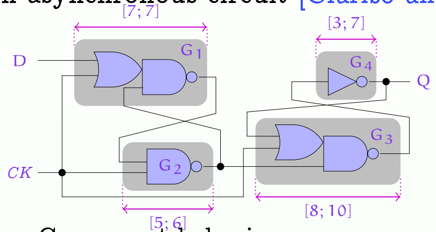


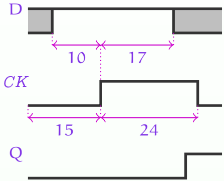


- Concurrent behavior
  - 4 elements: G_1, G_2, G_3, G_4
  - 2 input signals (D and CK), 1 output signal (Q)
- Timing delays
  - Traversal delays of the gates: one interval per gate
  - Environment timing constants

An Example of Flip-Flop Circuit

- An asynchronous circuit [Clarisó and Cortadella, 2007]



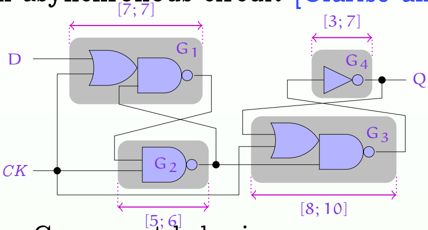


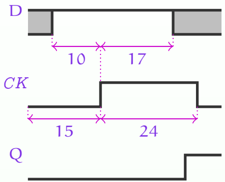
  - Concurrent behavior
    - 4 elements: G_1, G_2, G_3, G_4
    - 2 input signals (D and CK), 1 output signal (Q)
  - Timing delays
    - Traversal delays of the gates: one interval per gate
    - Environment timing constants

- Question
  - For these timing delays, does the rise of Q always occur before the fall of CK?

An Example of Flip-Flop Circuit

- An asynchronous circuit [Clarisó and Cortadella, 2007]





  - Concurrent behavior
    - 4 elements: G_1, G_2, G_3, G_4
    - 2 input signals (D and CK), 1 output signal (Q)
  - Timing delays
    - Traversal delays of the gates: one interval per gate
    - Environment timing constants

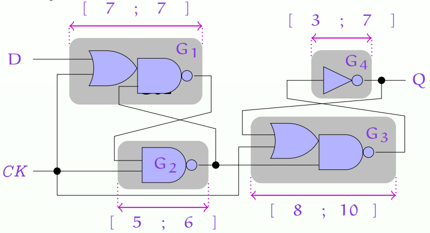
- Question
  - For these timing delays, does the rise of Q always occur before the fall of CK?
  - Timed model checking gives the answer: yes

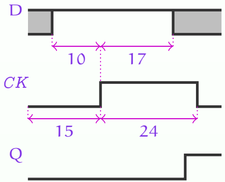
Synthesis of Parameters

- More difficult problem: find values of the timing delays for which the system behaves well
- Idea: reason with unknown constants or parameters
- Interesting applications
  - Ensure the robustness of the system
  - Allow the designer to optimize timing delays
- Difficult problem
  - Both concurrent behavior and timed behavior
  - Undecidable in general

Flip-Flop Circuit: Timing Parameters

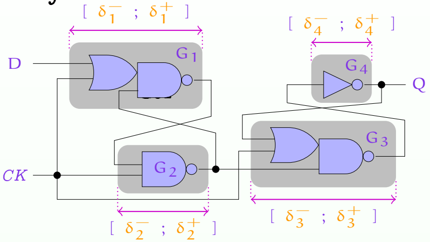
- An asynchronous circuit

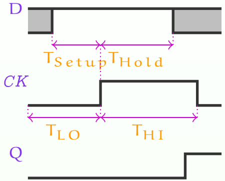




Flip-Flop Circuit: Timing Parameters

- An asynchronous circuit

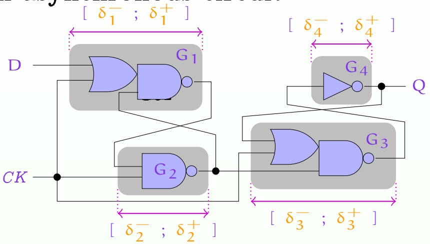


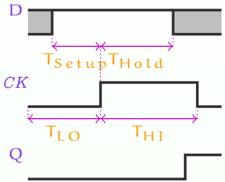


- Timing parameters
  - Traversal delays of the gates: one interval per gate
  - 4 environment parameters: T_{LO}, T_{HI}, T_{Setup} and T_{Hold}

Flip-Flop Circuit: Timing Parameters

- An asynchronous circuit



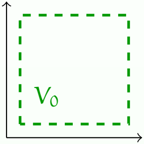


- Timing parameters
  - Traversal delays of the gates: one interval per gate
  - 4 environment parameters: T_{LO}, T_{HI}, T_{Setup} and T_{Hold}

- Question: for which values of the parameters does the rise of Q always occur before the fall of CK?

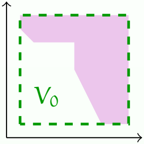
Problems

- The good parameters problem
  - "Given a bounded parameter domain V_0, find a set of parameter valuations of good behavior in V_0"



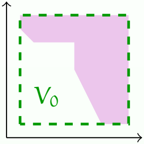
Problems

- The good parameters problem
  - “Given a bounded parameter domain V_0, find a set of parameter valuations of good behavior in V_0”

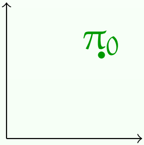


Problems

- The good parameters problem
  - "Given a bounded parameter domain V_0, find a set of parameter valuations of good behavior in V_0"

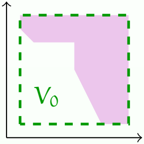


- The inverse problem: A simpler problem
  - "Given a reference parameter valuation π_0, find other valuations around π_0 of same behavior"

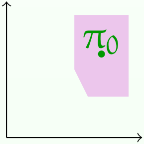


Problems

- The good parameters problem
  - "Given a bounded parameter domain V_0, find a set of parameter valuations of good behavior in V_0"



- The inverse problem: A simpler problem
  - "Given a reference parameter valuation π_0, find other valuations around π_0 of same behavior"



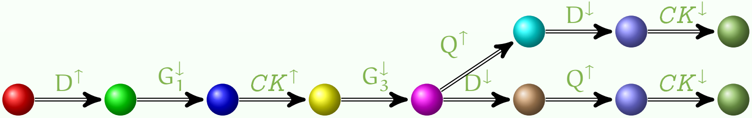
Outline

Outline

Functional view of IMITATOR

Trace Set

- Trace set: set of all traces of a PTA
- Graphical representation under the form of a tree
  - Does not give any information on the branching behavior though
  - Example of trace set for the flip-flop example

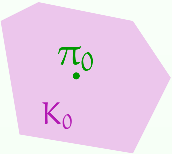


The Inverse Method

- Input
  - A PTA $\mathcal{A}$
  - A reference valuation π_0 of all the parameters of $\mathcal{A}$

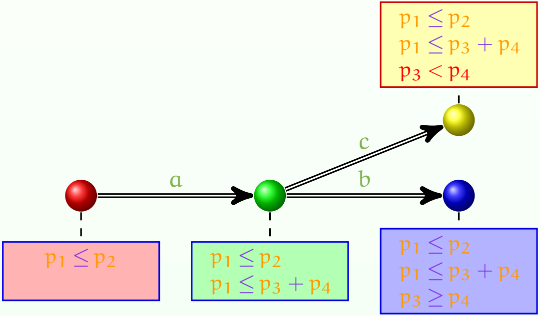
The Inverse Method

- Input
  - A PTA $\mathcal{A}$
  - A reference valuation π_0 of all the parameters of $\mathcal{A}$
- Output: tile K_0
  - Convex constraint on the parameters such that
    - $\pi_0 \models K_0$
    - For all points $\pi \models K_0$, $\mathcal{A}[\pi]$ and $\mathcal{A}[\pi_0]$ have the same trace sets



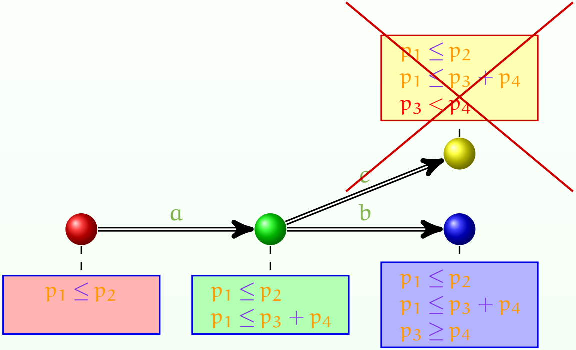
The Inverse Method: General Idea

- The idea [André et al., 2009]
  - Instead of negating bad states (as in "CEGAR" approaches), we remove π_0-incompatible states



The Inverse Method: General Idea

- The idea [André et al., 2009]
  - Instead of negating bad states (as in "CEGAR" approaches), we remove π_0-incompatible states



Application to the Flip-Flop Circuit

π_0 :

δ_1^-	$= 7$	δ_1^+	$= 7$	T_{HI}	$= 24$
δ_2^-	$= 5$	δ_2^+	$= 6$	T_{LO}	$= 15$
δ_3^-	$= 8$	δ_3^+	$= 10$	T_{Setup}	$= 10$
δ_4^-	$= 3$	δ_4^+	$= 7$	T_{Hold}	$= 17$

$K_0 = \texttt{true}$

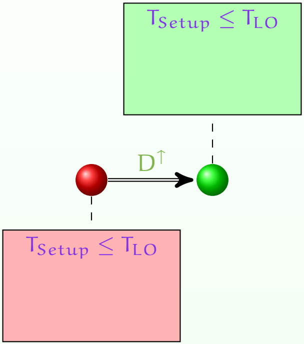
$T_{Setup} \leq T_{LO}$

Application to the Flip-Flop Circuit

π_0 :

$\delta_1^- = 7$	$\delta_1^+ = 7$	$T_{HI} = 24$
$\delta_2^- = 5$	$\delta_2^+ = 6$	$T_{LO} = 15$
$\delta_3^- = 8$	$\delta_3^+ = 10$	$T_{Setup} = 10$
$\delta_4^- = 3$	$\delta_4^+ = 7$	$T_{Hold} = 17$

$K_0 = \texttt{true}$

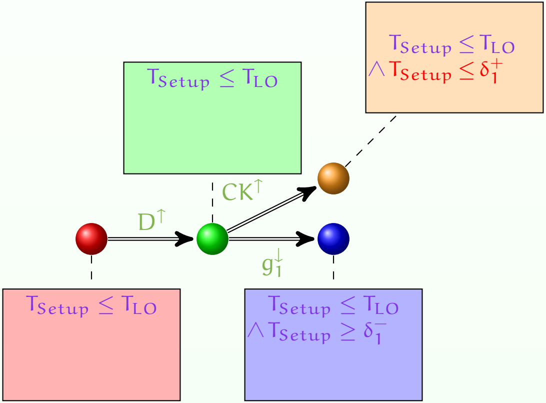


Application to the Flip-Flop Circuit

π_0 :

$\delta_1^- = 7$	$\delta_1^+ = 7$	$T_{HI} = 24$
$\delta_2^- = 5$	$\delta_2^+ = 6$	$T_{LO} = 15$
$\delta_3^- = 8$	$\delta_3^+ = 10$	$T_{Setup} = 10$
$\delta_4^- = 3$	$\delta_4^+ = 7$	$T_{Hold} = 17$

$K_0 = \mathtt{true}$



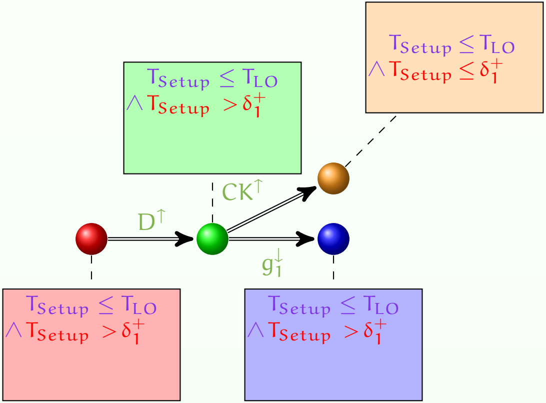
Application to the Flip-Flop Circuit

π_0 :

δ_1^-	$= 7$	δ_1^+	$= 7$	T_{HI}	$= 24$
δ_2^-	$= 5$	δ_2^+	$= 6$	T_{LO}	$= 15$
δ_3^-	$= 8$	δ_3^+	$= 10$	T_{Setup}	$= 10$
δ_4^-	$= 3$	δ_4^+	$= 7$	T_{Hold}	$= 17$

$K_0 =$

$T_{Setup} > \delta_1^+$



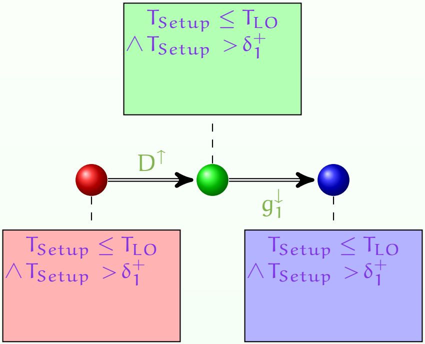
Application to the Flip-Flop Circuit

π_0 :

$\delta_1^- = 7$	$\delta_1^+ = 7$	$T_{HI} = 24$
$\delta_2^- = 5$	$\delta_2^+ = 6$	$T_{LO} = 15$
$\delta_3^- = 8$	$\delta_3^+ = 10$	$T_{Setup} = 10$
$\delta_4^- = 3$	$\delta_4^+ = 7$	$T_{Hold} = 17$

$K_0 =$

$T_{Setup} > \delta_1^+$



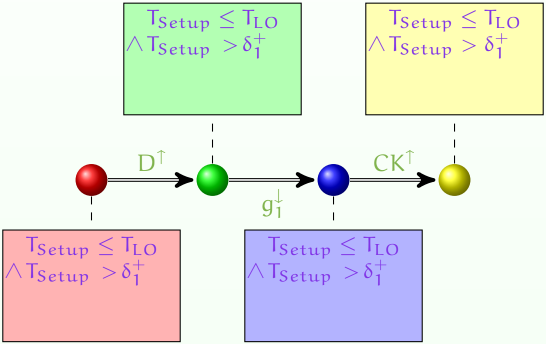
Application to the Flip-Flop Circuit

π_0 :

δ_1^-	$= 7$	δ_1^+	$= 7$	T_{HI}	$= 24$
δ_2^-	$= 5$	δ_2^+	$= 6$	T_{LO}	$= 15$
δ_3^-	$= 8$	δ_3^+	$= 10$	T_{Setup}	$= 10$
δ_4^-	$= 3$	δ_4^+	$= 7$	T_{Hold}	$= 17$

$K_0 =$

$T_{Setup} > \delta_1^+$



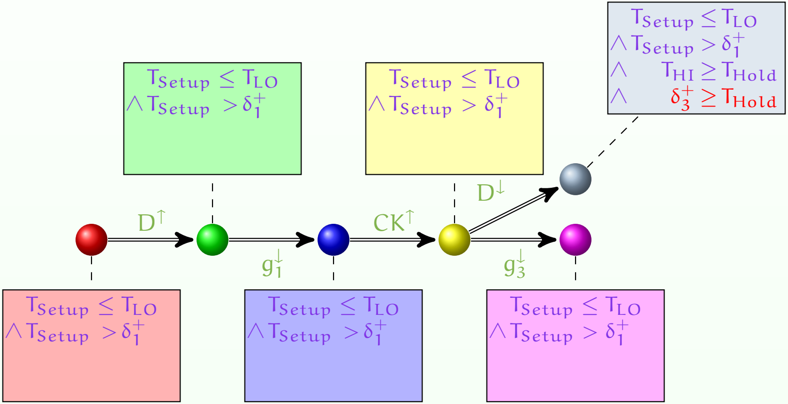
Application to the Flip-Flop Circuit

π_0 :

δ_1^-	$= 7$	δ_1^+	$= 7$	T_{HI}	$= 24$
δ_2^-	$= 5$	δ_2^+	$= 6$	T_{LO}	$= 15$
δ_3^-	$= 8$	δ_3^+	$= 10$	T_{Setup}	$= 10$
δ_4^-	$= 3$	δ_4^+	$= 7$	T_{Hold}	$= 17$

$K_0 =$

$T_{Setup} > \delta_1^+$



Application to the Flip-Flop Circuit

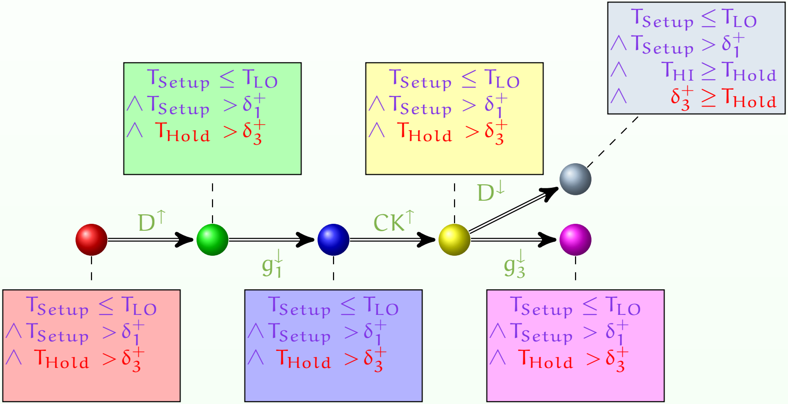
π_0 :

$\delta_1^- = 7$	$\delta_1^+ = 7$	$T_{HI} = 24$
$\delta_2^- = 5$	$\delta_2^+ = 6$	$T_{LO} = 15$
$\delta_3^- = 8$	$\delta_3^+ = 10$	$T_{Setup} = 10$
$\delta_4^- = 3$	$\delta_4^+ = 7$	$T_{Hold} = 17$

$K_0 =$

$T_{Setup} > \delta_1^+$

$\wedge\ T_{Hold} > \delta_3^+$

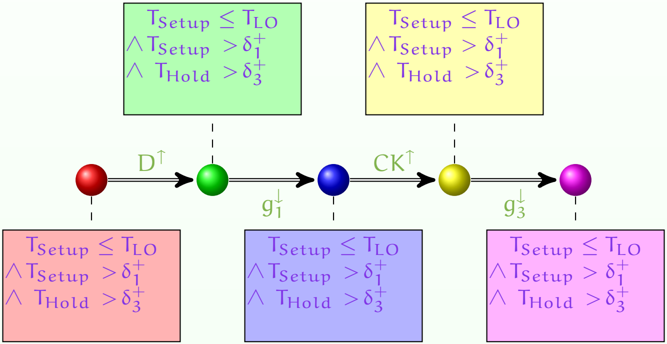


Application to the Flip-Flop Circuit

π_0 :

$\delta_1^- = 7$	$\delta_1^+ = 7$	$T_{HI} = 24$
$\delta_2^- = 5$	$\delta_2^+ = 6$	$T_{LO} = 15$
$\delta_3^- = 8$	$\delta_3^+ = 10$	$T_{Setup} = 10$
$\delta_4^- = 3$	$\delta_4^+ = 7$	$T_{Hold} = 17$

$K_0 =$
$T_{Setup} > \delta_1^+$
$\wedge\ T_{Hold} > \delta_3^+$

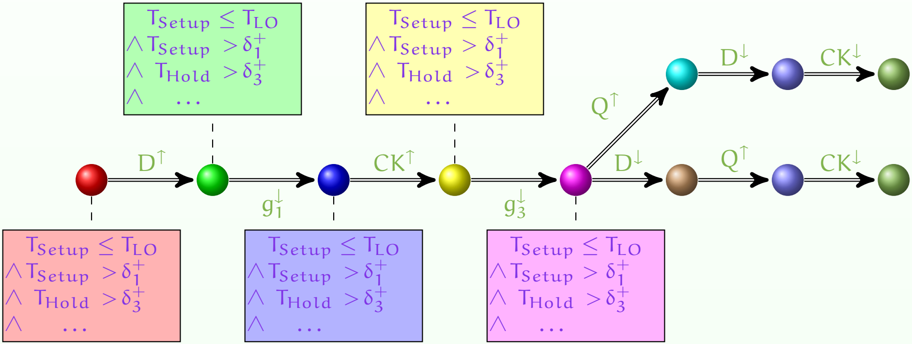


Application to the Flip-Flop Circuit

π_0 :

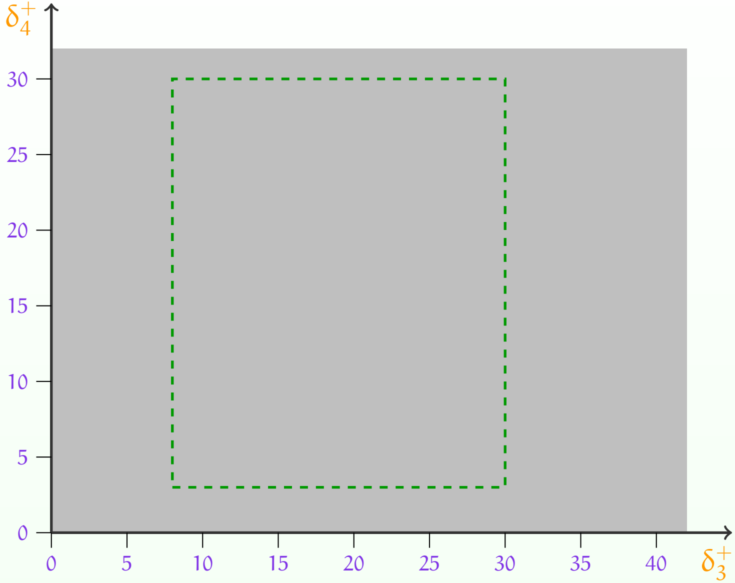
$\delta_1^- = 7$	$\delta_1^+ = 7$	$T_{HI} = 24$
$\delta_2^- = 5$	$\delta_2^+ = 6$	$T_{LO} = 15$
$\delta_3^- = 8$	$\delta_3^+ = 10$	$T_{Setup} = 10$
$\delta_4^- = 3$	$\delta_4^+ = 7$	$T_{Hold} = 17$

$$K_0 = \begin{array}{llll} & T_{Setup} > \delta_1^+ & \wedge & \delta_3^+ + \delta_4^+ \geq T_{Hold} \\ \wedge & T_{Hold} > \delta_3^+ & \wedge & \delta_3^+ + \delta_4^+ < T_{HI} \\ \wedge & T_{Setup} \leq T_{LO} & \wedge & \delta_3^- + \delta_4^- \leq T_{Hold} \\ \wedge & \delta_1^- > 0 & & \end{array}$$

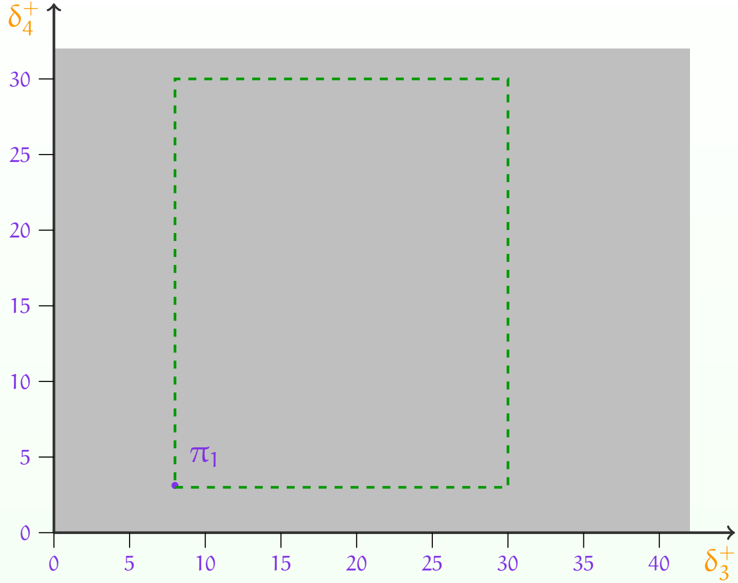


Outline

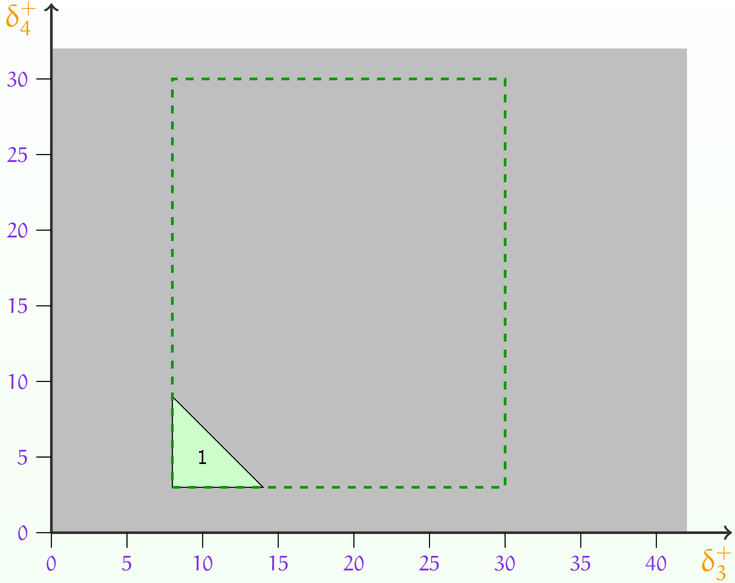
Behavioral Cartography of the Flip-Flop



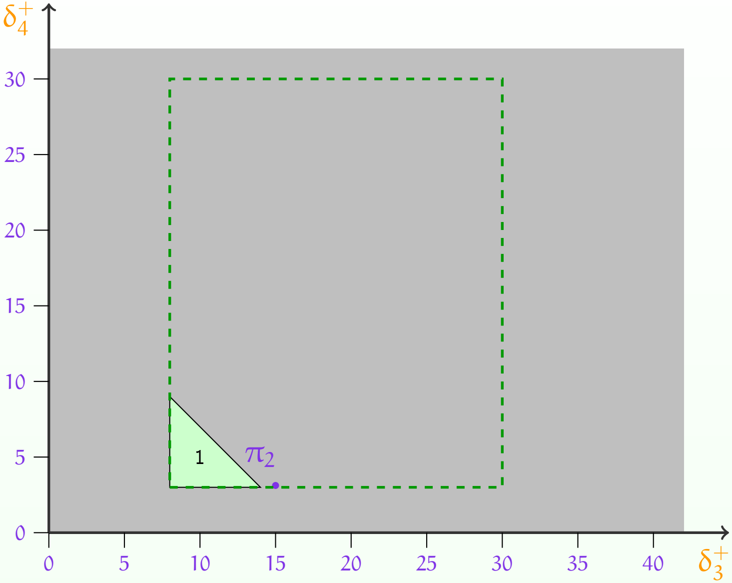
Behavioral Cartography of the Flip-Flop



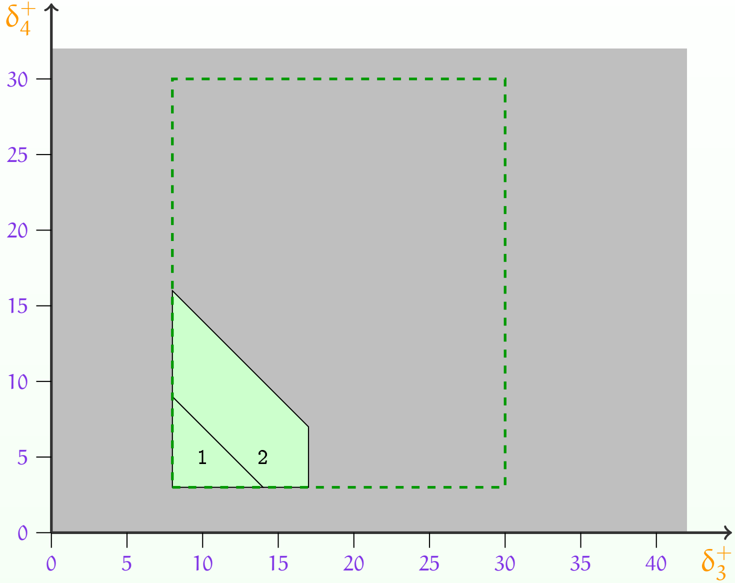
Behavioral Cartography of the Flip-Flop



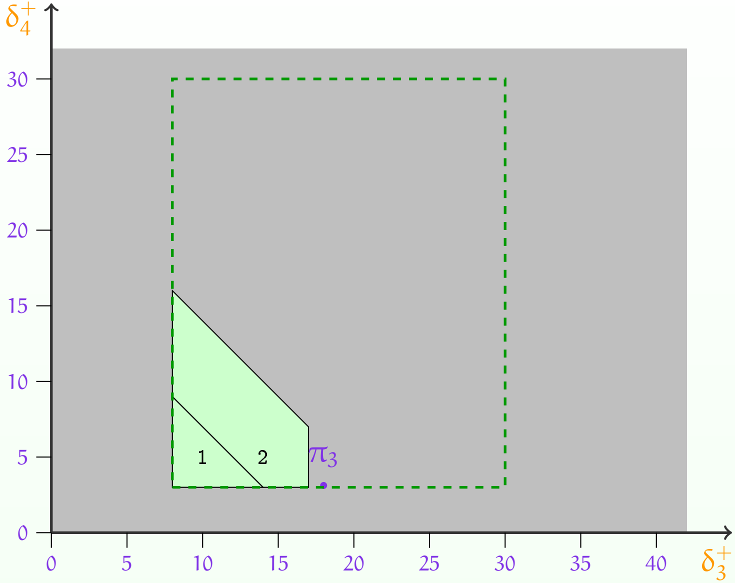
Behavioral Cartography of the Flip-Flop



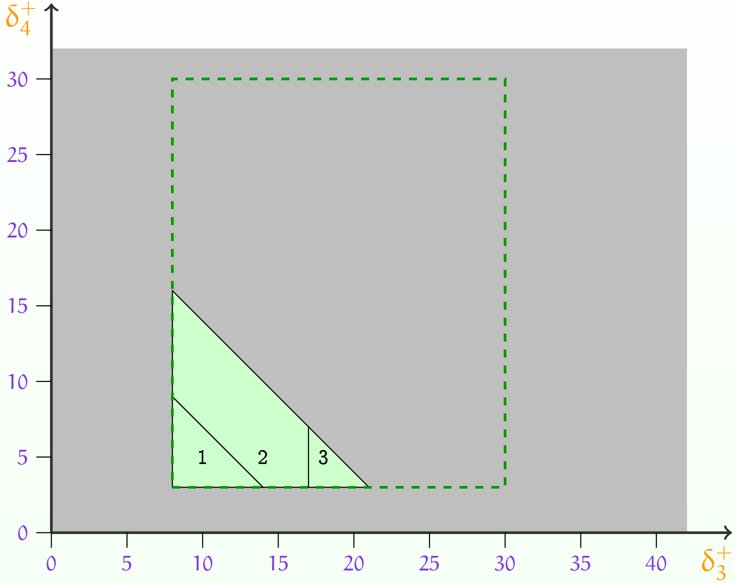
Behavioral Cartography of the Flip-Flop



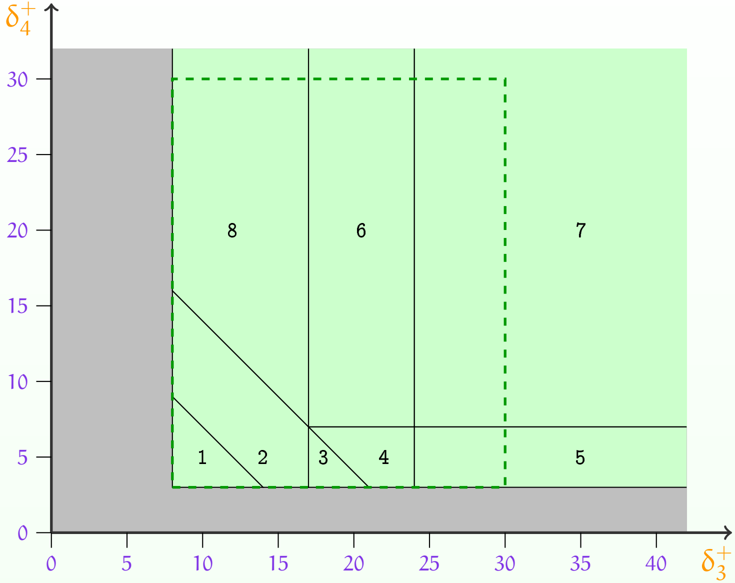
Behavioral Cartography of the Flip-Flop



Behavioral Cartography of the Flip-Flop

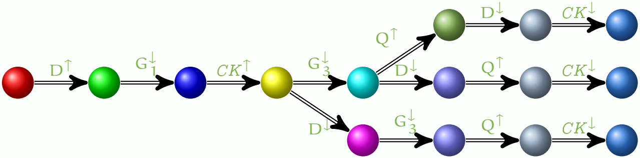


Behavioral Cartography of the Flip-Flop

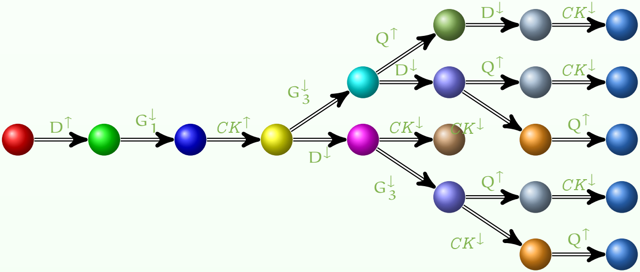


Examples of Good and Bad Tiles for the Flip-flop

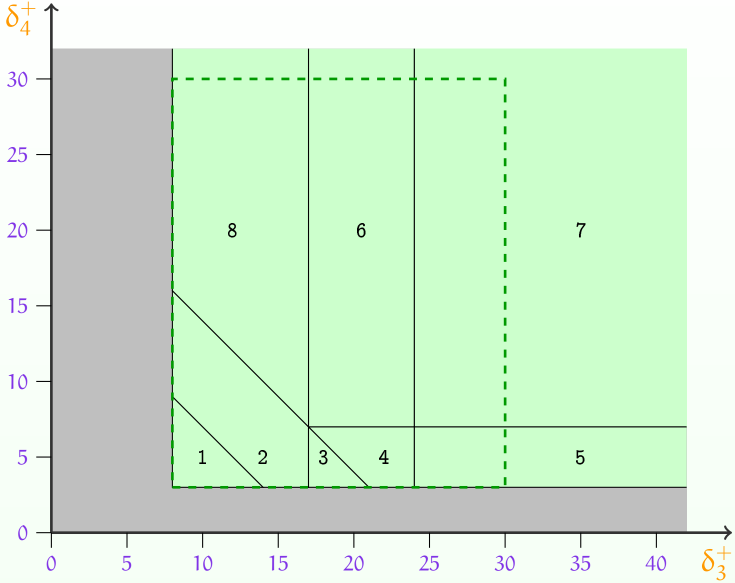
- Good tile 3



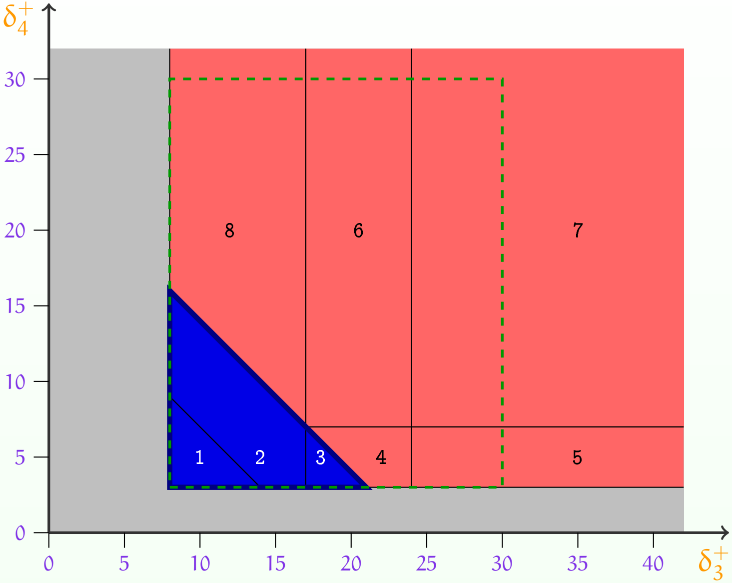
- Bad tile 7



Behavioral Cartography of the Flip-flop: Partition



Behavioral Cartography of the Flip-flop: Partition



Behavioral Cartography of the Flip-flop: Remarks

- Remarks on the cartography
  - For this example, all the real-valued part of the parametric space within and outside V_0 is covered
- The set of good tiles (in blue) corresponds to the maximal set of good values for δ_3^+ and δ_4^+
  - $\delta_3^+ + \delta_4^+ \leq 24 \quad \wedge \quad \delta_3^+ \geq 8 \quad \wedge \quad \delta_4^+ \geq 3$

Outline

Implementation

- IMITATOR 2.5 [André et al., 2012a]
  - "Inverse Method for Inferring Time AbstracT BehaviOR"
  - 10 000 lines of OCaml code
  - Makes use of the PPL library for handling polyhedra

- Main contributors
  - Étienne André
  - Laurent Fribourg
  - Ulrich Kühne
  - Romain Soulat

- Available on the Web
  - http://www.lsv.ens-cachan.fr/Software/imitator/

Implementation

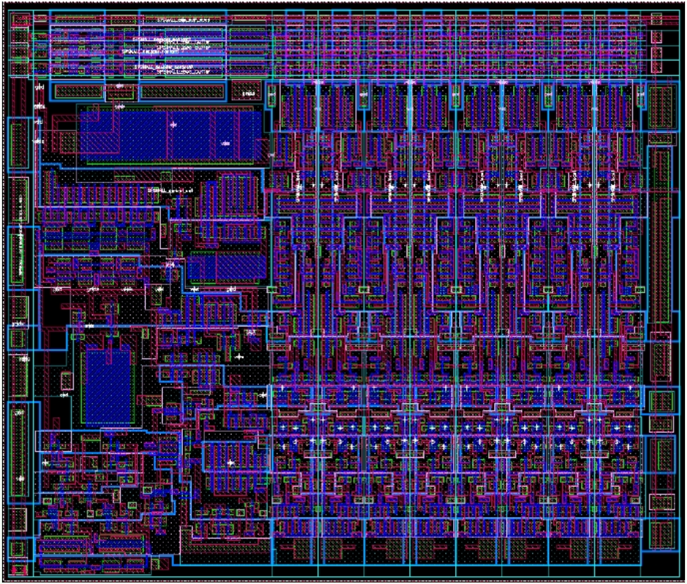
- IMITATOR 2.5 [André et al., 2012a]
    - "Inverse Method for Inferring Time AbstracT BehaviOR"
    - 10 000 lines of OCaml code
    - Makes use of the PPL library for handling polyhedra
- Main contributors
    - Étienne André
    - Laurent Fribourg
    - Ulrich Kühne
    - Romain Soulat
- Available on the Web
    - http://www.lsv.ens-cachan.fr/Software/imitator/
- And now part of CosyVerif!

Case Studies and Main Projects

- Applications
  - Asynchronous circuits
  - Communication protocols
  - Scheduling problems
- Industrial projects
  - ANR Valmem (with ST-Microelectronics) : 2007–2010 [André et al., 2009]
  - Farman ROSCOV (with EADS Astrium Space Transportation) : 2012–2013 [Fribourg et al., 2012]

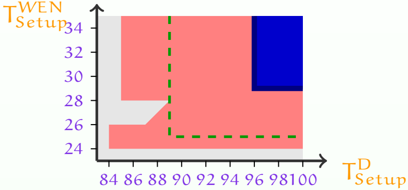
The SPSMALL Memory

The SPSMALL Memory



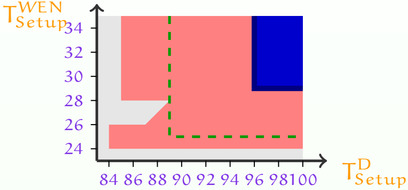
The SPSMALL Memory: Minimization of Timings

- Partition into good and bad tiles
  - Using the property of good behavior specified by the datasheet



The SPSMALL Memory: Minimization of Timings

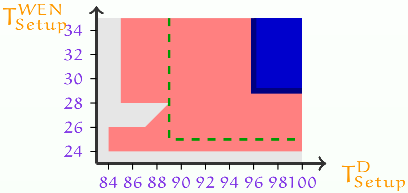
- Partition into good and bad tiles
  - Using the property of good behavior specified by the datasheet



- Minimization of timing delays
  - $T_{Setup}^{D} = 108$
  - $T_{Setup}^{WEN} = 48$

The SPSMALL Memory: Minimization of Timings

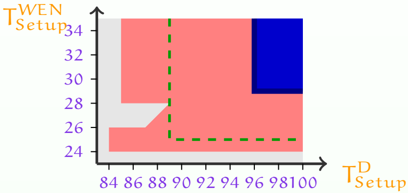
- Partition into good and bad tiles
  - Using the property of good behavior specified by the datasheet



- Minimization of timing delays
  - $T^{D}_{Setup} = 108 \rightsquigarrow 96$ (decrease of $11.1\,\%$)
  - $T^{WEN}_{Setup} = 48 \rightsquigarrow 29$ (decrease of $39.6\,\%$)

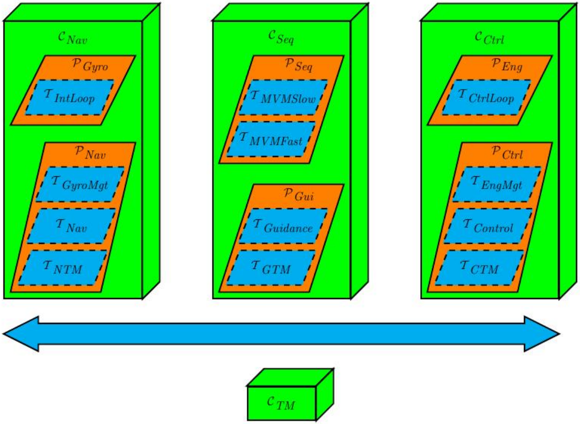
The SPSMALL Memory: Minimization of Timings

- Partition into good and bad tiles
  - Using the property of good behavior specified by the datasheet



- Minimization of timing delays
  - $T^{D}_{Setup} = 108 \rightsquigarrow 96$ (decrease of 11.1 %)
  - $T^{WEN}_{Setup} = 48 \rightsquigarrow 29$ (decrease of 39.6 %)
- Practical interest: allows to work in a faster environment
  - Optimization of the datasheet
  - Financial interest

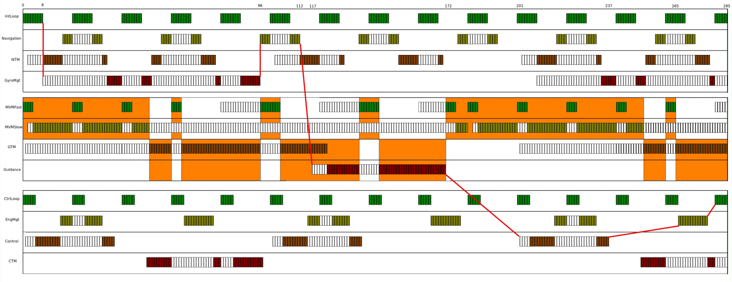
The ROSCOV Project with Astrium



Prospective architecture for the flight control system of the next generation of autonomous transfer vehicles (ATV)

The ROSCOV Project: Robust Scheduling

Robustness analysis for scheduling problems



- Use of IMITATOR to synthesize a constraint
  ⇝ Guarantee that the scheduling meets the deadline

References I

André, É. (2010).
An Inverse Method for the Synthesis of Timing Parameters in Concurrent Systems.
Thèse de doctorat, Laboratoire Spécification et Vérification, ENS Cachan, France.

André, É., Chatain, Th., Encrenaz, E., and Fribourg, L. (2009).
An inverse method for parametric timed automata.
International Journal of Foundations of Computer Science, 20(5):819–836.

André, É. and Fribourg, L. (2010).
Behavioral cartography of timed automata.
In *RP'10*, volume 6227 of *LNCS*, pages 76–90. Springer.

André, É., Fribourg, L., Kühne, U., and Soulat, R. (2012a).
IMITATOR 2.5: A tool for analyzing robustness in scheduling problems.
In *FM'12*, LNCS. Springer.
To appear.

André, É., Fribourg, L., and Soulat, R. (2012b).
Enhancing the inverse method with state merging.
In *NFM'12*, volume 7226 of *LNCS*, pages 100–105. Springer.

References II

André, É. and Kühne, U. (2012).
Parametric analysis of hybrid systems using HyMITATOR.
Tool paper and poster.

André, É. and Soulat, R. (2011).
Synthesis of timing parameters satisfying safety properties.
In *RP'11*, volume 6945 of *LNCS*, pages 31–44. Springer.

Clarisó, R. and Cortadella, J. (2007).
The octahedron abstract domain.
Sci. Comput. Program., 64(1):115–139.

Fribourg, L. and Kühne, U. (2011).
Parametric verification and test coverage for hybrid automata using the inverse method.
In *RP'11*, volume 6945 of *LNCS*, pages 191–204. Springer.

Fribourg, L., Lesens, D., Moro, P., and Soulat, R. (2012).
Robustness analysis for scheduling problems using the inverse method.
In *TIME'12*. IEEE Computer Society Press.
To appear.