Real-Time Systems

Lecture 07: DC Implementables

2014-06-03

Dr. Bernd Westphal

Albert-Ludwigs-Universität Freiburg, Germany

Contents & Goals

Last Lectures:

- Semantical Correctness Proof

This Lecture:

- **Educational Objectives:** Capabilities for following tasks/questions.
  - What does this standard forms mean? Give a satisfying interpretation.
  - What are implementables? What is a control automaton?
  - Please specify (and prove correct) a controller which satisfies this requirement.

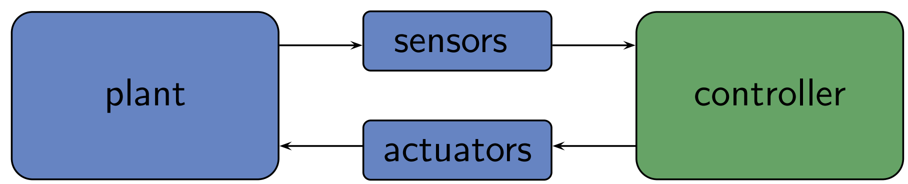
- **Content:**
  - DC Standard Forms
  - Control Automata
  - DC Implementables
  - Example

DC Implementables

Requirements vs. Implementations

- **Problem:** in general, a DC requirement doesn't tell **how** to achieve it, how to build a controller/write a program which ensures it.
- What a controller (clearly) can do is:
  - consider inputs now,
  - change (local) state, or
  - wait,
  - set outputs now.

  (But not, e.g., consider future inputs now.)



- So, if we have
  - a DC requirement **'**Req**'**,
  - a description **'**Impl**'** in DC, which "uses" **just these** operations,

  then
  - proving correctness amounts to proving $\models_0 \text{Impl} \implies \text{Req}$ (**in DC**)
  - and we (more or less) know how to program (the correct) **'Impl'** in a PLC language, or in C on a real-time OS, or or or. . .

Approach: Control Automata and DC Impl'bles

Plan:

- Introduce **DC Standard Forms**
- Introduce **Control Automata**
- Introduce **DC Implementables** as subset of **DC Standard Forms**
- Example: a correct controller design for the notorious Gas Burner

DC Standard Forms: Followed-by

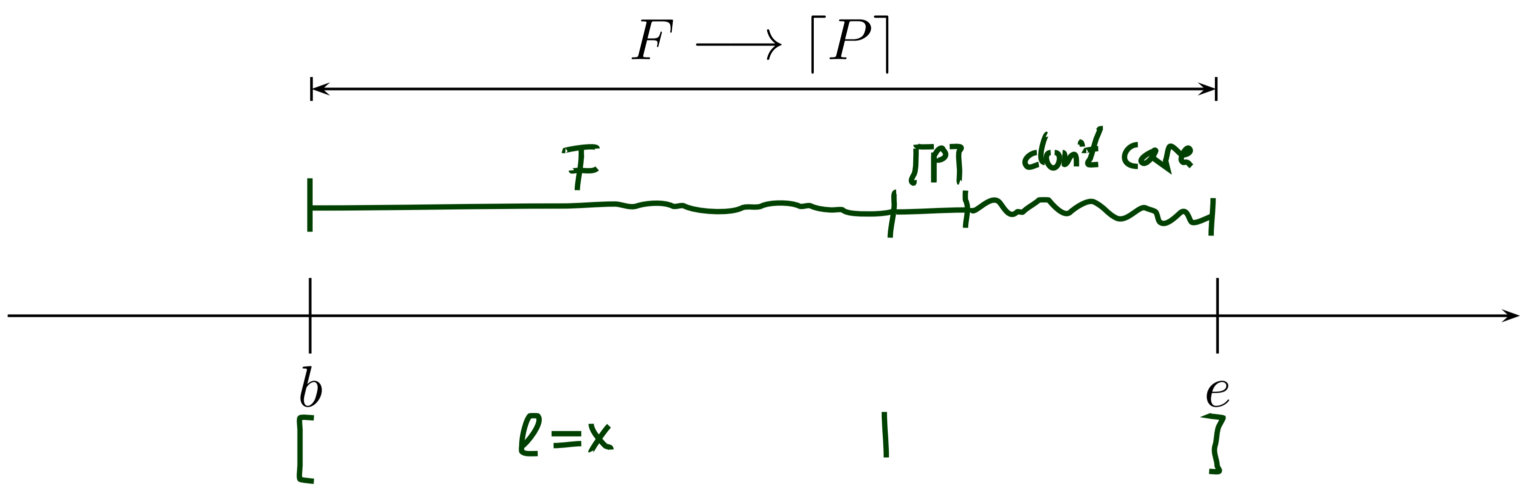
In the following: F is a DC **formula**, P a **state assertion**, θ a **rigid term**.

- **Followed-by:**

$$F \longrightarrow \lceil P \rceil :\iff \neg\Diamond(F\,;\lceil\neg P\rceil) \iff \Box\neg(F\,;\lceil\neg P\rceil)$$

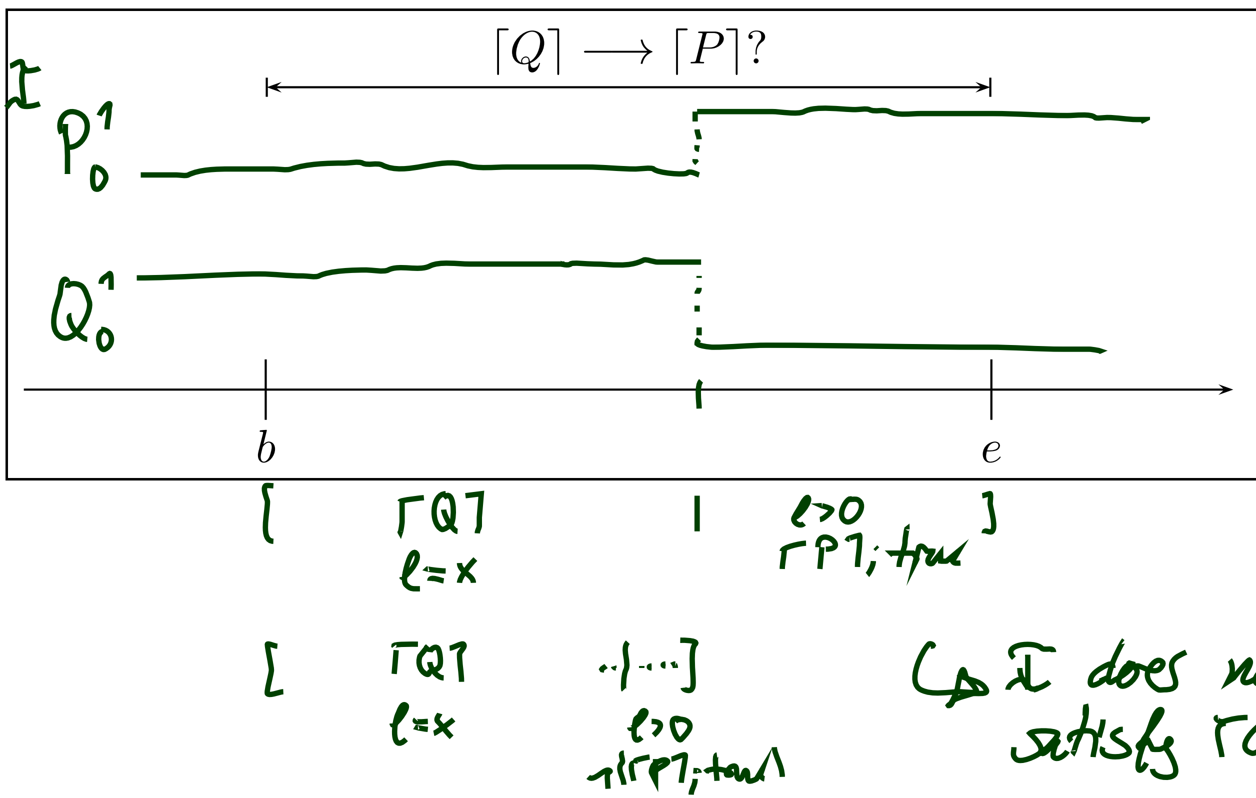
in other symbols

$$\forall\, x \bullet \Box((F \wedge \ell = x)\,;\ell > 0) \implies ((F \wedge \ell = x)\,;\lceil P\rceil\,;\mathit{true})$$



DC Standard Forms: Followed-by Examples

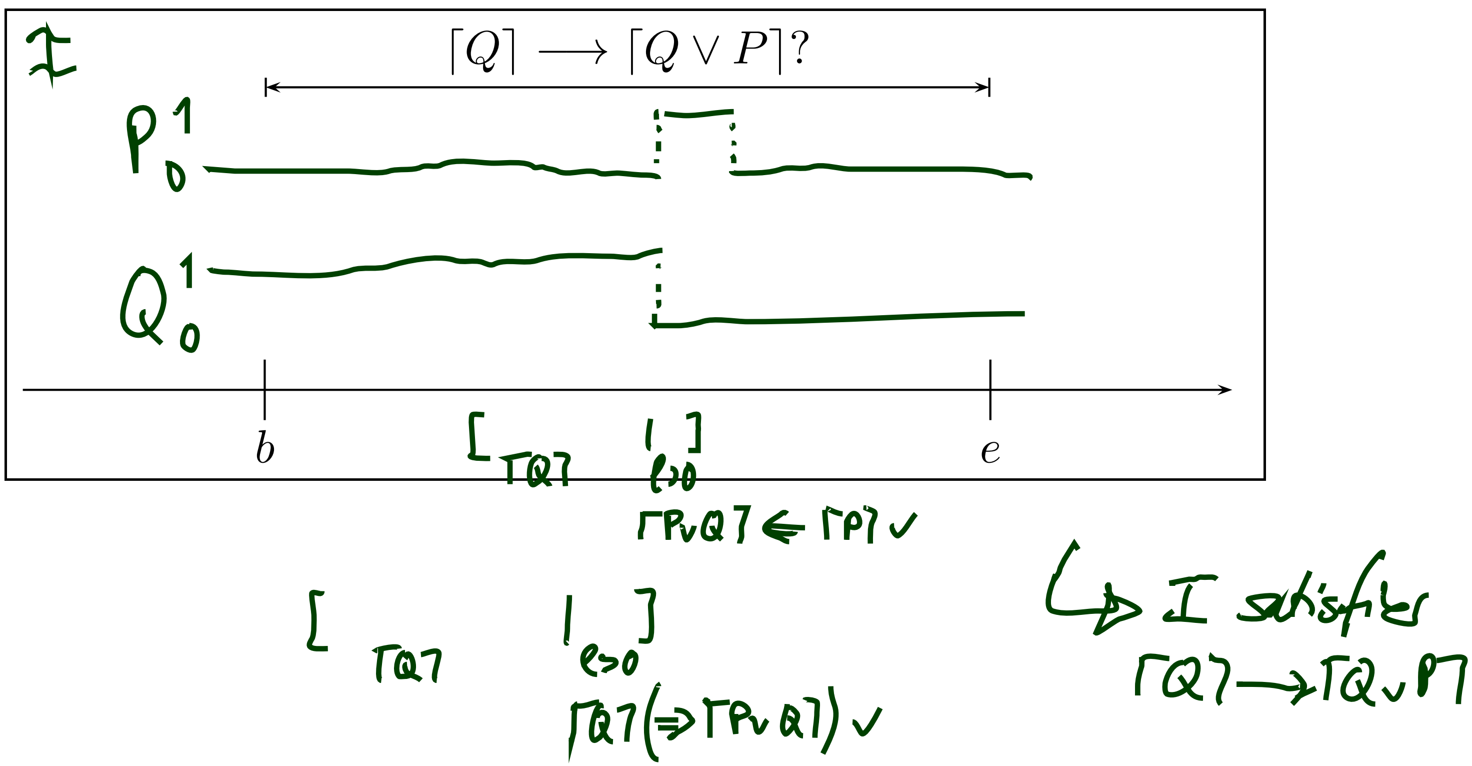
$$\forall x \bullet \Box((F \land \ell = x) \,;\, \ell > 0 \implies (F \land \ell = x) \,;\, \lceil P \rceil \,;\, \mathit{true})$$



↳ $\mathcal{I}$ does not satisfy $\lceil Q \rceil \longrightarrow \lceil P \rceil$

DC Standard Forms: Followed-by Examples

$$\forall x \bullet \Box((F \wedge \ell = x) \,;\, \ell > 0 \implies (F \wedge \ell = x) \,;\, \lceil P \rceil \,;\, \mathit{true})$$

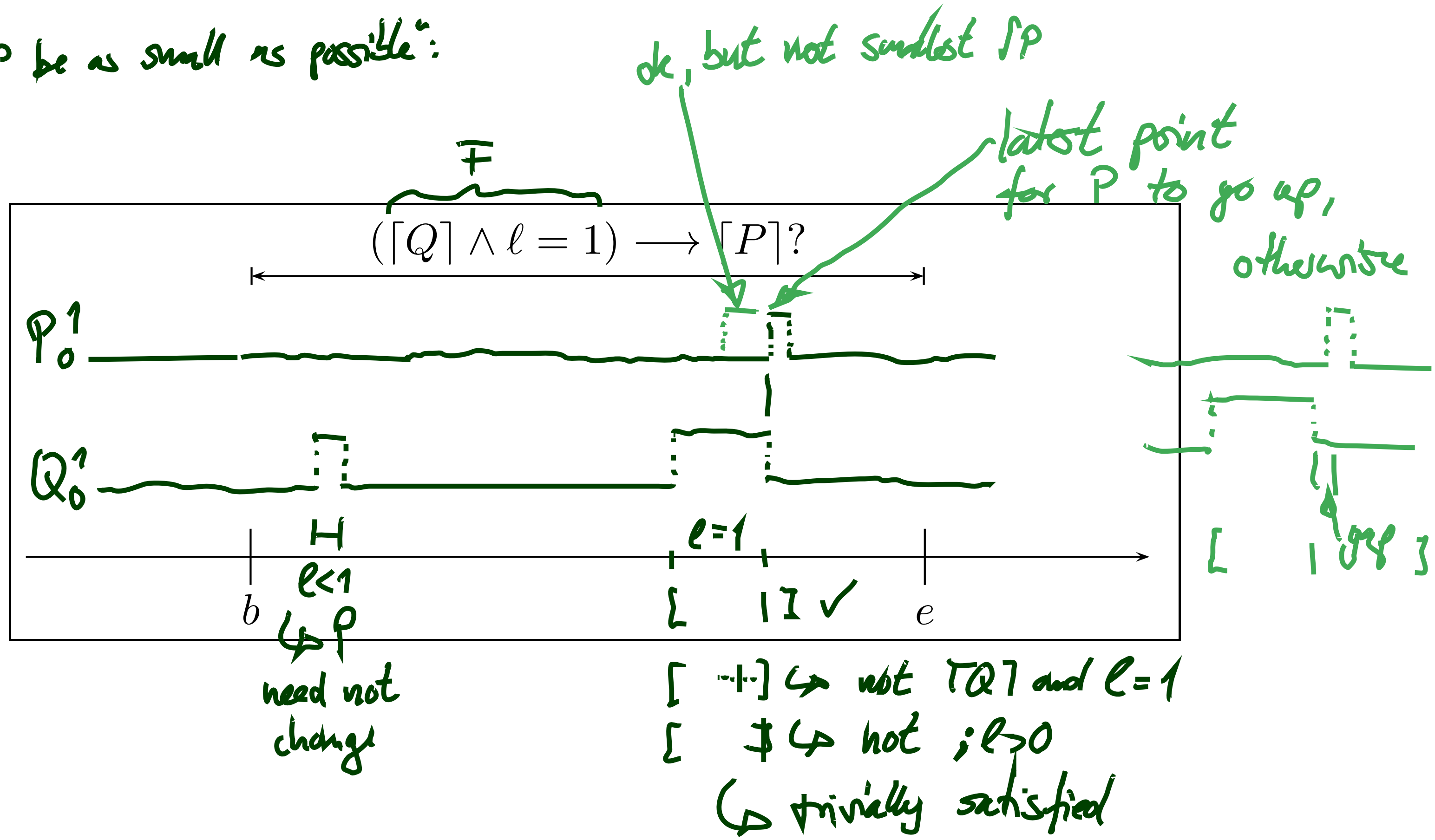


DC Standard Forms: Followed-by Examples

$$\forall x \bullet \Box((F \wedge \ell = x) ; \ell > 0 \implies (F \wedge \ell = x) ; \lceil P \rceil ; true)$$

... $\lceil Q \rceil \wedge \ell = 1 \wedge \ell = x$

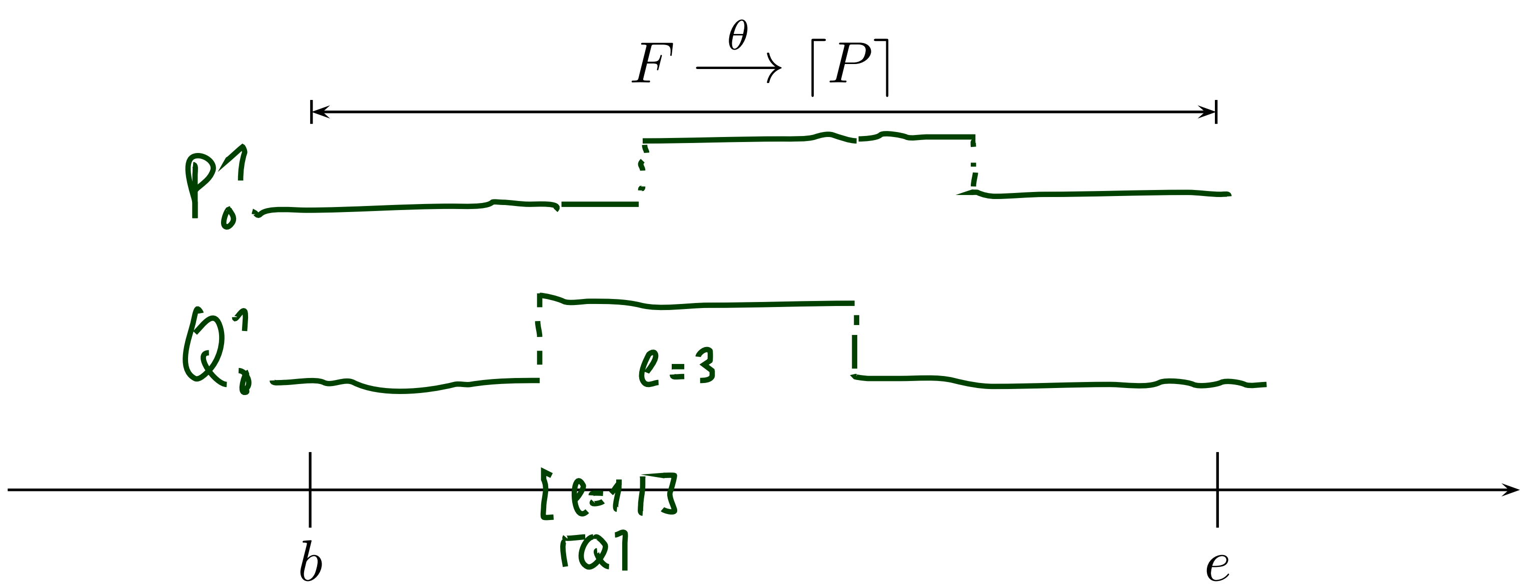
"let $\int P$ be as small as possible":



DC Standard Forms: (Timed) leads-to

- **(Timed) leads-to:**

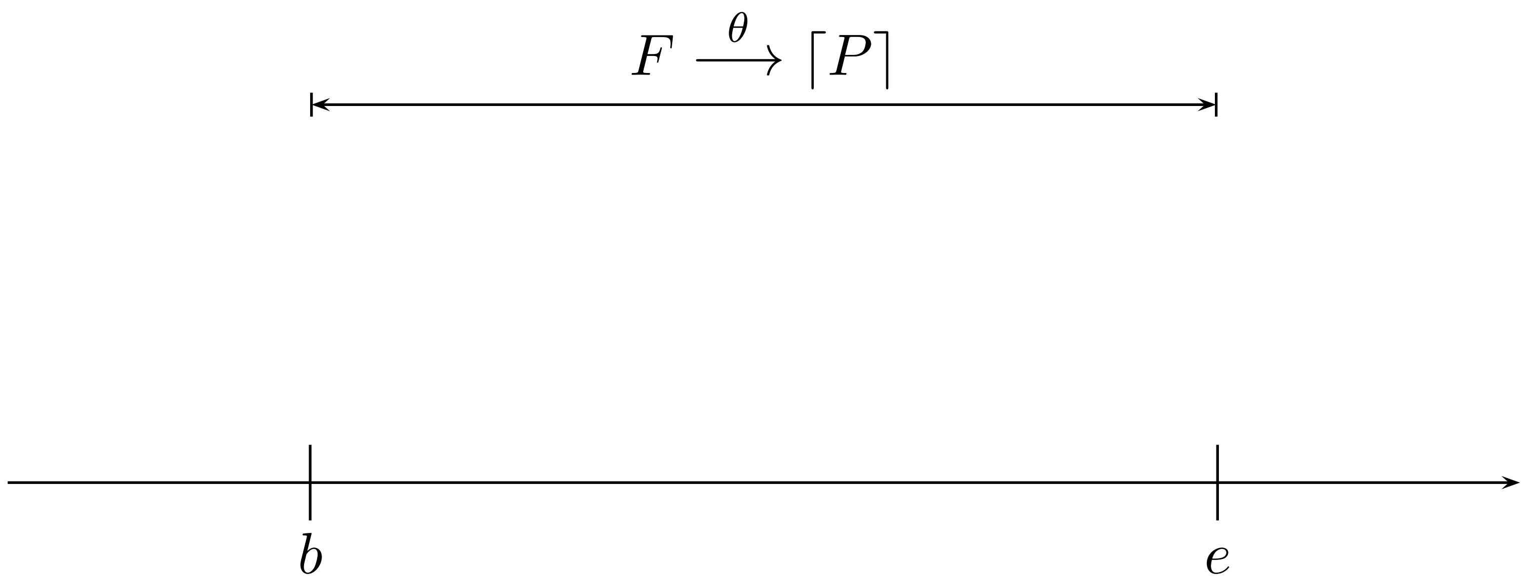
$$F \xrightarrow{\theta} \lceil P \rceil :\Longleftrightarrow (F \wedge \ell = \theta) \longrightarrow \lceil P \rceil$$



DC Standard Forms: (Timed) up-to

- **(Timed) up-to:**

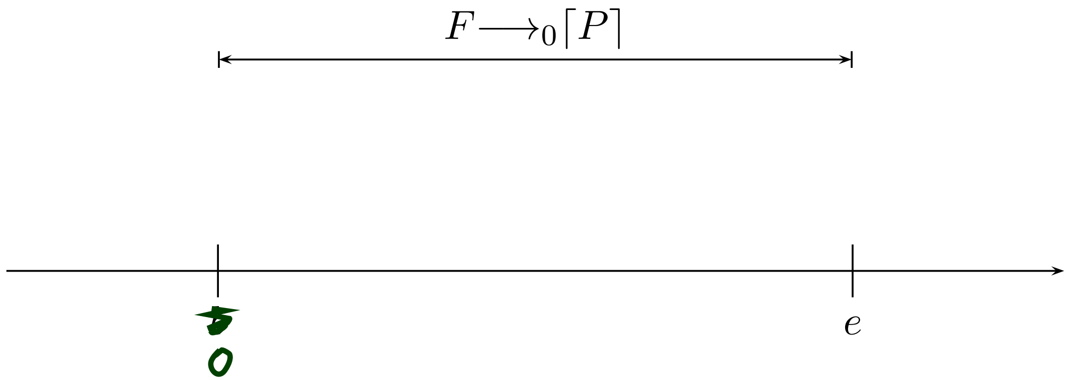
$$F \xrightarrow{\leq \theta} \lceil P \rceil :\Longleftrightarrow (F \wedge \ell \leq \theta) \longrightarrow \lceil P \rceil$$



DC Standard Forms: Initialisation

- **Followed-by-initially:**

$$F \longrightarrow_0 \lceil P \rceil :\Longleftrightarrow \neg(F \,;\, \lceil \neg P \rceil)$$



- **(Timed) up-to-initially:**

$$F \xrightarrow{\leq\theta}_0 \lceil P \rceil :\Longleftrightarrow (F \wedge \ell \leq \theta) \longrightarrow_0 \lceil P \rceil$$

- **Initialisation:**

$$\lceil\rceil \vee \lceil P \rceil \,;\, true$$

Control Automata

- Let $X_1, \ldots, X_k$ be k state variables ranging over **finite** domains $\mathcal{D}(X_1), \ldots, \mathcal{D}(X_k)$.
- With a DC formula 'Impl' ranging over $X_1, \ldots, X_k$ we have a **system of k control automata**.
- 'Impl' is typically a conjunction of **DC implementables**.
- A state assertion of the form

$$X_i = d_i, \quad d_i \in \mathcal{D}(X_i),$$

which constrains the values of X_i, is called **basic phase** of X_i.
- A **phase** of X_i is a Boolean combination of basic phases of X_i.

- **Abbreviations:**
  - Write X_i instead of $X_i = 1$, if X_i is Boolean.
  - Write d_i instead of $X_i = d_i$, if $\mathcal{D}(X_i)$ is disjoint from $\mathcal{D}(X_j)$, $i \neq j$.

Examples: $T = y \vee T = g$

basic phase

phase

$T = g \wedge B = p$

basic phase

not a phase, different observables!

Control Automata: Example

Model of Gas Burner controller as a system of four control automata:

- H Boolean,
  representing **heat request**, (input)
- F Boolean,
  representing **flame**, (input)
- C with $\mathcal{D}(C) = \{\text{idle}, \text{purge}, \text{ignite}, \text{burn}\}$,
  representing the (status of the) **controller**, *new!* (local)
- G Boolean,
  representing **gas valve**. (output)

controller reads inputs

sets its local status accordingly and writes outputs

- **Basic phase** of C:

$$C = \text{purge} \qquad \text{(or only: purge)}$$

- **Phase** of C:

$$\text{purge} \vee \text{idle}$$

DC Implementables

- DC Implementables
  are special patterns of DC Standard Forms (due to A.P. Ravn).
- Within one pattern,
  - $\pi, \pi_1, \ldots, \pi_n$, $n \geq 0$, denote **phases** of **the same** state variable X_i,
  - φ denotes a state assertion not depending on X_i.
- θ denotes a **rigid** term.

- **Initialisation**:

$$\lceil\rceil \vee \lceil\pi\rceil \, ; \, true$$

- **Sequencing**:

$$\lceil\pi\rceil \longrightarrow \lceil\pi \vee \pi_1 \vee \cdots \vee \pi_n\rceil$$

- **Progress**:

$$\lceil\pi\rceil \xrightarrow{\theta} \lceil\neg\pi\rceil$$

DC Implementables Cont'd

- **Bounded Stability**:

$$\lceil \neg\pi \rceil \,;\, \lceil \pi \wedge \varphi \rceil \xrightarrow{\leq\theta} \lceil \pi \vee \pi_1 \vee \cdots \vee \pi_n \rceil$$

- **Unbounded Stability**:

$$\lceil \neg\pi \rceil \,;\, \lceil \pi \wedge \varphi \rceil \longrightarrow \lceil \pi \vee \pi_1 \vee \cdots \vee \pi_n \rceil$$

- **Bounded initial stability**:

$$\lceil \pi \wedge \varphi \rceil \xrightarrow{\leq\theta}_0 \lceil \pi \vee \pi_1 \vee \cdots \vee \pi_n \rceil$$

- **Unbounded initial stability**:

$$\lceil \pi \wedge \varphi \rceil \longrightarrow_0 \lceil \pi \vee \pi_1 \vee \cdots \vee \pi_n \rceil$$

Specification by DC Implementables

- Let $X_1, \dots, X_k$ be a system of k control automata.
- Let 'Impl' be a conjunction of **DC implementables**.
- Then 'Impl' **specifies** all interpretations $\mathcal{I}$ of $X_1, \dots, X_k$ and all valuations $\mathcal{V}$ such that

$$\mathcal{I}, \mathcal{V} \models_0 \mathsf{Impl}$$

- Hmm: And what does this have to do with controllers...?

Example: Gas Burner

Recall: Control Automata

Model of Gas Burner controller as a system of four control automata:

- H : Boolean,
  representing **heat request**, (input)
- F : Boolean,
  representing **flame**, (input)
- C with $\mathcal{D}(C) = \{\text{idle}, \text{purge}, \text{ignite}, \text{burn}\}$,
  representing the **controller**, (local)
- G : Boolean,
  representing **gas valve**. (output)

Gas Burner Controller Specification

$\lceil\rceil \vee \lceil \mathsf{idle} \rceil \,;\, true, \quad \lceil\rceil \vee \lceil \neg H \rceil \,;\, true, \quad \lceil\rceil \vee \lceil \neg F \rceil \,;\, true, \quad \lceil\rceil \vee \lceil \neg G \rceil \,;\, true$	(Init-1 - 4)
$\lceil \mathsf{idle} \rceil \longrightarrow \lceil \mathsf{idle} \vee \mathsf{purge} \rceil$	(Seq-1)
$\lceil \mathsf{purge} \rceil \longrightarrow \lceil \mathsf{purge} \vee \mathsf{ignite} \rceil$	(Seq-2)
$\lceil \mathsf{ignite} \rceil \longrightarrow \lceil \mathsf{ignite} \vee \mathsf{burn} \rceil$	(Seq-3)
$\lceil \mathsf{burn} \rceil \longrightarrow \lceil \mathsf{burn} \vee \mathsf{idle} \rceil$	(Seq-4)
$\lceil \mathsf{purge} \rceil \xrightarrow{30+\varepsilon} \lceil \neg \mathsf{purge} \rceil$	(Prog-1)
$\lceil \mathsf{ignite} \rceil \xrightarrow{0.5+\varepsilon} \lceil \neg \mathsf{ignite} \rceil$	(Prog-2)
$\lceil \mathsf{idle} \wedge H \rceil \xrightarrow{\varepsilon} \lceil \neg \mathsf{idle} \rceil$	(Syn-1)
$\lceil \mathsf{burn} \wedge (\neg H \vee \neg F) \rceil \xrightarrow{\varepsilon} \lceil \neg \mathsf{burn} \rceil$	(Syn-2)
$\lceil G \wedge (\mathsf{idle} \vee \mathsf{purge}) \rceil \xrightarrow{\varepsilon} \lceil \neg G \rceil$	(Syn-3)
$\lceil \neg G \wedge (\mathsf{ignite} \vee \mathsf{burn}) \rceil \xrightarrow{\varepsilon} \lceil G \rceil$	(Syn-4)
$\lceil \neg \mathsf{idle} \rceil \,;\, \lceil \mathsf{idle} \wedge \neg H \rceil \longrightarrow \lceil \mathsf{idle} \rceil$	(Stab-1)
$\lceil \mathsf{idle} \wedge \neg H \rceil \longrightarrow_0 \lceil \mathsf{idle} \rceil$	(Stab-1-init)
$\lceil \neg \mathsf{purge} \rceil \,;\, \lceil \mathsf{purge} \rceil \xrightarrow{\leq 30} \lceil \mathsf{purge} \rceil$	(Stab-2)
$\lceil \neg \mathsf{ignite} \rceil \,;\, \lceil \mathsf{ignite} \rceil \xrightarrow{\leq 0.5} \lceil \mathsf{ignite} \rceil$	(Stab-3)
$\lceil \neg \mathsf{burn} \rceil \,;\, \lceil \mathsf{burn} \wedge H \wedge F \rceil \longrightarrow \lceil \mathsf{burn} \rceil$	(Stab-4)
$\lceil F \rceil \,;\, \lceil \neg F \wedge \neg \mathsf{ignite} \rceil \longrightarrow \lceil \neg F \rceil$	(Stab-5)
$\lceil \neg F \wedge \neg \mathsf{ignite} \rceil \longrightarrow_0 \lceil \neg F \rceil$	(Stab-5-init)
$\lceil G \rceil \,;\, \lceil \neg G \wedge (\mathsf{idle} \vee \mathsf{purge}) \rceil \longrightarrow \lceil \neg G \rceil$	(Stab-6)
$\lceil \neg G \wedge (\mathsf{idle} \vee \mathsf{purge}) \rceil \longrightarrow_0 \lceil \neg G \rceil$	(Stab-6-init)
$\lceil \neg G \rceil \,;\, \lceil G \wedge (\mathsf{ignite} \vee \mathsf{burn}) \rceil \longrightarrow \lceil G \rceil$	(Stab-7)

timing

input effect

link local status to output

env. assumption

Gas Burner Controller Specification: Untimed

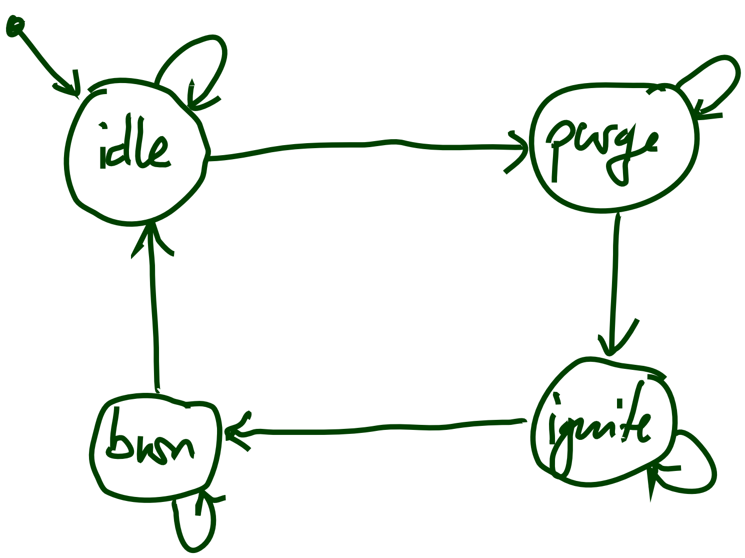
$$\lceil\rceil \vee \lceil \mathsf{idle} \rceil \,;\, true \qquad \text{(Init-1)}$$

$$\lceil \mathsf{idle} \rceil \longrightarrow \lceil \mathsf{idle} \vee \mathsf{purge} \rceil \qquad \text{(Seq-1)}$$

$$\lceil \mathsf{purge} \rceil \longrightarrow \lceil \mathsf{purge} \vee \mathsf{ignite} \rceil \qquad \text{(Seq-2)}$$

$$\lceil \mathsf{ignite} \rceil \longrightarrow \lceil \mathsf{ignite} \vee \mathsf{burn} \rceil \qquad \text{(Seq-3)}$$

$$\lceil \mathsf{burn} \rceil \longrightarrow \lceil \mathsf{burn} \vee \mathsf{idle} \rceil \qquad \text{(Seq-4)}$$



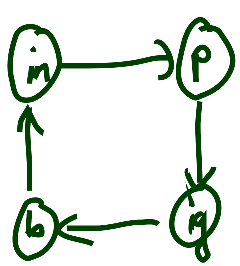
Gas Burner Controller Specification: Timing

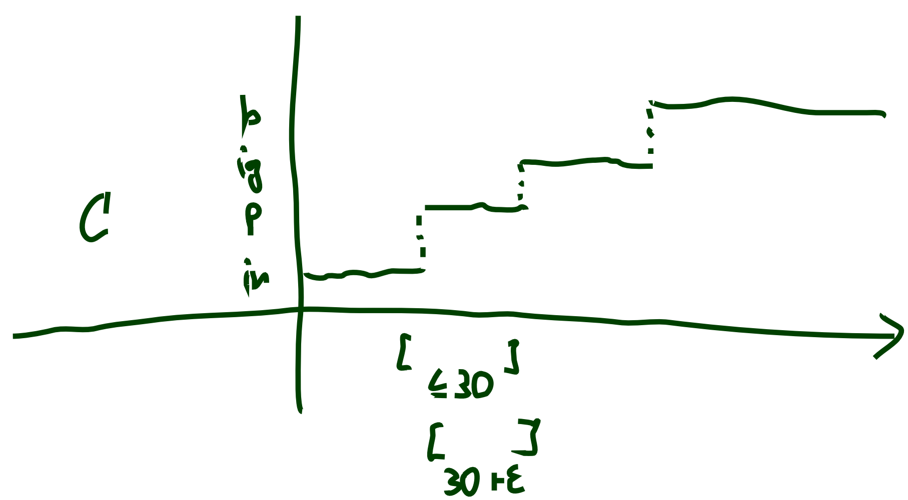
$$\lceil \mathsf{purge} \rceil \xrightarrow{30+\varepsilon} \lceil \neg \mathsf{purge} \rceil \qquad \text{(Prog-1)}$$

$$\lceil \mathsf{ignite} \rceil \xrightarrow{0.5+\varepsilon} \lceil \neg \mathsf{ignite} \rceil \qquad \text{(Prog-2)}$$

$$\lceil \neg \mathsf{purge} \rceil \,;\, \lceil \mathsf{purge} \rceil \xrightarrow{\leq 30} \lceil \mathsf{purge} \rceil \qquad \text{(Stab-2)}$$

$$\lceil \neg \mathsf{ignite} \rceil \,;\, \lceil \mathsf{ignite} \rceil \xrightarrow{\leq 0.5} \lceil \mathsf{ignite} \rceil \qquad \text{(Stab-3)}$$





Gas Burner Controller Specification: Outputs

π φ $\neg\pi$

$$\lceil G \wedge (\mathsf{idle} \vee \mathsf{purge}) \rceil \xrightarrow{\varepsilon} \lceil \neg G \rceil \qquad \text{(Syn-3)}$$

$$\lceil \neg G \wedge (\mathsf{ignite} \vee \mathsf{burn}) \rceil \xrightarrow{\varepsilon} \lceil G \rceil \qquad \text{(Syn-4)}$$

$$\lceil G \rceil \,;\, \lceil \neg G \wedge (\mathsf{idle} \vee \mathsf{purge}) \rceil \longrightarrow \lceil \neg G \rceil \qquad \text{(Stab-6)}$$

$$\lceil \neg G \wedge (\mathsf{idle} \vee \mathsf{purge}) \rceil \longrightarrow_0 \lceil \neg G \rceil \qquad \text{(Stab-6-init)}$$

$$\lceil \neg G \rceil \,;\, \lceil G \wedge (\mathsf{ignite} \vee \mathsf{burn}) \rceil \longrightarrow \lceil G \rceil \qquad \text{(Stab-7)}$$

$\lceil\rceil \vee \lceil \neg G \rceil ; true$

valve closes/opens after ε time units the latest

valve stays closed/open

Gas Burner Controller Specification: Inputs

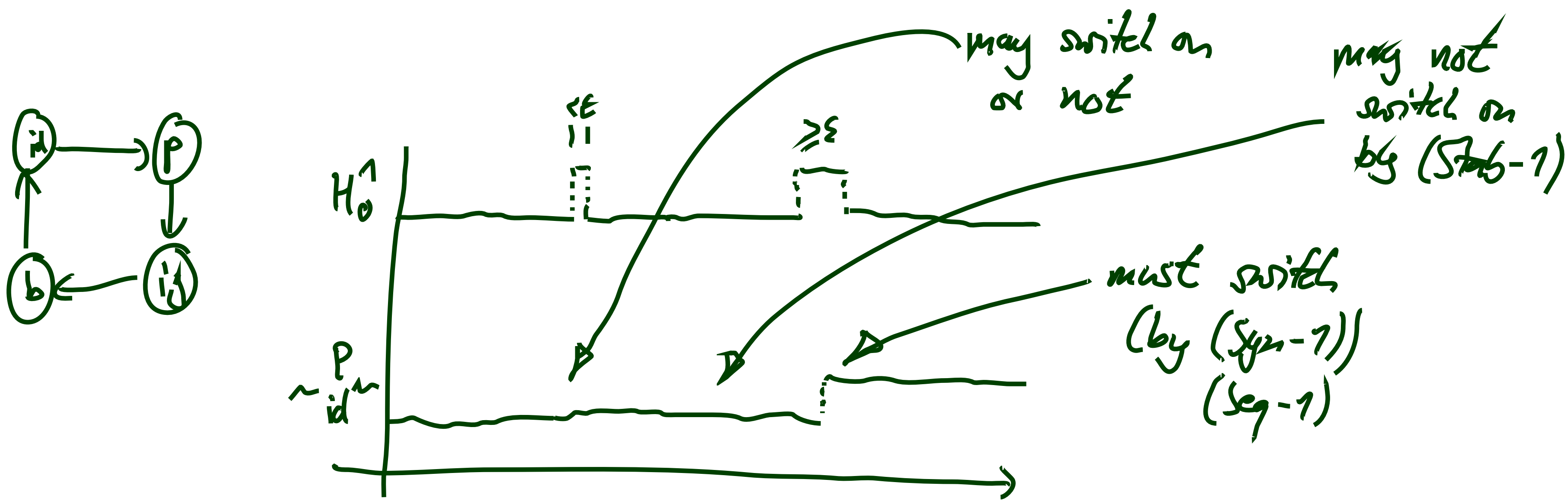
$$\lceil \mathsf{idle} \wedge H \rceil \xrightarrow{\varepsilon} \lceil \neg \mathsf{idle} \rceil \qquad \text{(Syn-1)}$$

$$\lceil \mathsf{burn} \wedge (\neg H \vee \neg F) \rceil \xrightarrow{\varepsilon} \lceil \neg \mathsf{burn} \rceil \qquad \text{(Syn-2)}$$

$$\lceil \neg \mathsf{idle} \rceil \, ; \lceil \mathsf{idle} \wedge \neg H \rceil \longrightarrow \lceil \mathsf{idle} \rceil \qquad \text{(Stab-1)}$$

$$\lceil \mathsf{idle} \wedge \neg H \rceil \longrightarrow_0 \lceil \mathsf{idle} \rceil \qquad \text{(Stab-1-init)}$$

$$\lceil \neg \mathsf{burn} \rceil \, ; \lceil \mathsf{burn} \wedge H \wedge F \rceil \longrightarrow \lceil \mathsf{burn} \rceil \qquad \text{(Stab-4)}$$



Gas Burner Controller Specification: Assumptions

$$\lceil\rceil \vee \lceil \neg H \rceil \, ; \, true \qquad \text{(Init-2)}$$

$$\lceil\rceil \vee \lceil \neg F \rceil \, ; \, true \qquad \text{(Init-3)}$$

$$\lceil\rceil \vee \lceil \neg G \rceil \, ; \, true \qquad \text{(Init-4)}$$

$$\lceil F \rceil \, ; \, \lceil \neg F \wedge \neg \mathsf{ignite} \rceil \longrightarrow \lceil \neg F \rceil \qquad \text{(Stab-5)}$$

$$\lceil \neg F \wedge \neg \mathsf{ignite} \rceil \longrightarrow_0 \lceil \neg F \rceil \qquad \text{(Stab-5-init)}$$

no spontaneous flames

References

[Olderog and Dierks, 2008] Olderog, E.-R. and Dierks, H. (2008). *Real-Time Systems - Formal Specification and Automatic Verification*. Cambridge University Press.