Program Semantics in Model-Based WCET Analysis

--- State of the Art Perspective ---

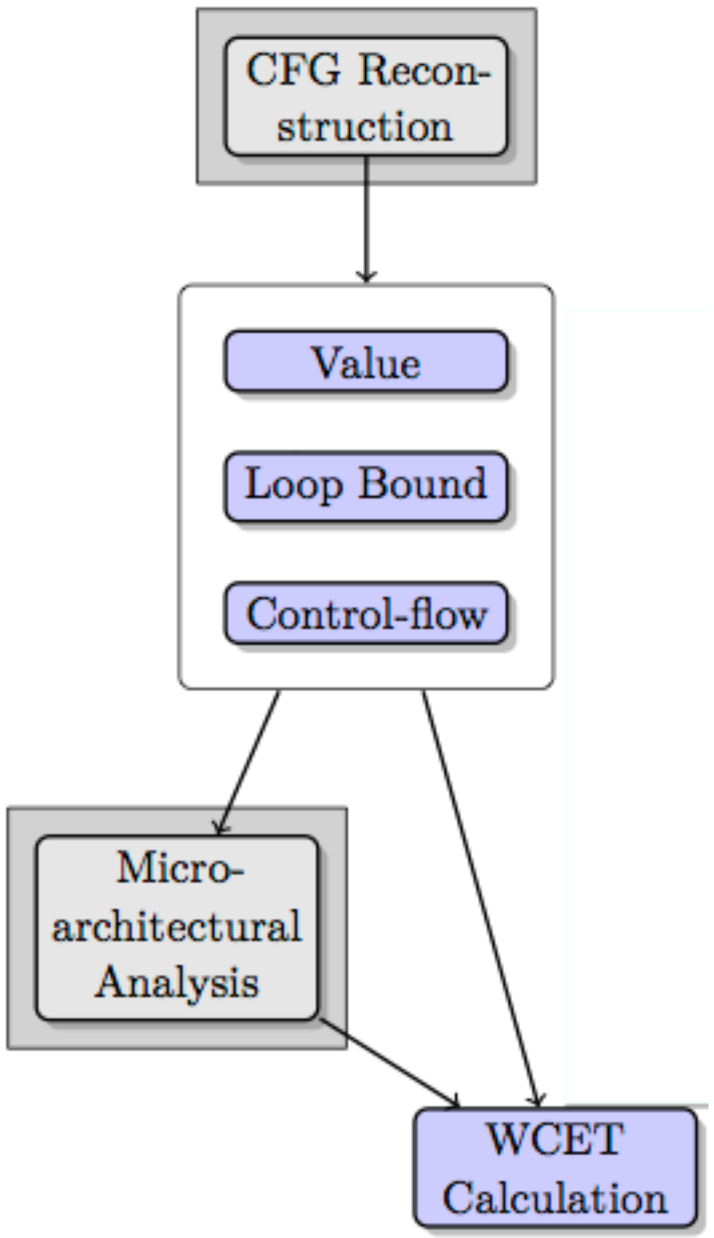
Mihail Asavoae   Claire Maiza   Pascal Raymond

W-SEPT

WCET: **SE**mantics, **P**recision and **T**raceability

WCET 2013 - PARIS

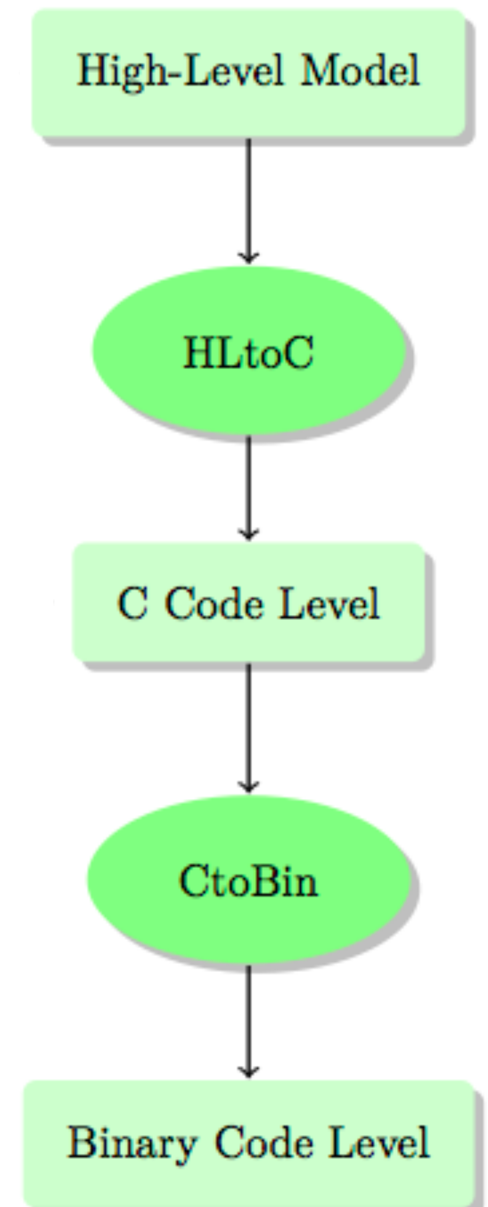
Key Point! - Advocate for WCET analysis based on the program semantics



(1) Model-Based Design

+

(2) WCET Analysis



- Synchronous Programming

"This spirit is in full accordance with modern thinking about system design, sometimes called ***model-based design***, which advocates the use of high-level models with ***"ideal", implementation-independent semantics***, and ***the separation of concerns between design and implementation***."

"Synchronous Programming" - P. Caspi et al.
in Handbook of Real-Time and Embedded Systems - 2008

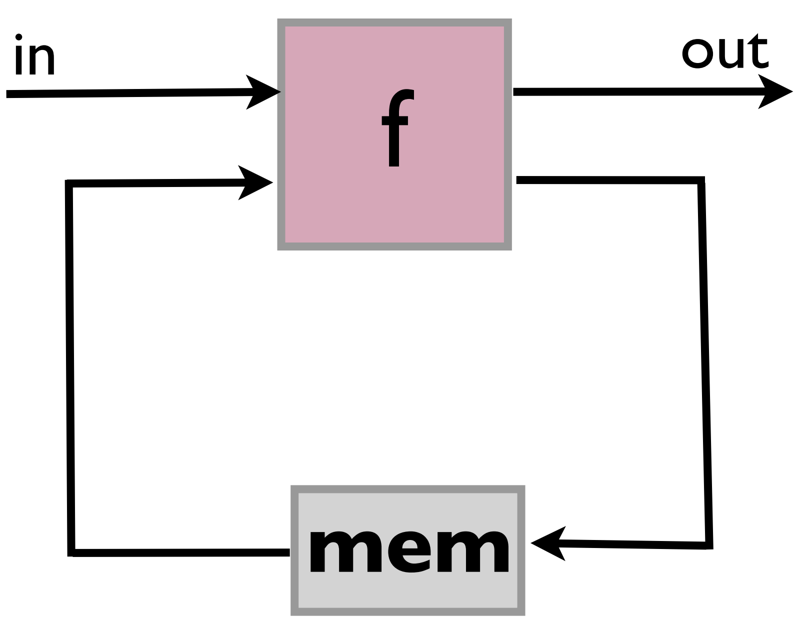
- Generic programming paradigm - **designed to generate code**

  - deterministic concurrent programming style
  - in **data-flow** (e.g. Lustre), **control-flow** (e.g. Esterel) flavors, or **both** (e.g. Scade 6 - industrial version of Lustre + finite state machines)

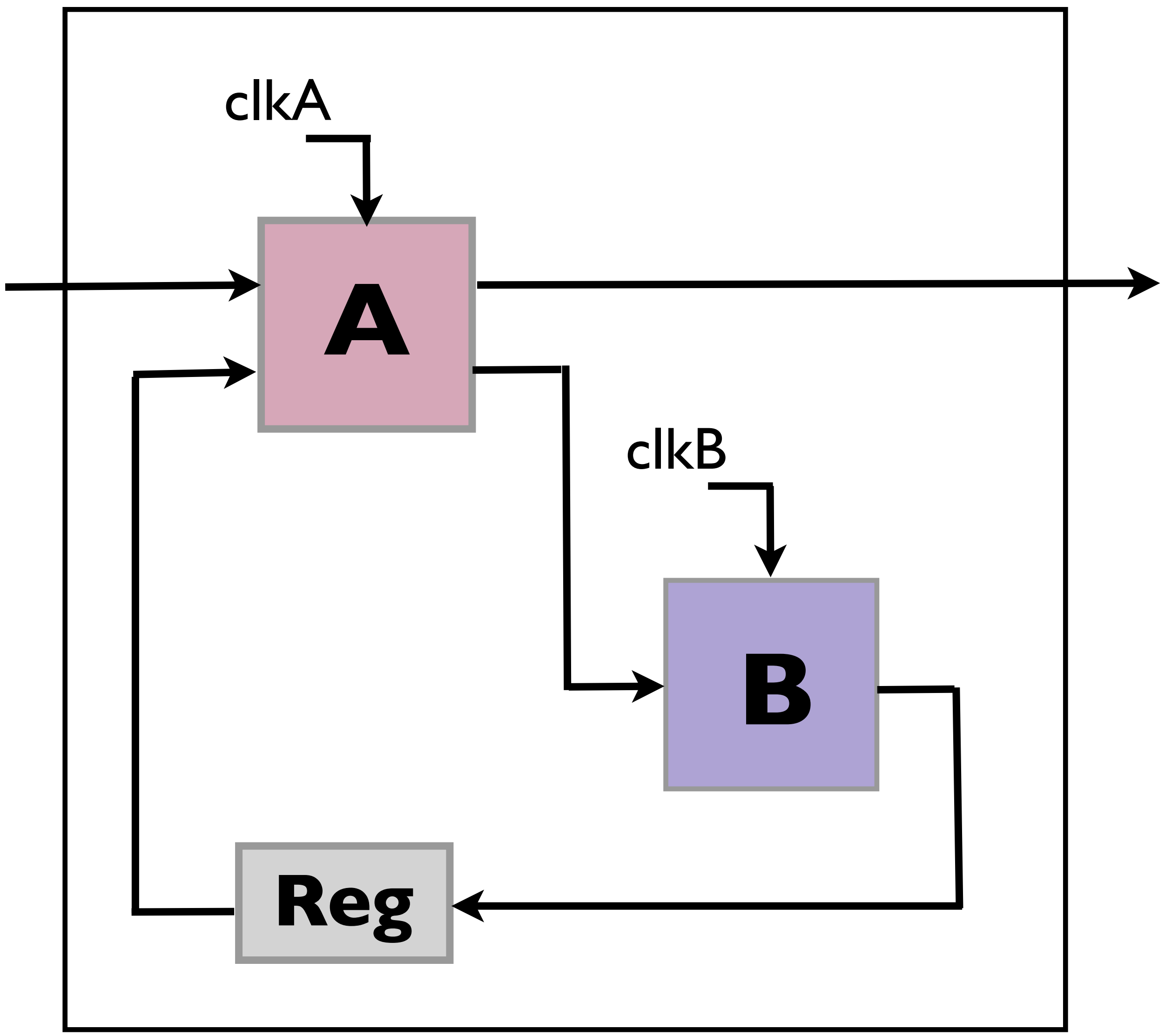
- **Lustre**

  - ***semantics*** - functions over flows
  - ***style*** - network of operating nodes (similar to sequential circuits or block diagrams)

  - Scade - in critical systems: transportation, energy



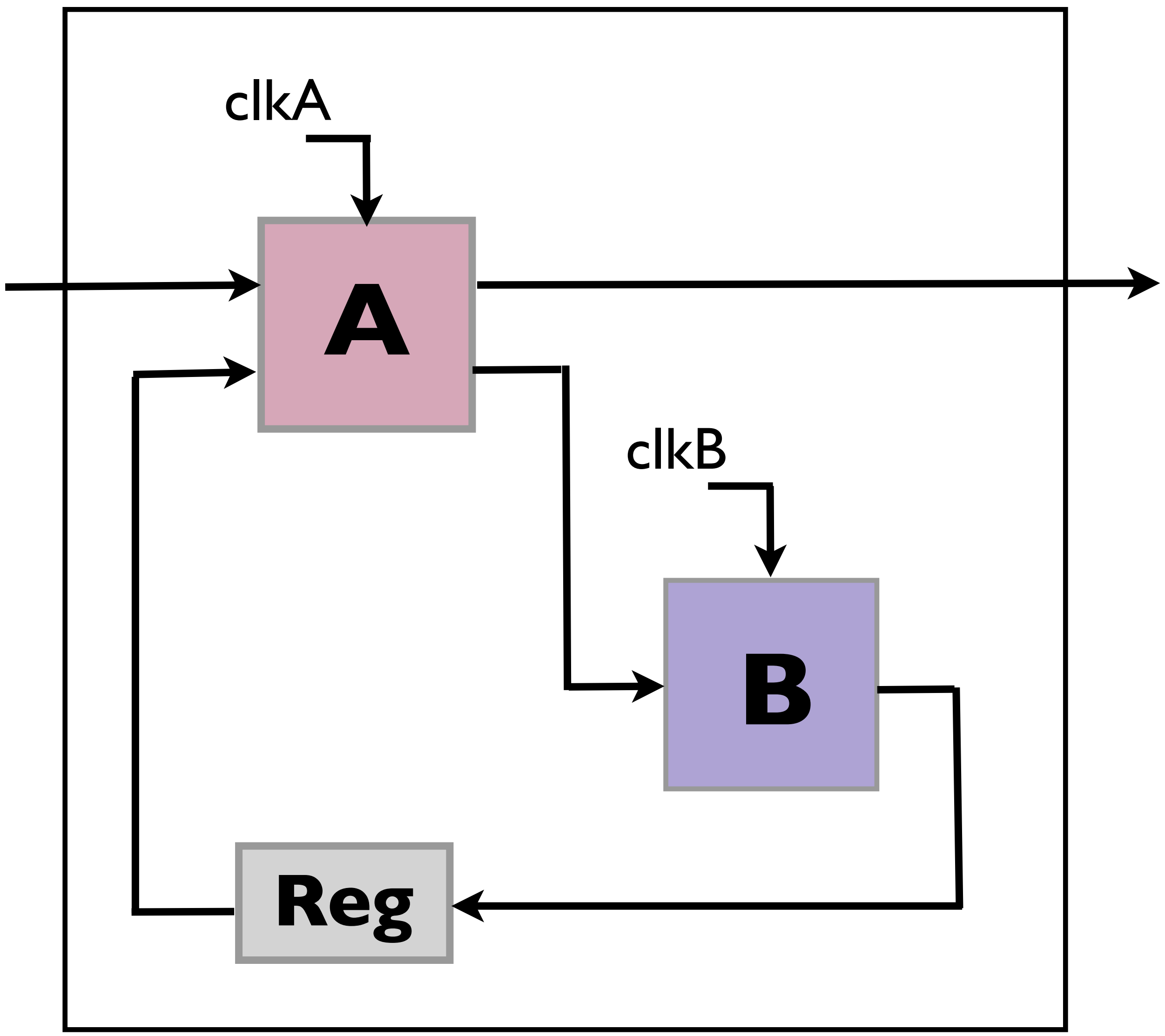
Design



Elements

- overall → subsystems (**A**, **B**)
- subsystem → inter-dependent data flows
- dependencies → direct and logical (delays, i.e. **Reg**)
- clocks (*clkA* and *clkB*)

Design



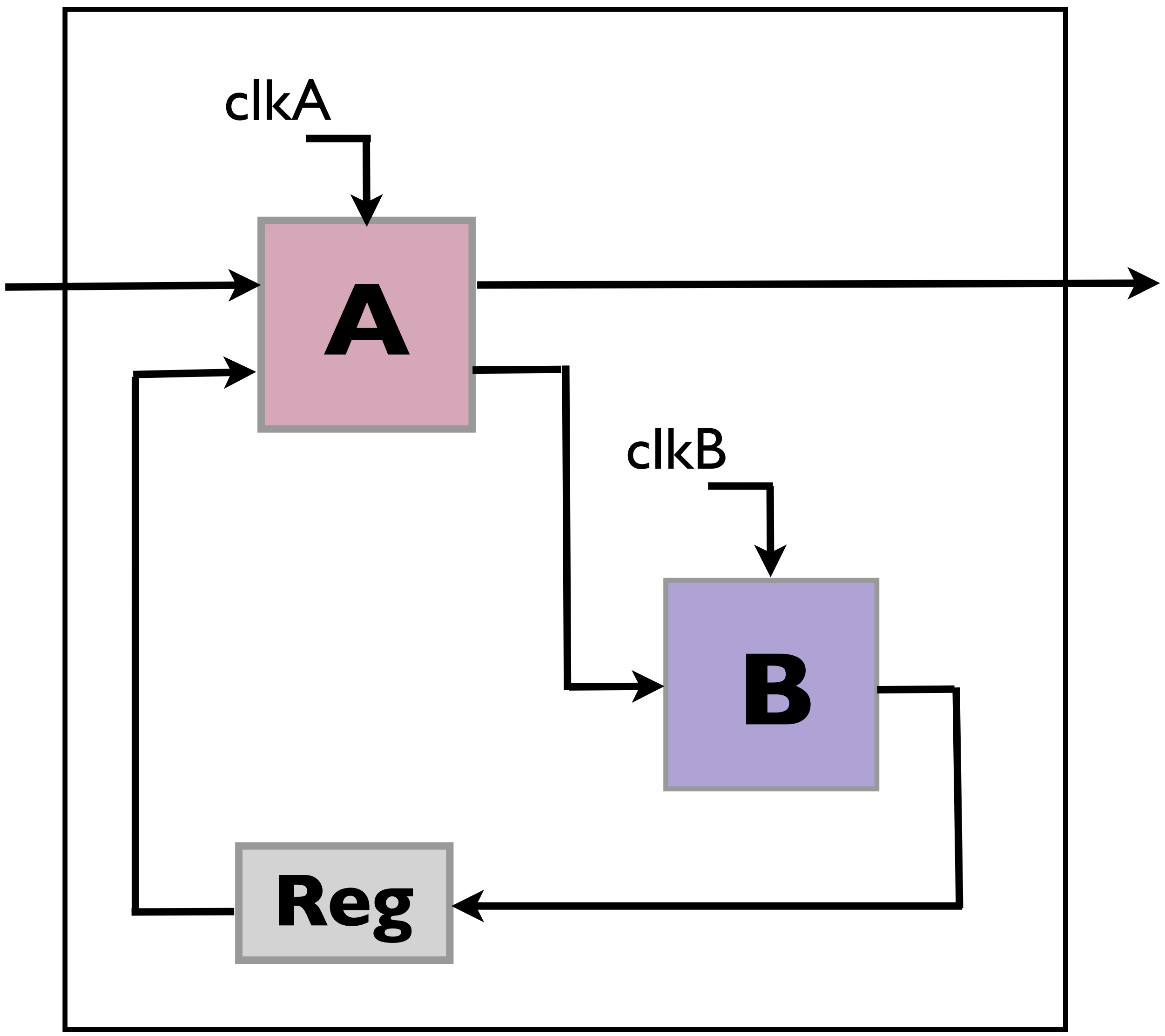
Elements

→ *"... the separation of concerns between design and implementation."*

Design:
- instantaneous execution

Implementation:
- bounded execution (with a ***guaranteed upper bound on time***)

Design



C Code

- scheduling → **A** before **B**

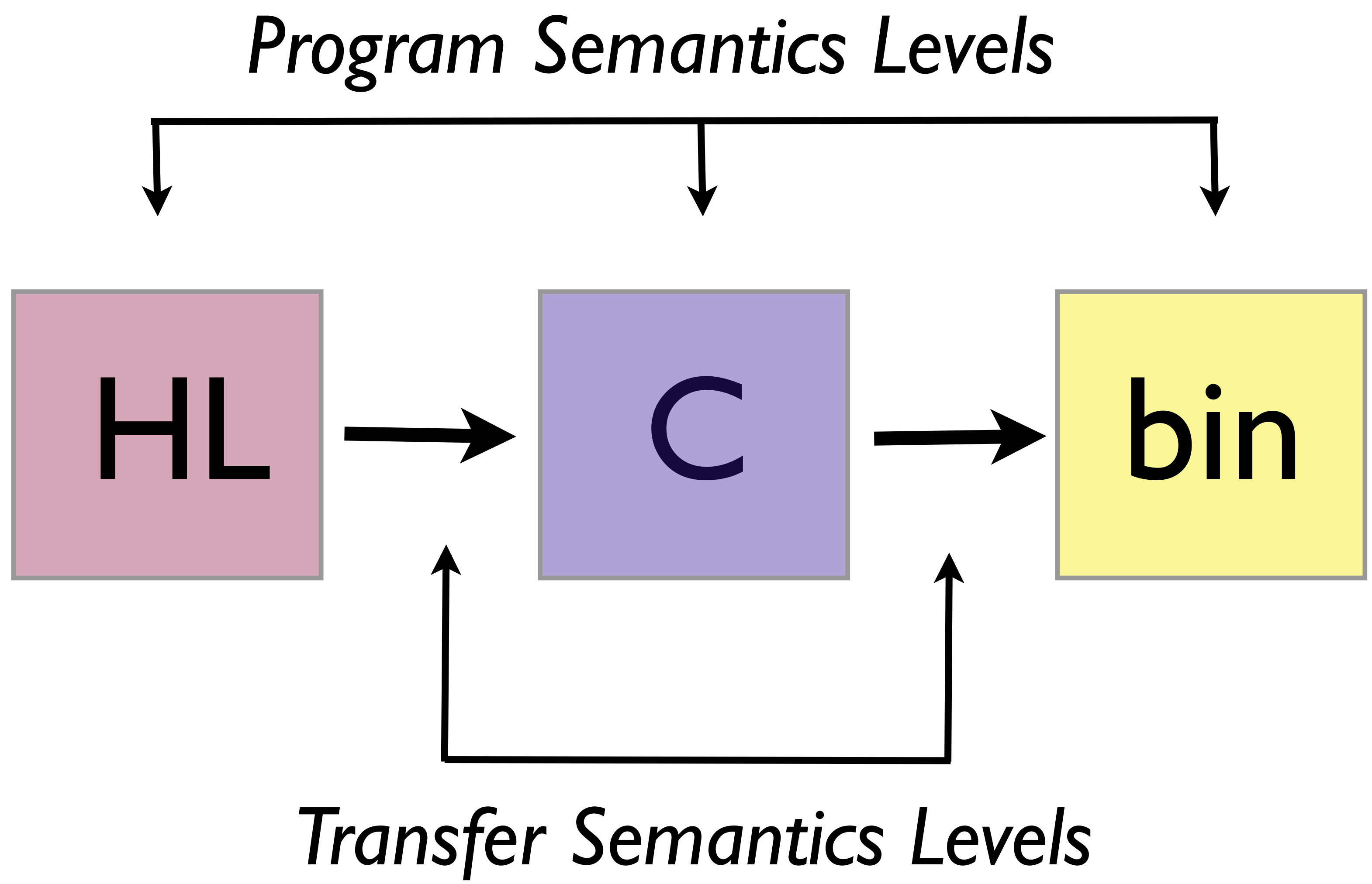
- code →

```
global Reg;

if clkA then A_Step()
if clkB then {
  B_Step();
  Reg = B.out;
}
```

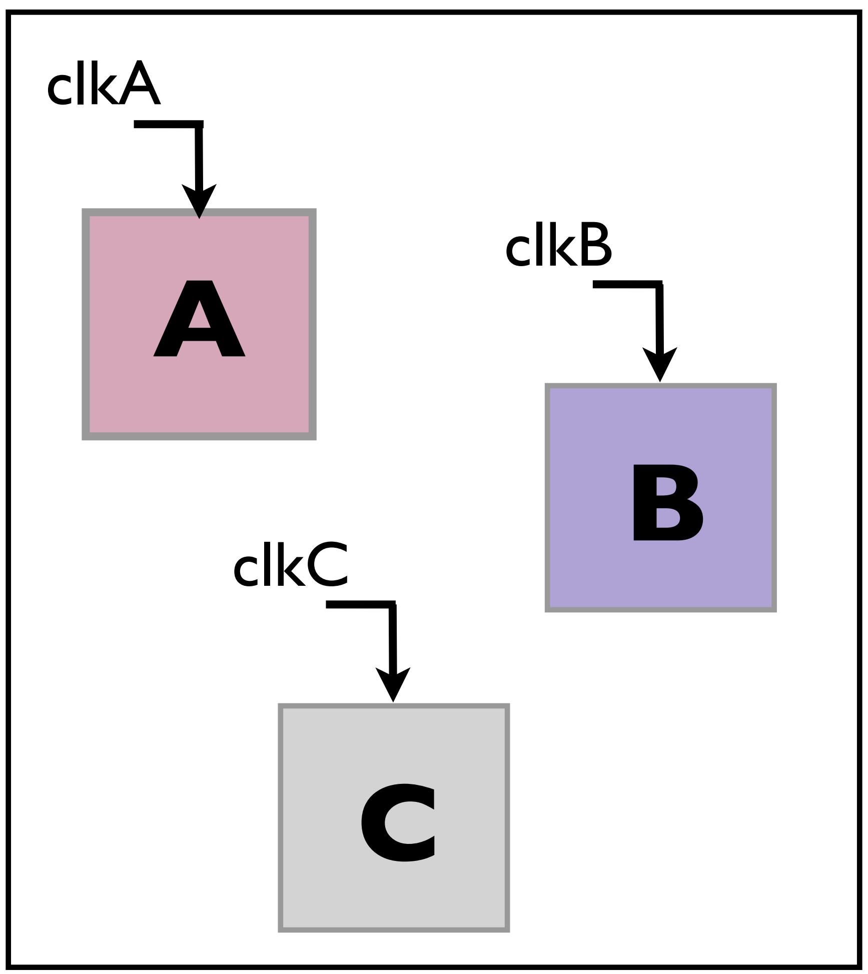
- Deterministic, traceable C code (heap-free, no recursion, bounded loops)

- Typical Synchronous Program + Compilation

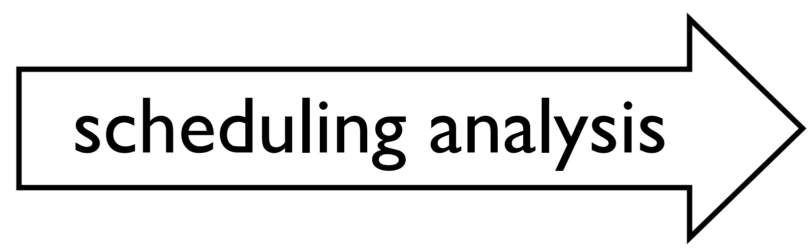


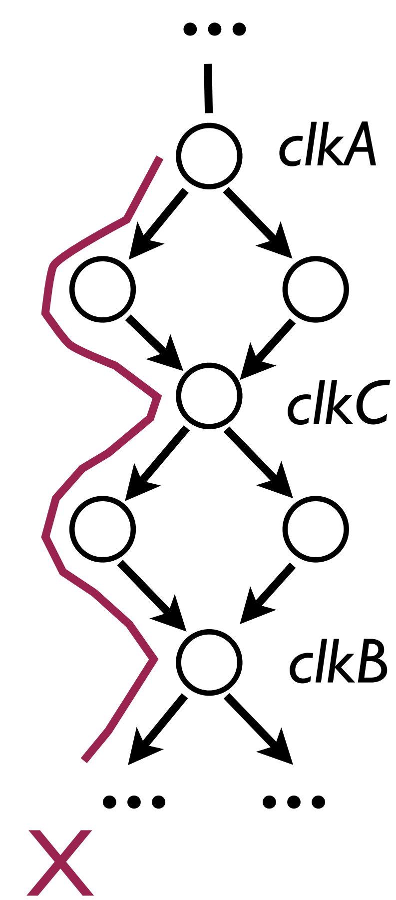
- From the WCET analysis perspective : both global and local interest

- Generated code from model-based designs:
  - → subsystems
    - → inter-dependent
      - → **clocks (Lustre)**

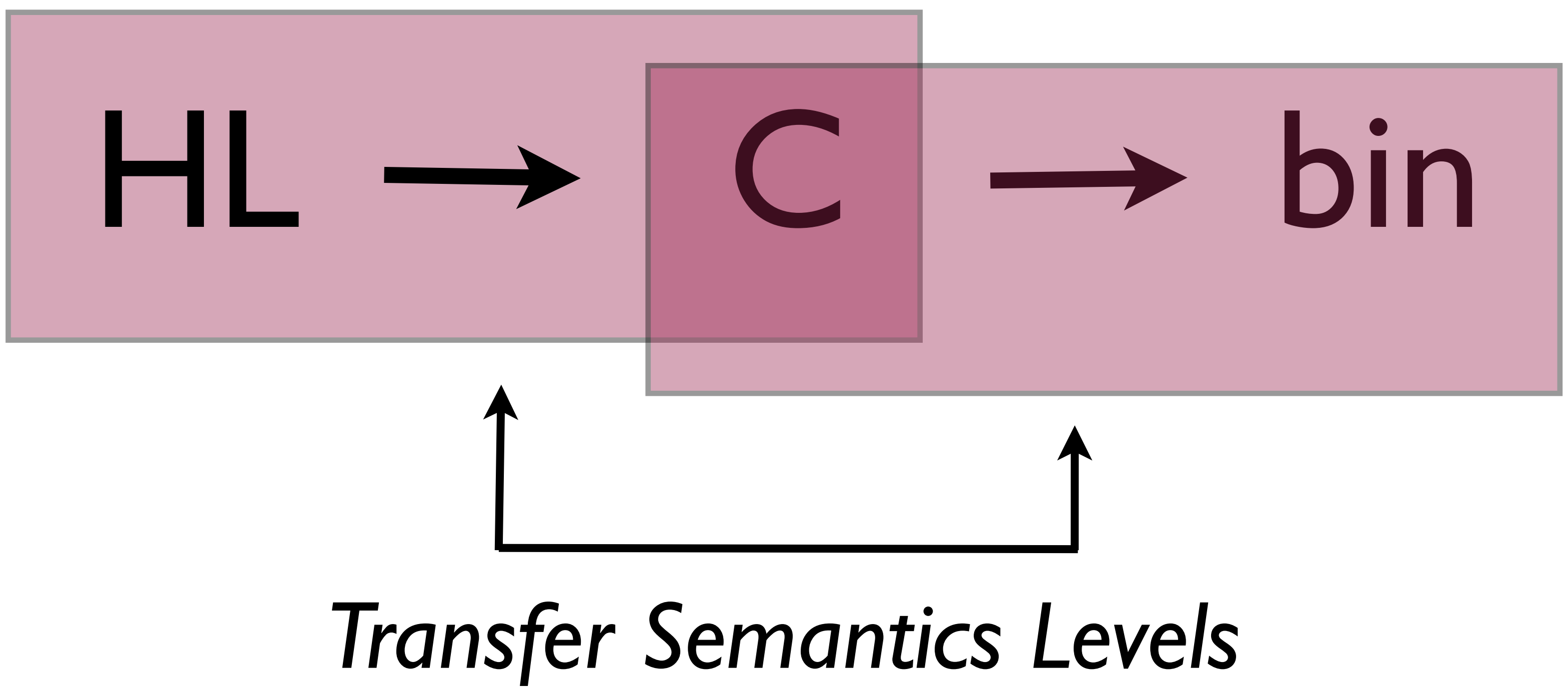


- assume a relation on the nodes: at most one of **A**, **B**, **C** is not executed





- Two levels of interest:



- **HL to C** : it features a higher degree of control over the model semantics transformation

- **C to bin** : compilation with optimizations

Traceability

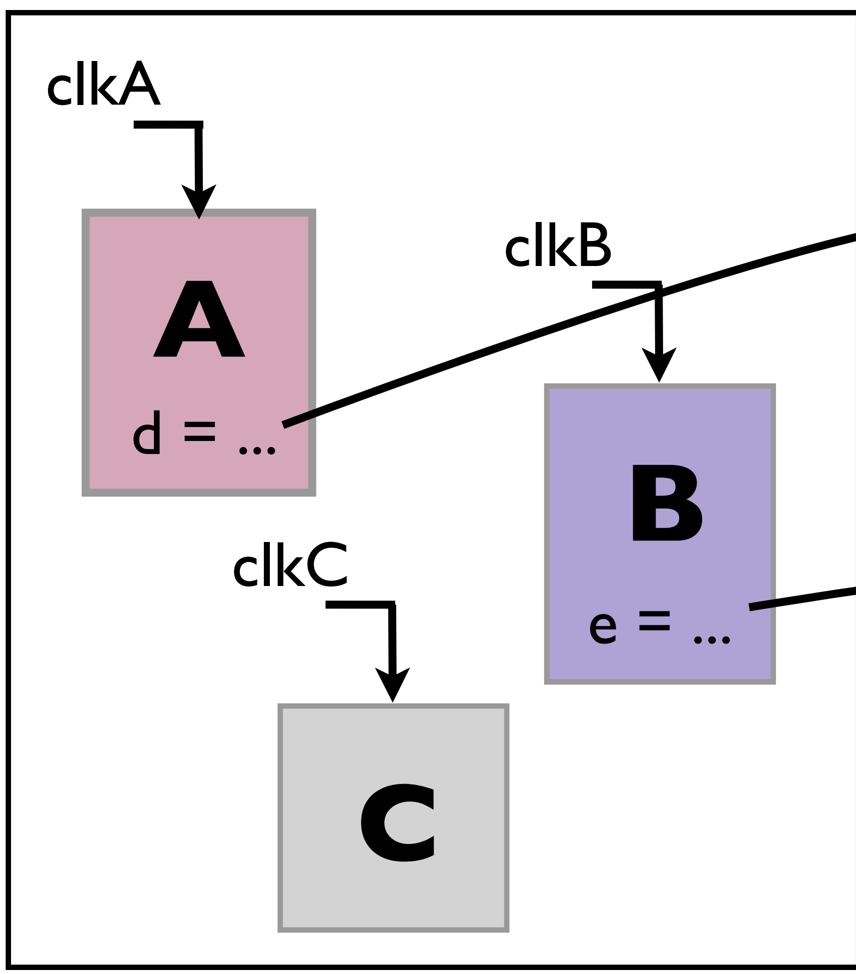
- identify relations between different representations

HL → C

.C FILE

if *clkA* then **A**_Step()

if *clkB* then **B**_Step()



A_Step() {
d = ...
}

B_step() {
e = ...
}

- transfer of semantics through optimizations

C → bin

if *clkA* then **A**_Step()
if *clkB* then **B**_Step()

```
A_Step() {
    d = ...
}
B_step() {
    e = ...
}
```

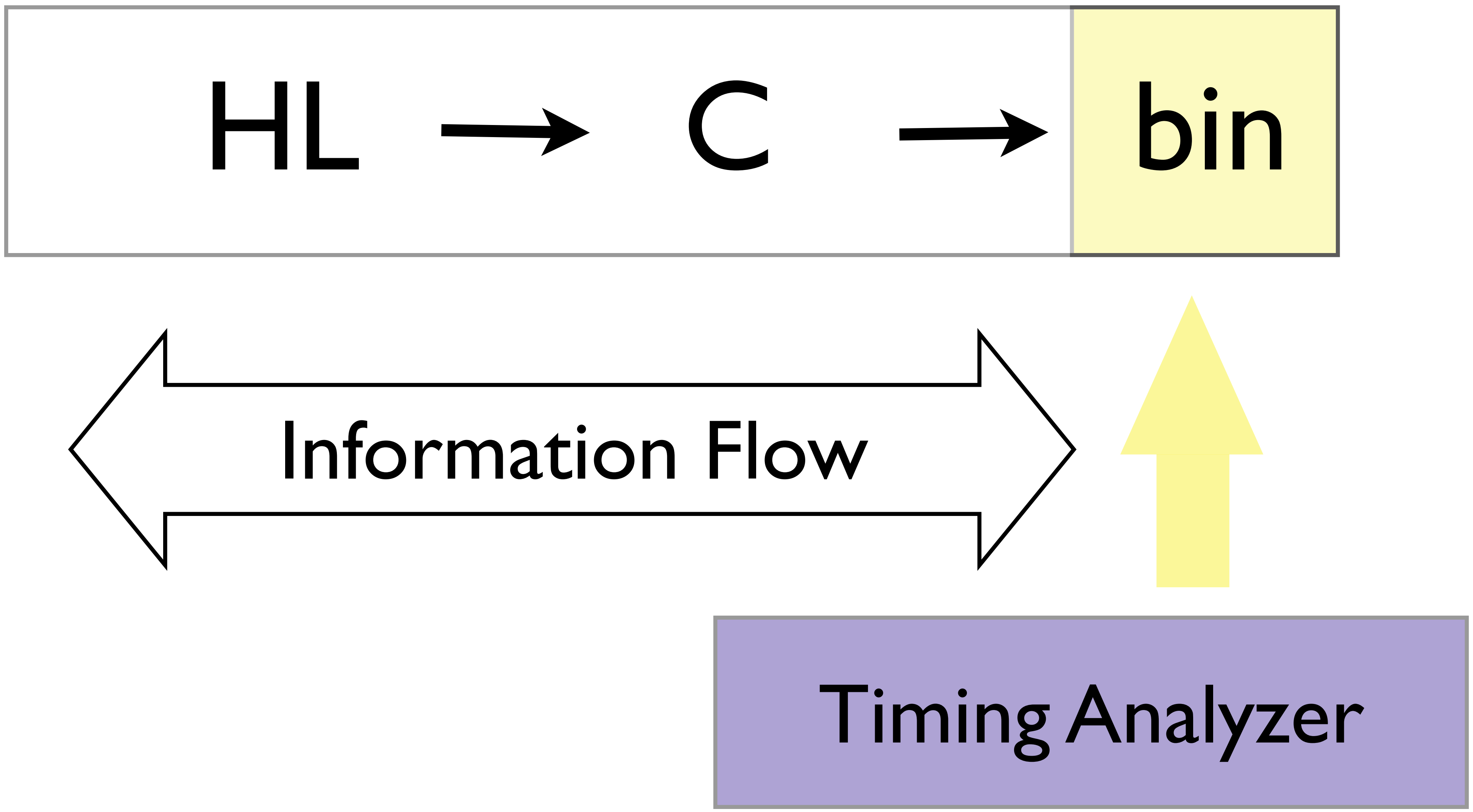
- **obfuscation of the control structure**

- no clear solution, in general optimizations are handled in a case by case manner

  - specialized annotation language

  - transfer functions

- Integration methods



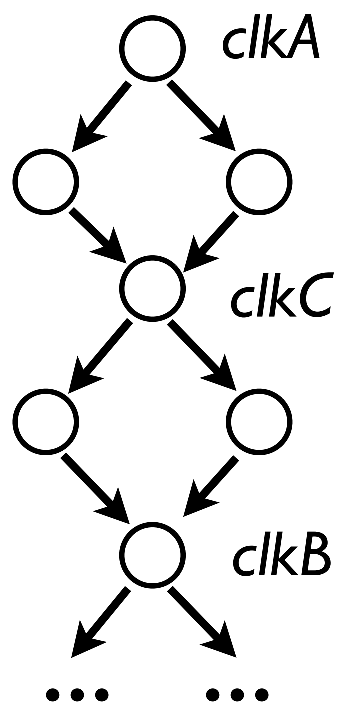
- seamless integration vs modified MBD infrastructure

Path Analysis

- the path analysis determines the worst-case execution time based on previously computed (or annotated) artifacts

    - represented outside MBD

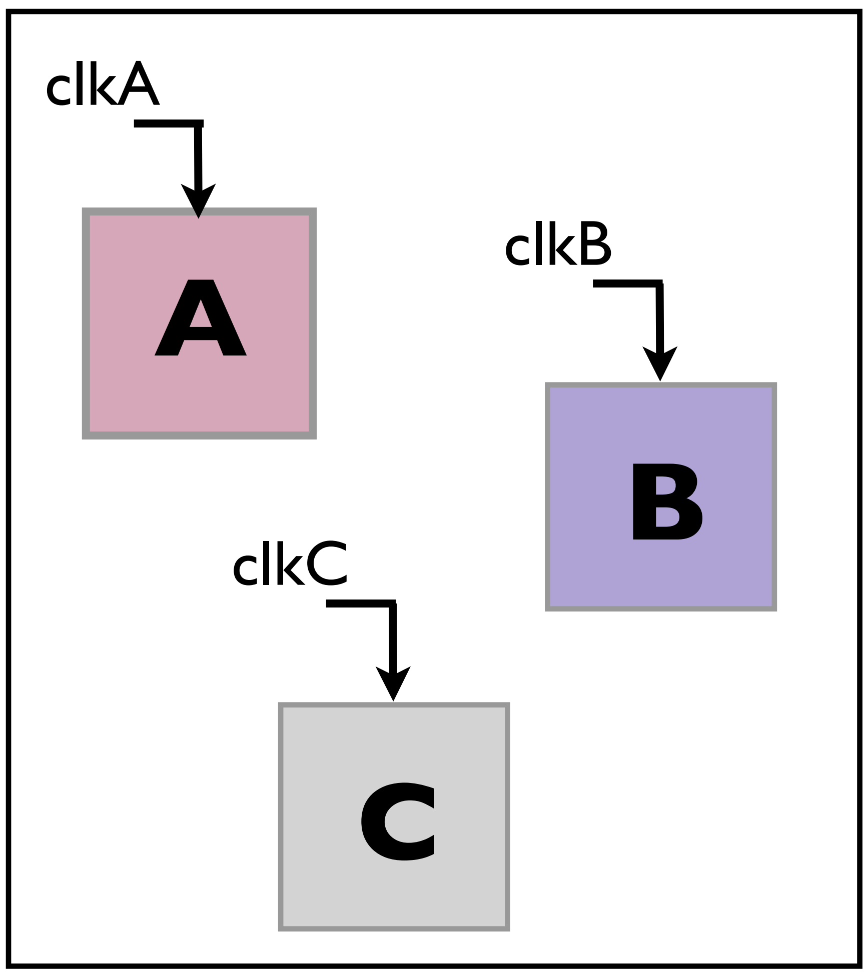
- **IPET** - alternatives: on the program syntax, on model checking



Is IPET the best approach to represent the semantics?

- separation of concerns at the level of WCET analysis workflow

- IPET - path analysis through ILP constraint solving
  - at most one of **A**, **B**, **C** is not executed

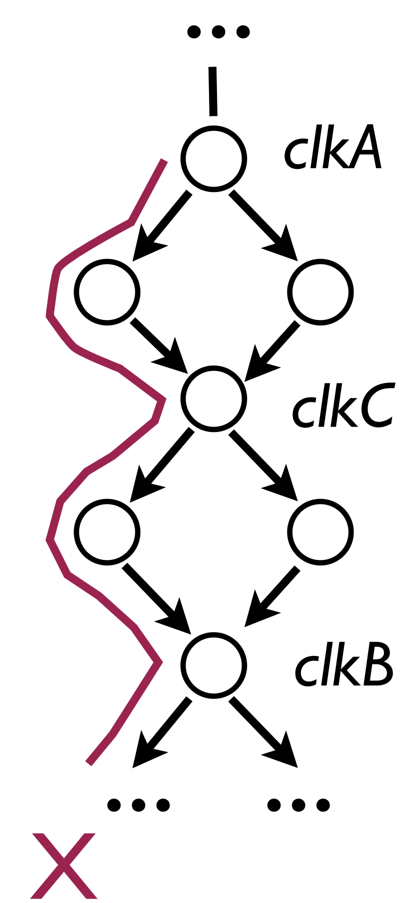


Solve:

$X_A + X_B + X_C \leq 2$

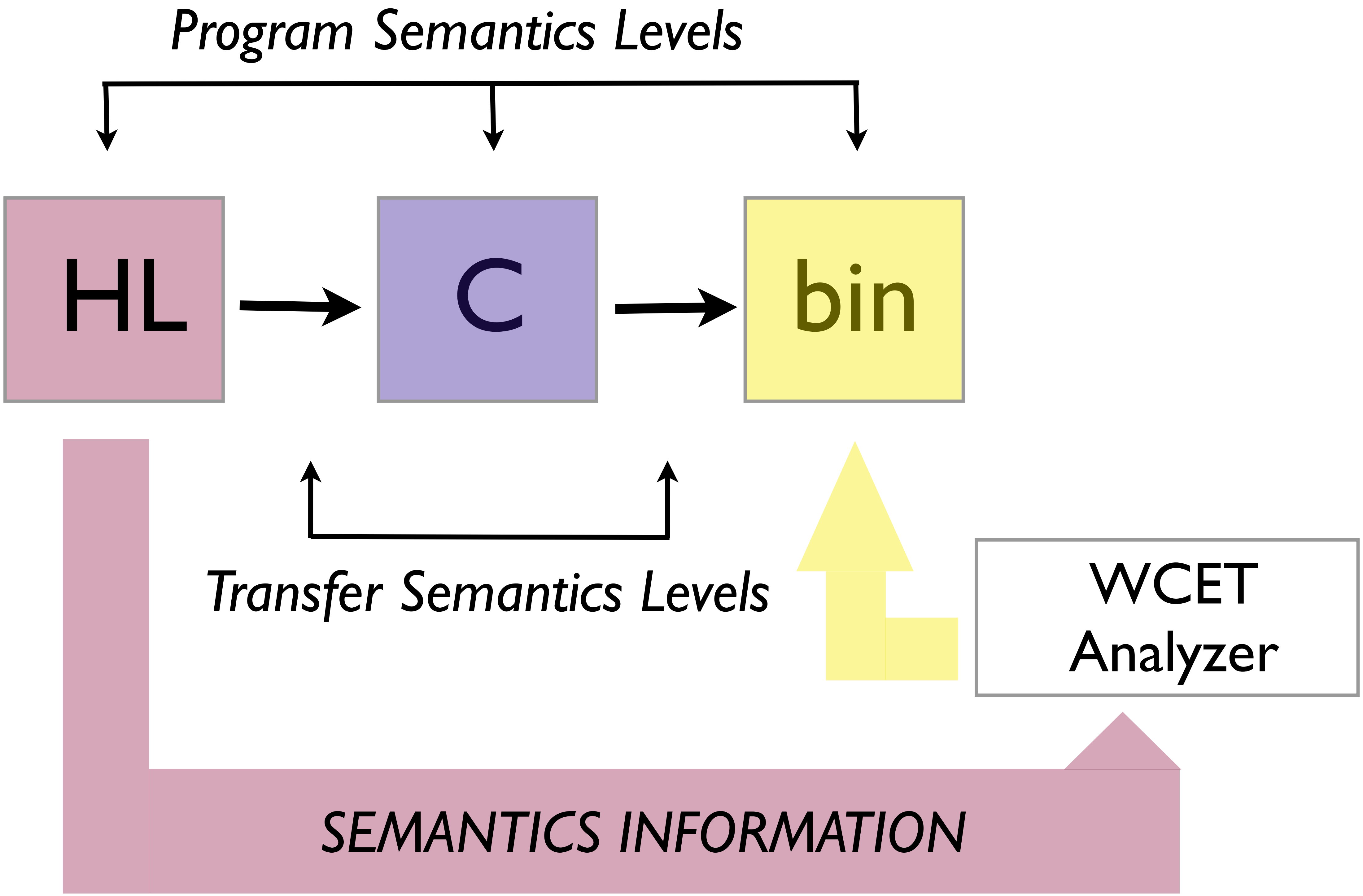
and

ILP representation of the control structure



- IPET enhancements are possible

Manifesto



Program Semantics Levels
HL
C
bin
Transfer Semantics Levels
WCET
Analyzer
SEMANTICS INFORMATION

Conclusions

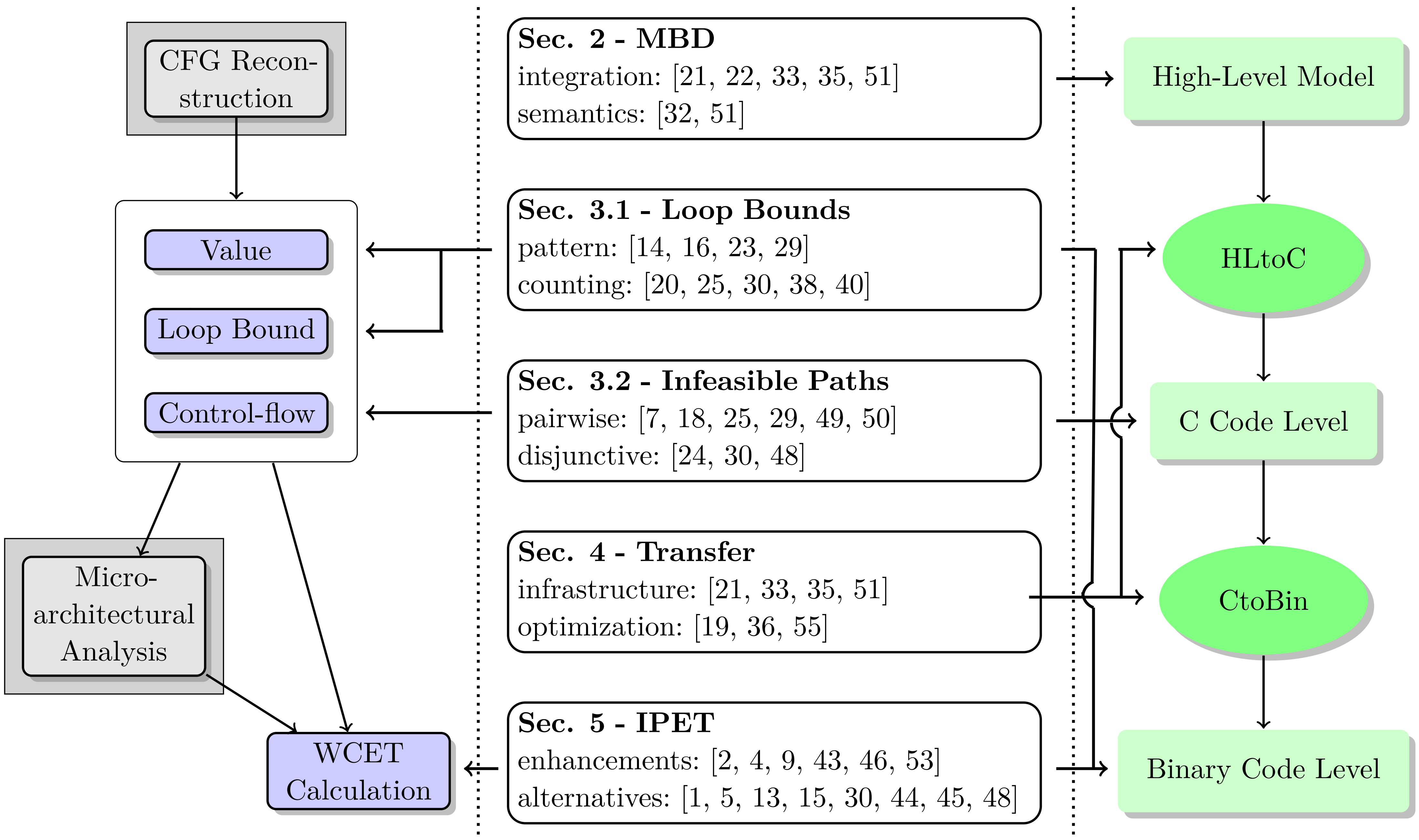
- the **paper** presents a state-of-the-art overview of approaches for both local and global w.r.t. MBD workflow, from the WCET analysis perspective

  - particular MBD: synchronous programs as models and C code generation capabilities

- the **presentation** (complements the paper)

  - follows several aspects of the synchronous programming

  - advocates for a separation of concerns at the level of WCET analysis


CFG Recon-struction
Value
Loop Bound
Control-flow
Micro-architectural Analysis
WCET Calculation
Sec. 2 - MBD
integration: [21, 22, 33, 35, 51]
semantics: [32, 51]
Sec. 3.1 - Loop Bounds
pattern: [14, 16, 23, 29]
counting: [20, 25, 30, 38, 40]
Sec. 3.2 - Infeasible Paths
pairwise: [7, 18, 25, 29, 49, 50]
disjunctive: [24, 30, 48]
Sec. 4 - Transfer
infrastructure: [21, 33, 35, 51]
optimization: [19, 36, 55]
Sec. 5 - IPET
enhancements: [2, 4, 9, 43, 46, 53]
alternatives: [1, 5, 13, 15, 30, 44, 45, 48]
High-Level Model
HLtoC
C Code Level
CtoBin
Binary Code Level