Model Level Timing Analysis for IEC 61499 and 4DIAC

Luka Lednicki, Jan Carlson
Mälardalen Real-time Research Centre
Mälardalen University, Västerås, Sweden

MÄLARDALEN UNIVERSITY
SWEDEN

ABB

Outline

- Motivation
- Background
- WCET Analysis
  - Overview
  - WCET data
  - Analysis algorithms
- Implementation
- Future Work & Conclusion

What are the problems we are trying to solve?

Motivation

The Problem

- Traditionally, timing analysis is performed late in the development
  - Analysis is applied only to deployed systems
  - Late detection of problems – increased development time and cost
- Many timing analysis methods are inefficient
  - Performing analysis on complex systems takes a long time

Our Goal

- Early analysis
  - Apply analysis before the systems are deployed or fully implemented
  - Detect potential problems early in the development process
- Analyze often
  - Efficient analysis technique
  - Use analysis during whole development, not as a separate stage
  - If possible, with each change to the system model

What is WCET?

Background

Worst-Case Execution Time

- Maximum amount of time that will be spent to finish execution of a functionality
  - Any error/approximation must be on the safe side
- Time is a resource
  - We are interested in resource usage, not elapsed time
  - Interruptions are irrelevant
- Used for various timing analysis
  - Schedulability, response time, processor utilization

How do we address the problems?

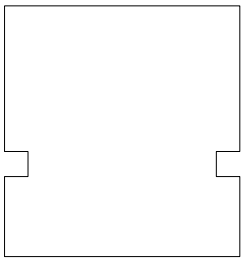
WCET Analysis

Overview

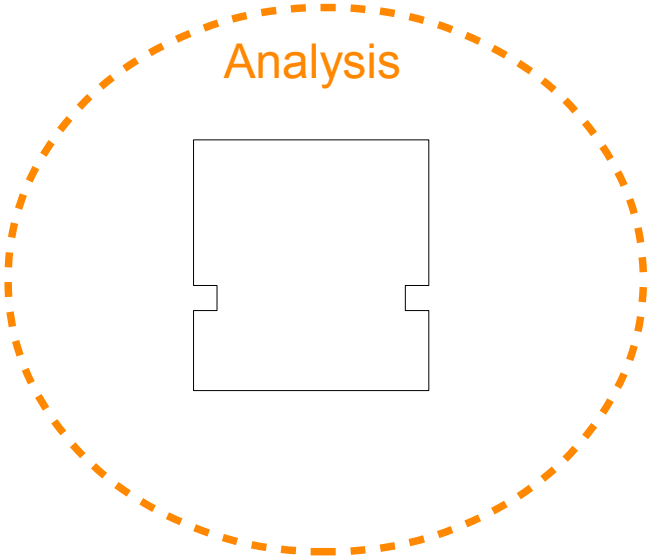
- We perform analysis on system models
  - Abstract view of a system – more efficient analysis technique
  - Models can be available early in system development
  - Unimplemented parts can be replaced by dummies / estimations
  - IEC 61499 systems are built using models – models are available and up-to-date
- Analysis is performed in a compositional manner
  - Each FB is analyzed only once
  - Analysis results (WCET data for FBs) are stored with FBs
  - WCET data for composites/applications is derived by composing existing WCET data for enclosed FBs

Overview

1. Take a FB



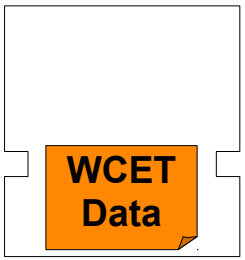
Overview



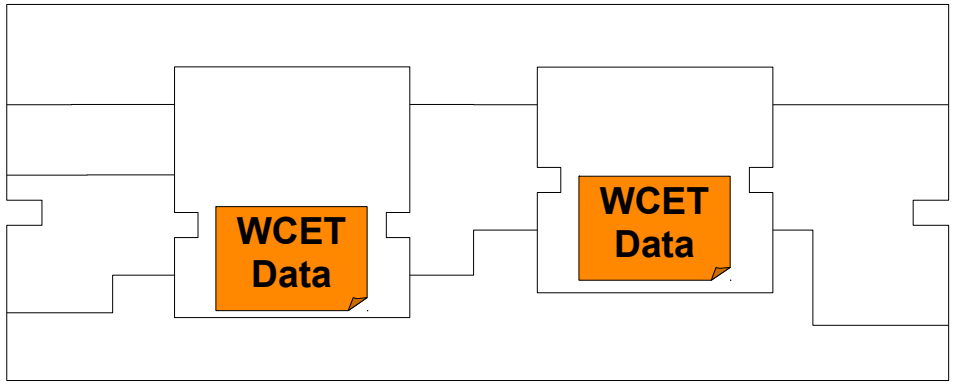
1. Take a FB
2. Perform analysis

Overview

1. Take a FB
2. Perform analysis
3. Store results with the FB

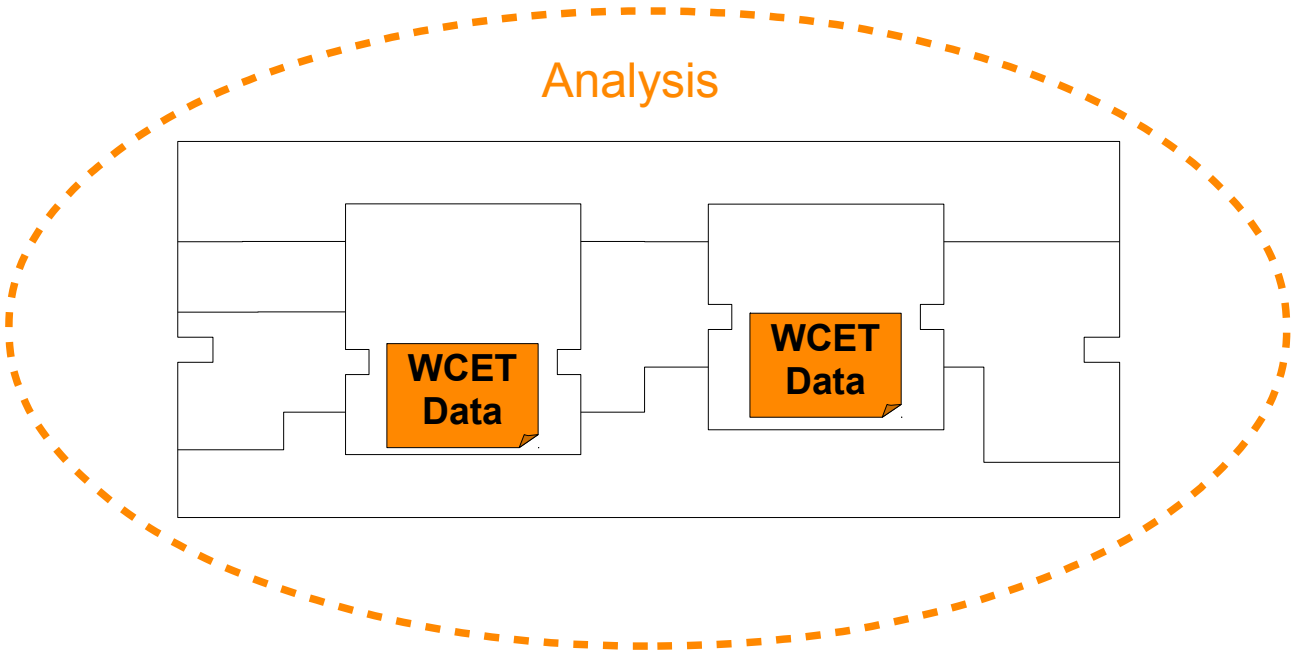


Overview



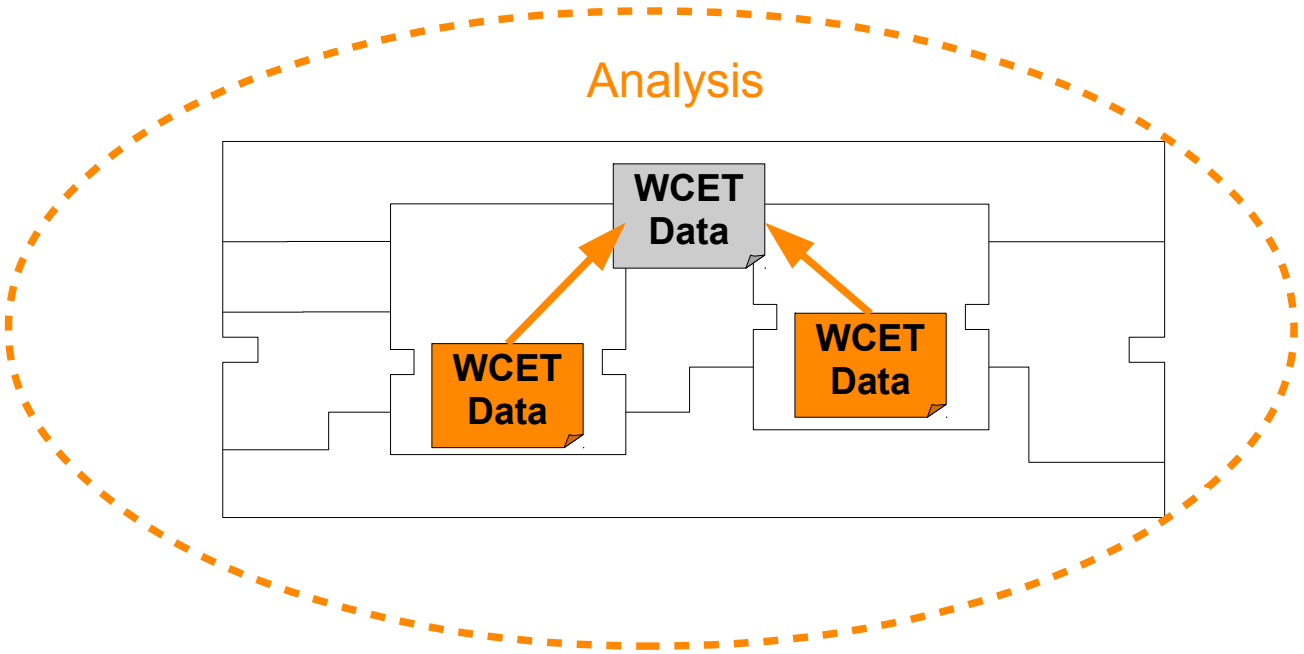
1. Take a FB
2. Perform analysis
3. Store results with the FB
4. Use the FB in a composite/application

Overview



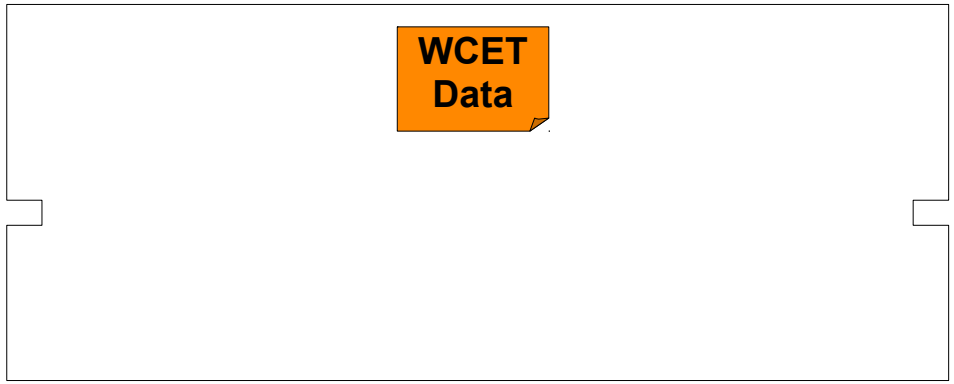
1. Take a FB
2. Perform analysis
3. Store results with the FB
4. Use the FB in a composite/application
5. Perform analysis of the composite/application

Overview



1. Take a FB
2. Perform analysis
3. Store results with the FB
4. Use the FB in a composite/application
5. Perform analysis of the composite/application
6. Combine already existing WCET data to get results

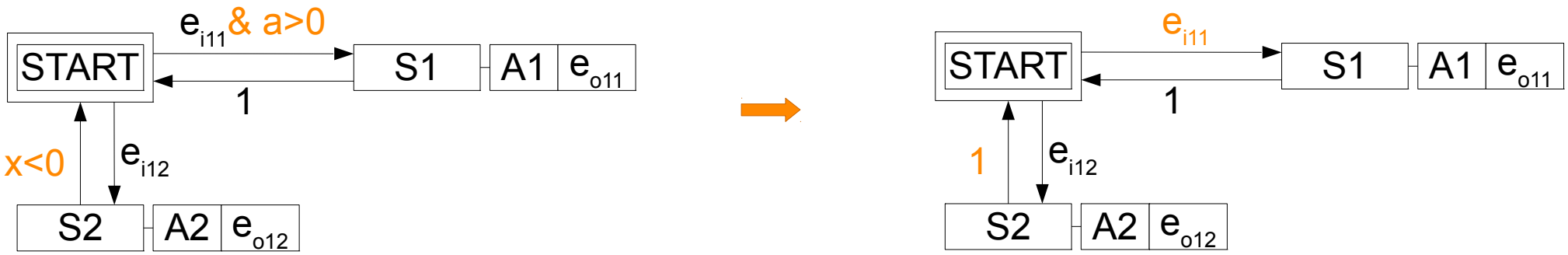
Overview



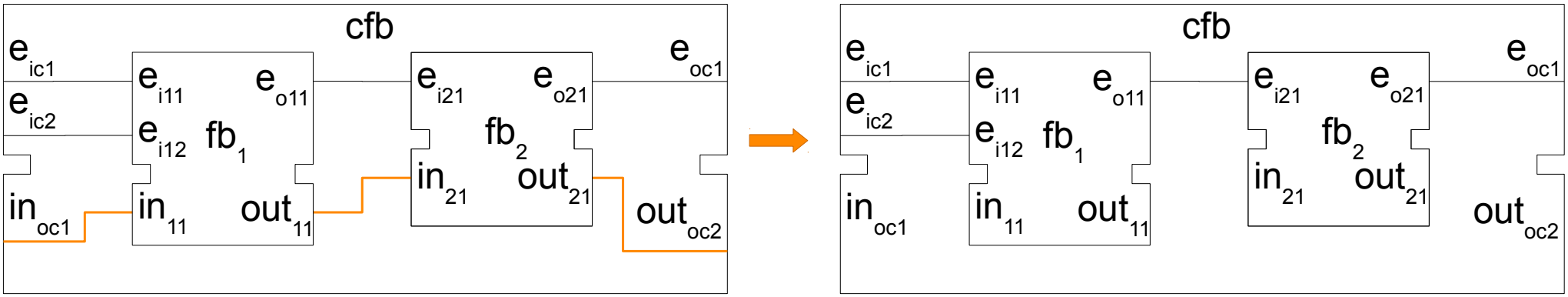
1. Take a FB
2. Perform analysis
3. Store results with the FB
4. Use the FB in a composite/application
5. Perform analysis of the composite/application
6. Combine already existing WCET data to get results
7. Store results with the composite/application

Data Independent Analysis

- Disregarding data conditions from ECC transition guards



- Disregarding data connections



Limitations

- No analysis of SIFBs
  - Their implementation is not defined by a model
  - WCET data must be defined by hand
- Event loops are not supported
  - What is the WCET or period of execution of something that never stops?

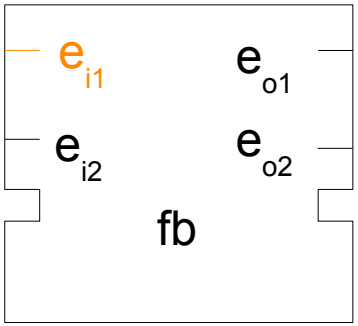
WCET Data

WCET Data

- Algorithms – only a WCET value
  - We assume it has already been acquired
  - Code analysis, measurements...
- Function blocks – context independent data
  - Analyze each FB only once and reuse results
  - Besides the WCET value, we need to know what effect will the execution have on the rest of the system

FB WCET Data

- Data for each input and internal trigger
- WCET value
- Information about generated outputs
- Multiple execution paths – multiple entries



Input event or period | WCET value | Generated outputs

Information for input events

<{ <e_{i1}, { < 10, {e_{o1}=1}>,
< 5, {e_{o1}=2, e_{o2}=1}> }>,

Multiple Execution paths

<e_{i2}, WCET info> },

Information for periodical execution

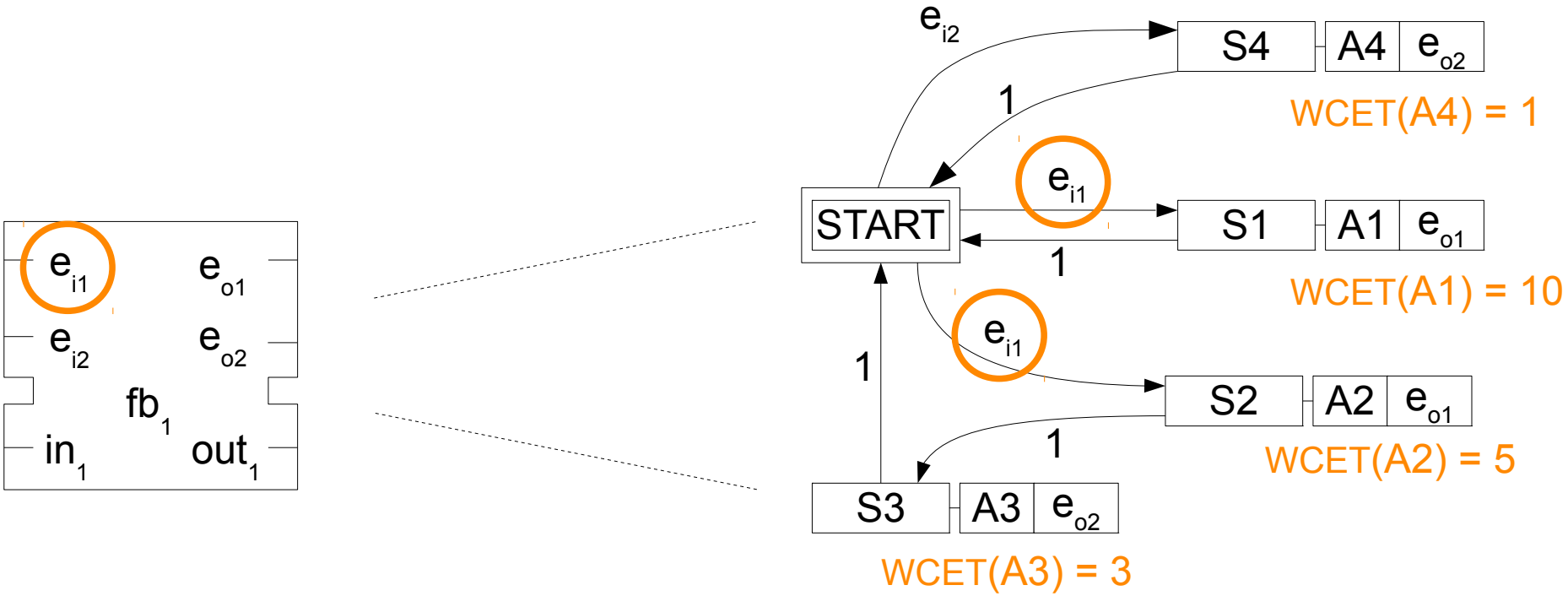
{ <p_1, WCET info> }>

Analysis algorithms

Basic FB Analysis

- For each input event port we find all ECC transitions guarded by that event
- We find all possible ECC runs that start with these transitions
- For each run we
  - Add together WCET values of all executed algorithms
  - Collect information about generated output events

Basic FB Analysis

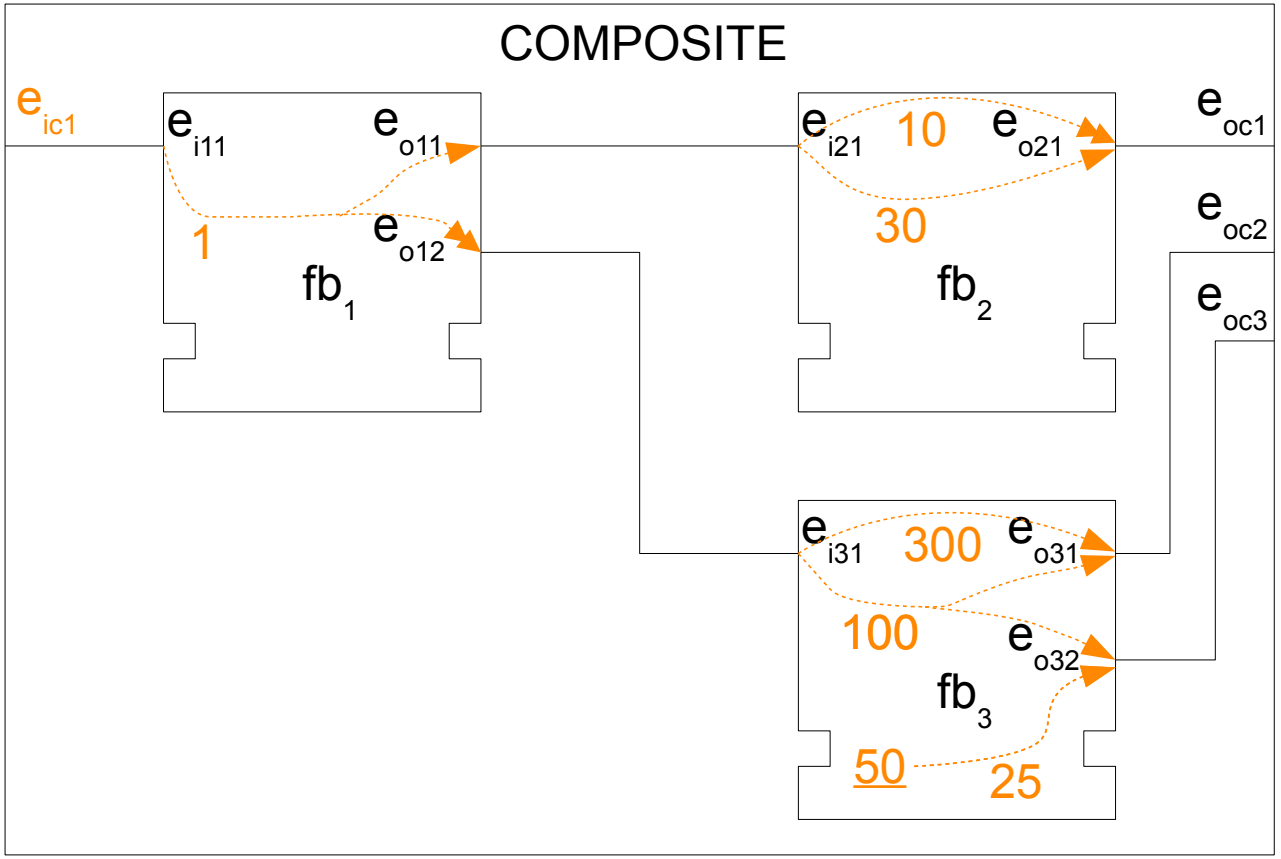


WCET = <{ <e_{i1},{<10, {e_{o1} = 1}> }>,
<8, {e_{o1} = 1, e_{o2} = 1,}> }>,
<e_{i1} , ... > }>

Composite FB Analysis

- We find all execution paths starting with
  - Each input event port
  - Each internal execution trigger of contained FBs
- Execution paths are determined by
  - Event connection of the FB network
  - Information about generated output events stored in the WCET data of contained FBs
- For each execution path we
  - Add together WCET values
  - Gather information about generated output events

Composite FB Analysis



- Each arrow represents an execution path.
- Number next to an arrow is the WCET value of the execution path.
- Number of arrowheads denotes the multiplicity of generated output events

WCET = <{ <e_{i1},{<611, {e_{oc1} = 2, e_{oc2} = 2}>,
<211, {e_{oc1} = 2, e_{oc2} = 2, e_{oc3} = 2}>,
<631, {e_{oc1} = 1, e_{oc2} = 2}>,
<231, {e_{oc1} = 1, e_{oc2} = 2, e_{oc3} = 2}> }>
<50,{ 25, {e_{oc3} = 2} }> }>

Application Analysis

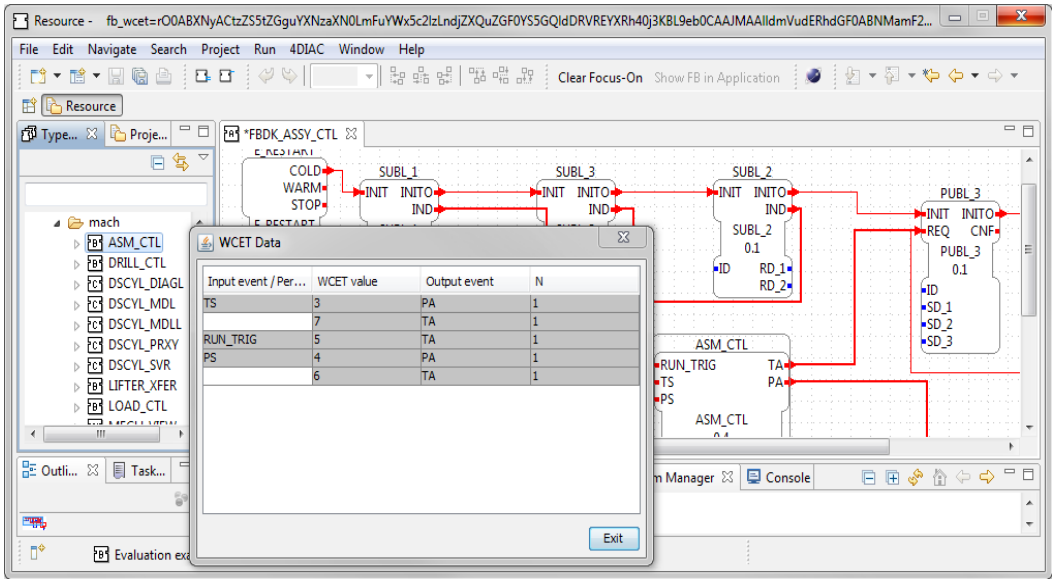
- Based on the same FB network analysis as for composites

- Applications have no input events – only internal triggers start execution

Implementation

Prototype Analysis Tool

- Goal – analysis as a part of the development environment, not a separate tool
- 4DIAC-IDE (v1.1.3) plug-in
  - Eclipse-based tool allowed for an easy integration
  - No changes were made to the 4DIAC source files
- Integrated with Type Navigator and System manager



Experiments

- Applicability of the tool demonstrated on 4 systems from open-source libraries
  - 4DIAC and FBDK examples
  - We wanted to apply the tool on existing systems, not invent our own
- Largest system – 158 FB instances, 6 levels of hierarchy
  - Analysis time ~ 10ms

Problems

- Storing WCET data together with FB models
  - No support for IEC 61499 Attributes
  - We did not want to make changes to the 4DIAC source files
  - For now, the data is stored inside FB comments
- Finding test systems
  - Most of the example systems have low complexity
  - We could not find systems with known WCET values

Future Work & Conclusion

Future Work

- Processing resource utilization analysis (ETFA'13)
  - Based on WCET analysis
- Hardware-specific WCET data
  - Provide WCET values which are specific for each device type
- Automated analysis
  - Perform the analysis with each change to the system model
- Extensive evaluation
  - Determine how close the analysis results are to real WCET values

Conclusion

- Early timing analysis using IEC 61499 system models
  - We can perform the analysis before a system is deployed or fully implemented
- Efficiency through reusing analysis results
  - Analysis results for one hierarchical level are reused when performing analysis on a higher hierarchical level
- Prototype tool integrated with the 4DIAC-IDE
  - Test show the applicability of the analysis

Thank you for your attention!

Questions