Lifting Verification Results for Preemption Statements

Manuel Gesell and Klaus Schneider
{gesell,schneider}@cs.uni-kl.de
es.cs.uni-kl.de

Embedded Systems Group
University of Kaiserslautern

11th International Conference on Software Engineering and Formal Methods, Madrid, Spain, 25-27th September 2013

Outline

1. Introduction
   - Motivation
   - Synchronous Model of Computation
   - Preemption Behavior
2. Lifting Verification Results for Preemption Statements
   - Abort Transformation
   - Suspend Transformation
3. Conclusion

Outline

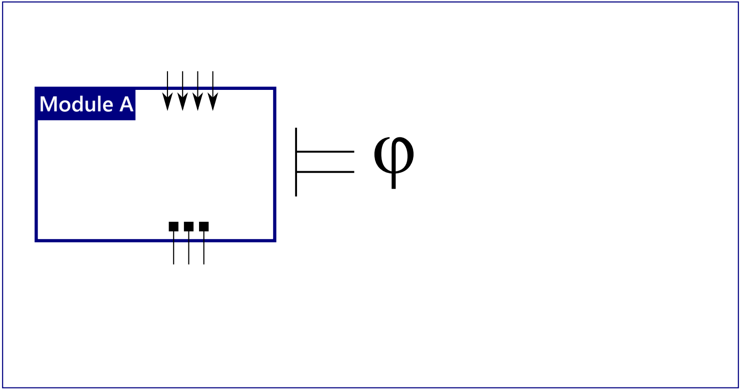
1. Introduction
   - Motivation
   - Synchronous Model of Computation
   - Preemption Behavior
2. Lifting Verification Results for Preemption Statements
3. Conclusion

Outline

1. Introduction
   - Motivation
   - Synchronous Model of Computation
   - Preemption Behavior
2. Lifting Verification Results for Preemption Statements
3. Conclusion

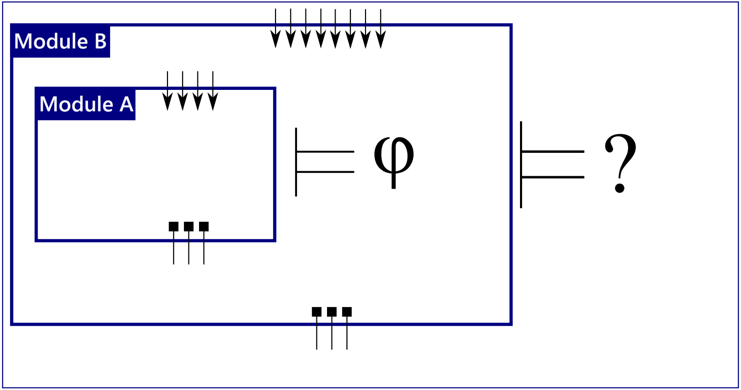
Motivation

interactive proof rules for module calls in synchronous programs

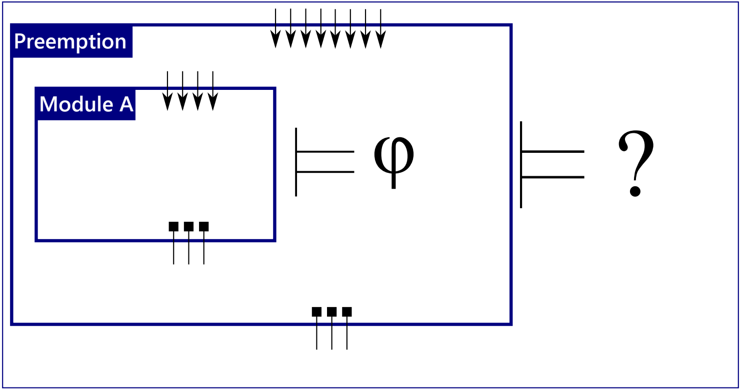


Motivation

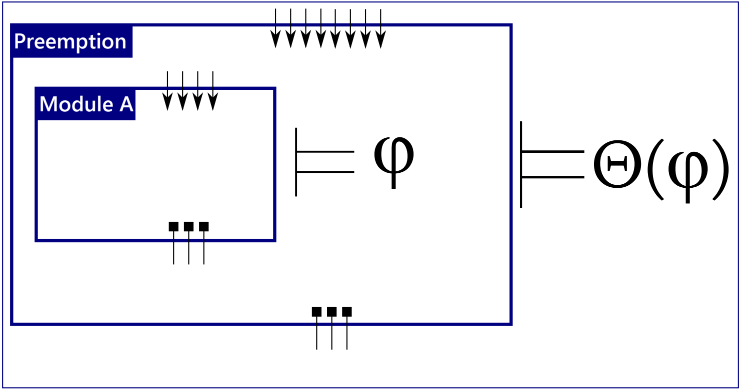
interactive proof rules for module calls in synchronous programs



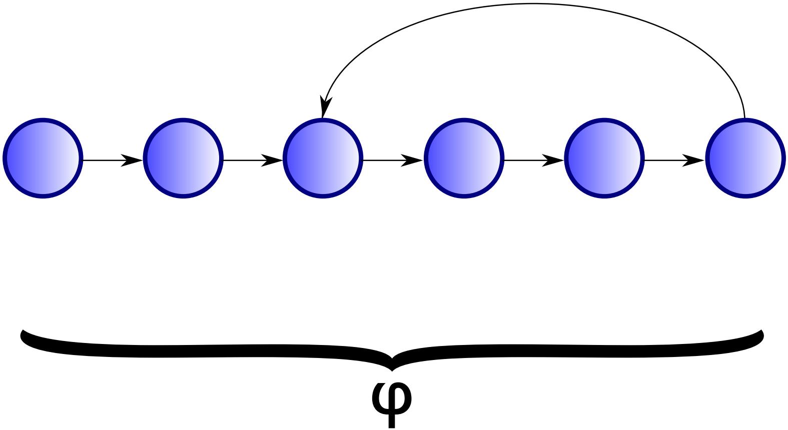
What is this talk about?



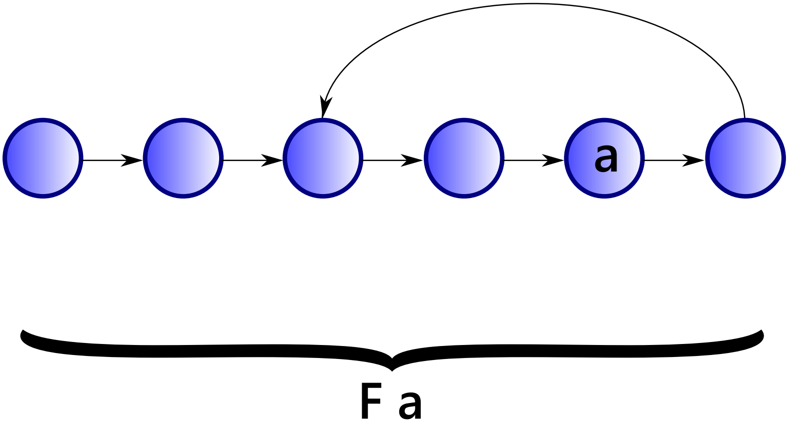
What is this talk about?



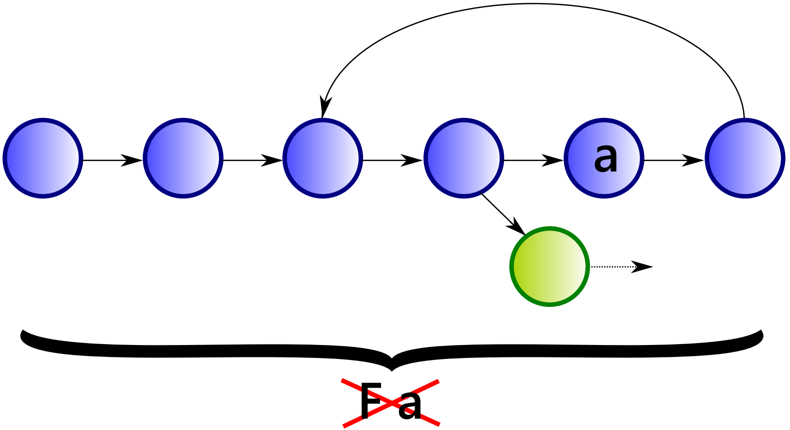
Influence of Preemption on Modules



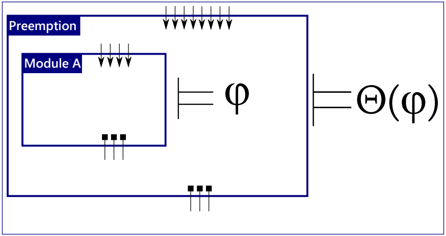
Influence of Preemption on Modules



Influence of Preemption on Modules



What is this talk about?



Approach

- restriction to preemption specific behavior
- step wise application possible
- preemption-specific Θ
- specification should preserved 'as much as possible'
- correct by construction

Outline

1. Introduction
   - Motivation
   - Synchronous Model of Computation
   - Preemption Behavior
2. Lifting Verification Results for Preemption Statements
3. Conclusion

Synchronous Model of Computation

- execution is divided into a sequence of reactions steps
- behavior in a reaction step
  - starts/ends in a **pause**
  - all inputs are read
  - all outputs are produced (instantaneously)
  - new internal state is determined
  - each variable has a unique value

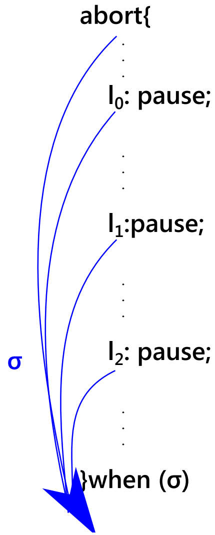
Outline

Example

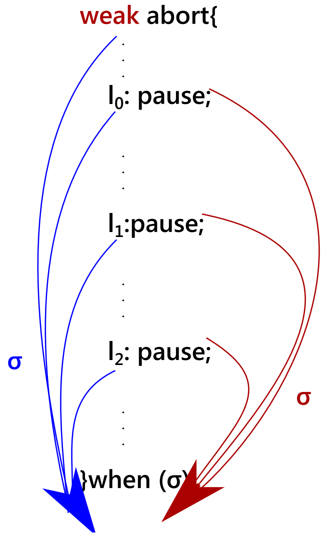
`abort` behavior

- execution of nested program part is stopped

Example



Example

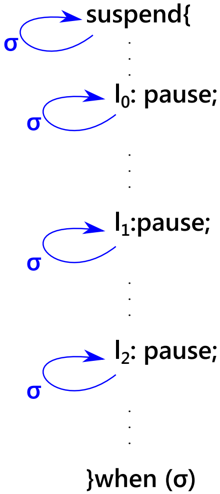


Example

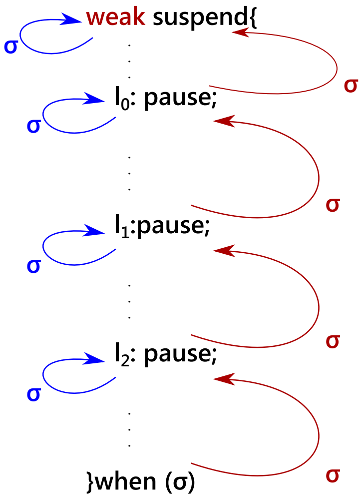
suspend behavior

- execution of nested program part is (infinitely) postponed

Example



Example

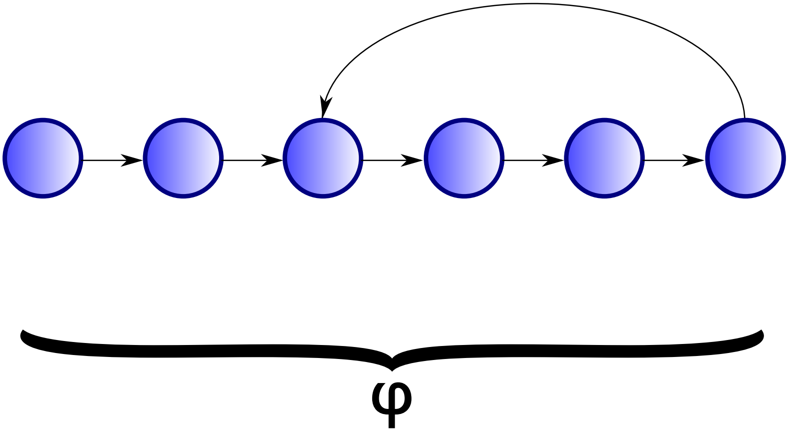


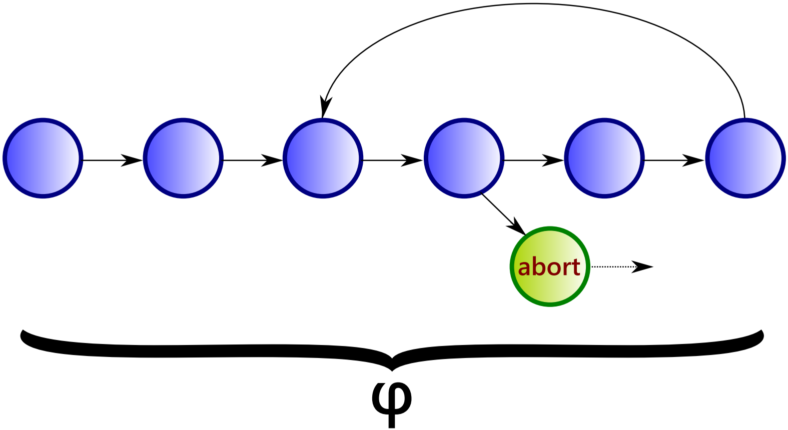
Outline

Outline

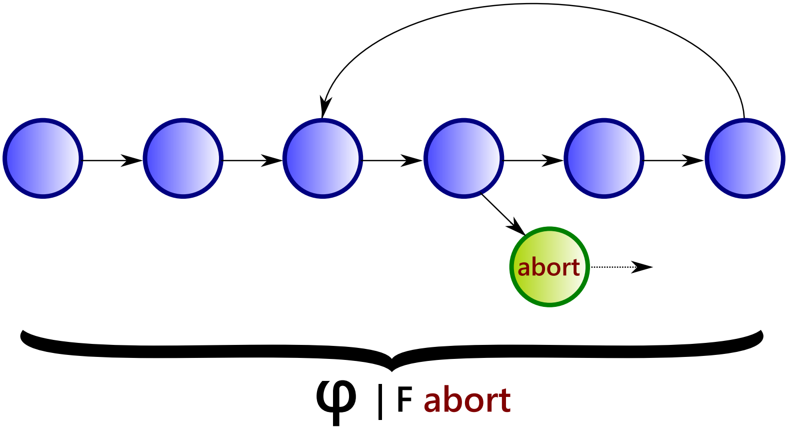
Idea

- **abort** stops execution
- preemptive specification preserves 'as much as possible'
- preemptive specification $\Theta^{st}_{ab}(\varphi)$ for φ is satisfied
  - φ already satisfied
  - abortion in a state not violating φ

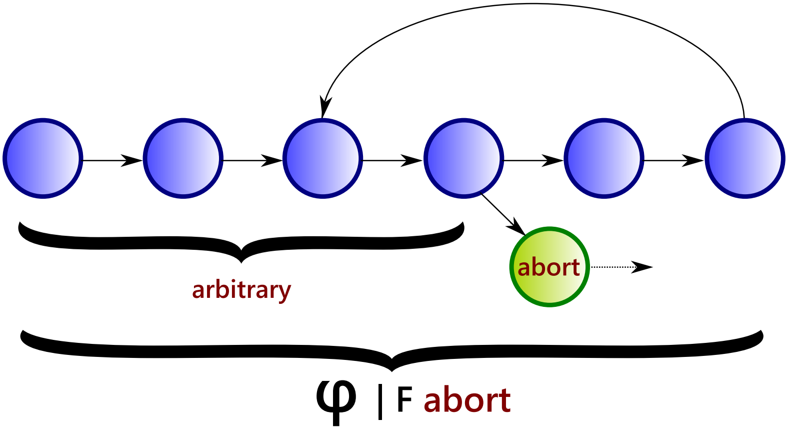

φ


abort
φ

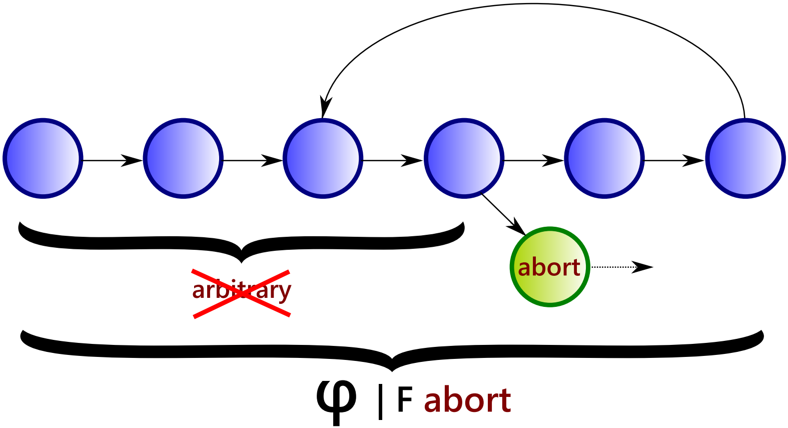
Naive Transformation #1 (too weak)



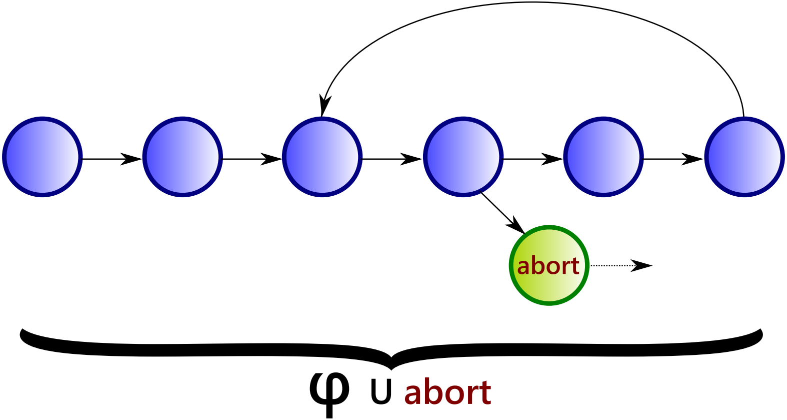
Naive Transformation #1 (too weak)



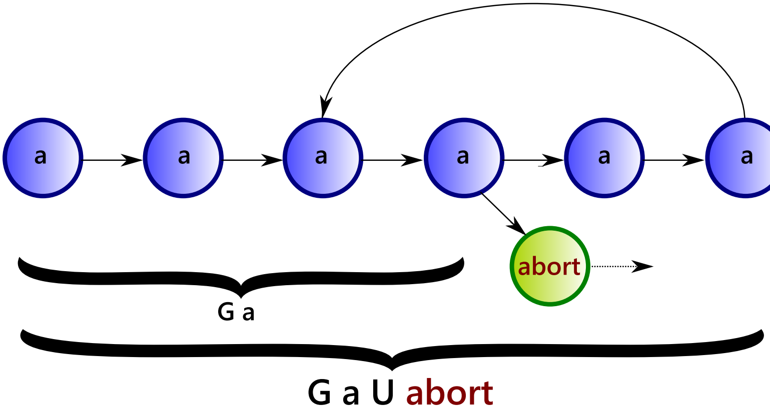
Naive Transformation #1 (too weak)



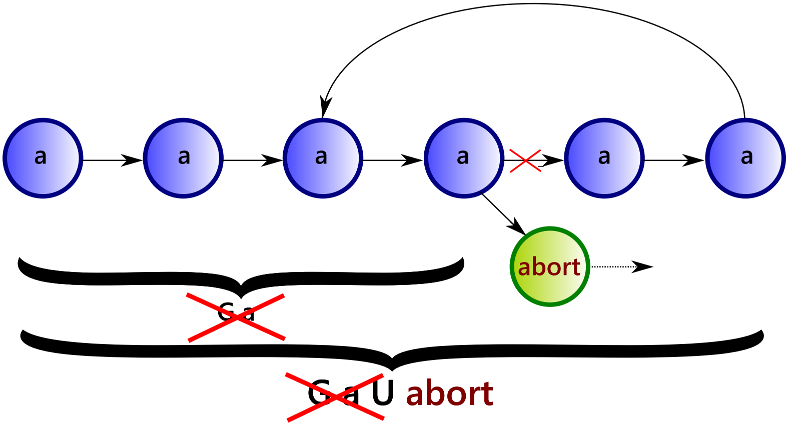
Naive Transformation #2 (too strong)



Naive Transformation #2 (too strong)



Naive Transformation #2 (too strong)



Correct Transformation for Abort

Transformation $\Theta^{st}_{ab}(\varphi, \sigma)$

The transformation $\Theta^{st}_{ab}(\varphi, \sigma)$ that generates an **abort**-sensitive specification for $A\varphi$ given in NNF is defined recursively as

$$\Theta^{st}_{ab}(\varphi, \sigma) := \begin{cases} \sigma \vee \varphi, & \text{if } \varphi \text{ is propositional} \\ \sigma \vee \mathbf{X}(\Theta^{st}_{ab}(\psi, \sigma)), & \text{if } \varphi = \mathbf{X}\psi \\ \Theta^{st}_{ab}(\psi, \sigma) \otimes \Theta^{st}_{ab}(\gamma, \sigma), & \text{if } \varphi = \psi \otimes \gamma \text{ with} \\ & \otimes \in \{\underline{\mathsf{U}}, \mathsf{U}, \wedge, \vee\}. \end{cases}$$

Examples

$$\begin{array}{lclclcl} \Theta^{st}_{ab}(\mathsf{Xa}, \sigma) & \equiv & \sigma \vee \Theta^{st}_{ab}(\mathsf{Xa}, \sigma) & \equiv & \sigma \vee \mathsf{X}(\sigma \vee \mathsf{a}) & & \\ \Theta^{st}_{ab}(\mathsf{Ga}, \sigma) & \equiv & \Theta^{st}_{ab}([\mathsf{a}\ \mathsf{U}\ 0], \sigma) & \equiv & [(\sigma \vee \mathsf{a})\ \mathsf{U}\ \sigma] & \equiv & [\mathsf{a}\ \mathsf{U}\ \sigma] \\ \Theta^{st}_{ab}(\mathsf{Fa}, \sigma) & \equiv & \Theta^{st}_{ab}([1\ \underline{\mathsf{U}}\ \mathsf{a}], \sigma) & \equiv & [1\ \underline{\mathsf{U}}\ (\mathsf{a} \vee \sigma)] & \equiv & \mathsf{F}(\mathsf{a} \vee \sigma) \end{array}$$

Interactive Proof Rule

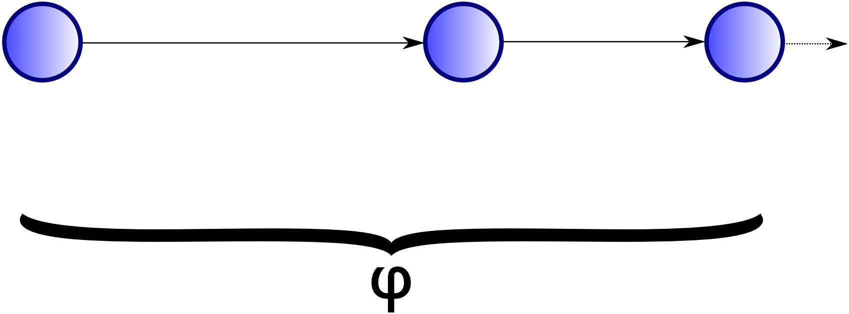
$$\frac{M \models \varphi}{\texttt{\textbf{abort}}\{\texttt{M}\}\texttt{\textbf{when}}(\sigma) \models \Theta^{\mathsf{st}}_{\mathsf{ab}}(\varphi, \sigma)}$$

Outline

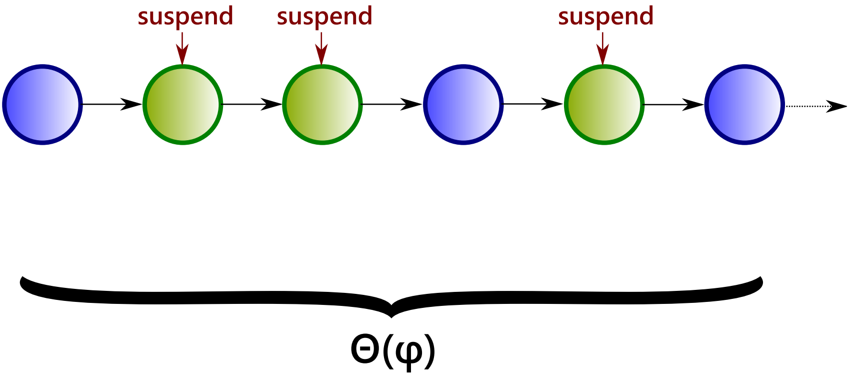
Idea

- **suspend** postpones execution
- preemptive specification $\Theta^{\mathrm{st}}_{\mathrm{sp}}(\varphi)$ for φ is satisfied
  - ignoring suspended states satisfies φ
  - possibility of infinite suspension

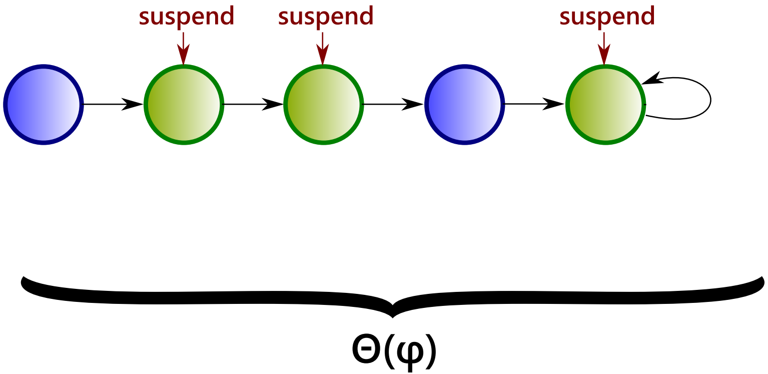
Idea for Suspend



Idea for Suspend



Idea for Suspend



Correct Transformation for Suspend

Transformation $\Theta_{sp}^{st}(\varphi, \sigma)$

For a given specification $\mathsf{A}\varphi$, the transformation $\Theta_{sp}^{st}(\varphi, \sigma)$ is defined as

$$\Theta_{sp}^{st}(\varphi,\sigma) := \begin{cases} [\varphi \ \mathsf{W} \ \neg\sigma], & \text{if } \varphi \text{ is propositional} \\ \left[(\mathbf{X}\Theta_{sp}^{st}(\psi,\sigma)) \ \mathsf{W} \ \neg\sigma\right], & \text{if } \varphi = \mathbf{X}\psi \\ \Theta_{sp}^{st}(\psi,\sigma) \otimes \Theta_{sp}^{st}(\gamma,\sigma), & \text{if } \varphi = \psi \otimes \gamma \text{ with} \\ & \otimes \in \{\underline{\mathsf{U}}, \mathsf{U}, \wedge, \vee\}. \end{cases}$$

Examples

$$\begin{aligned} \Theta_{sp}^{st}(\mathsf{Xa},\sigma) &\equiv \left[(\mathsf{X}\Theta_{sp}^{st}(\mathsf{a},\sigma)) \ \mathsf{W} \ \neg\sigma\right] \equiv [(\mathsf{X}\,[\mathsf{a} \ \mathsf{W} \ \neg\sigma]) \ \mathsf{W} \ \neg\sigma] \\ \Theta_{sp}^{st}(\mathsf{Ga},\sigma) &\equiv \Theta_{sp}^{st}([\mathsf{a} \ \mathsf{U} \ 0],\sigma) \equiv [[\mathsf{a} \ \mathsf{W} \ \neg\sigma] \ \mathsf{U} \ 0] \equiv \mathsf{G}\,[\mathsf{a} \ \mathsf{W} \ \neg\sigma] \\ \Theta_{sp}^{st}(\mathsf{Fa},\sigma) &\equiv \Theta_{sp}^{st}([1 \ \underline{\mathsf{U}} \ \mathsf{a}],\sigma) \equiv [1 \ \underline{\mathsf{U}} \ [\mathsf{a} \ \mathsf{W} \ \neg\sigma]] \equiv \mathsf{F}\,[\mathsf{a} \ \mathsf{W} \ \neg\sigma] \end{aligned}$$

Interactive Proof Rule

$$\frac{M \models \varphi}{\textbf{suspend}\{\texttt{M}\}\textbf{when}(\sigma) \models \Theta_{\mathsf{sp}}^{\mathsf{st}}(\varphi, \sigma)}$$

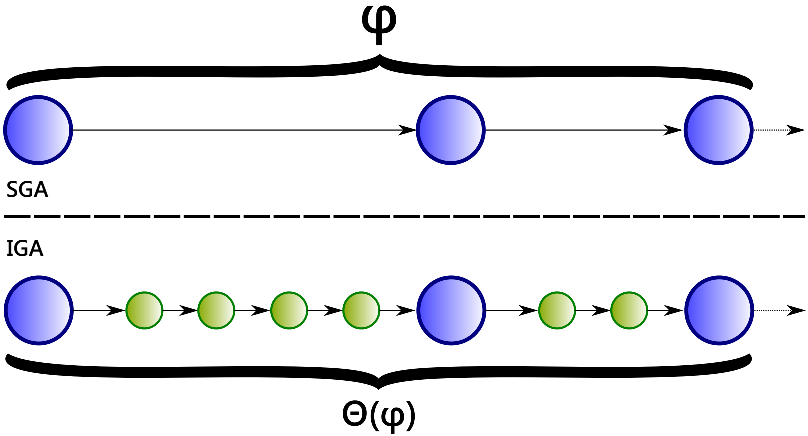
Outline

Summary

- synchronous model of computation
- preemption behavior
- reuse of verification results in a preemption context
- automatic and correct-by-construction transformation

Application of Suspend Transformation

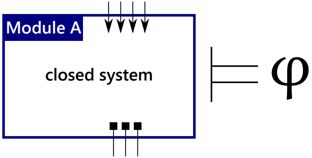
- Clock refinement
- Translation from synchronous to interleaved guarded actions



The End

Questions?

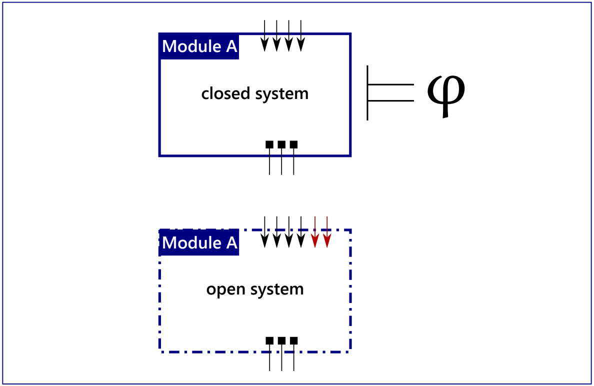
Already Published Approach



[GeSc13a]

M. Gesell and K. Schneider
Modular Verification of Synchronous Programs, ACSD 2013

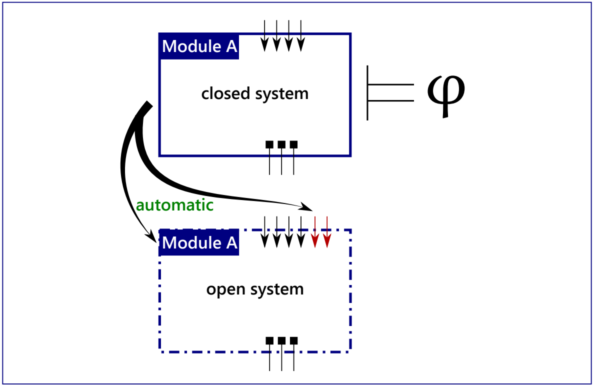
Already Published Approach



[GeSc13a]

M. Gesell and K. Schneider
Modular Verification of Synchronous Programs, ACSD 2013

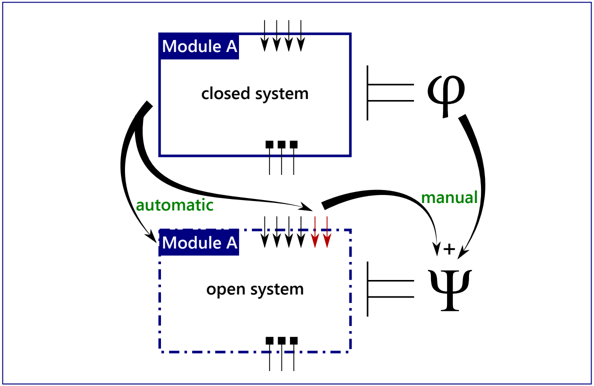
Already Published Approach



[GeSc13a]

M. Gesell and K. Schneider
Modular Verification of Synchronous Programs, ACSD 2013

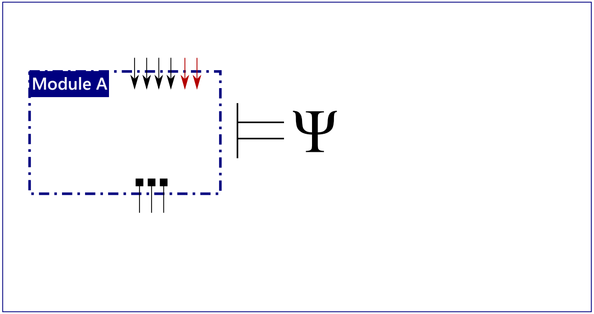
Already Published Approach



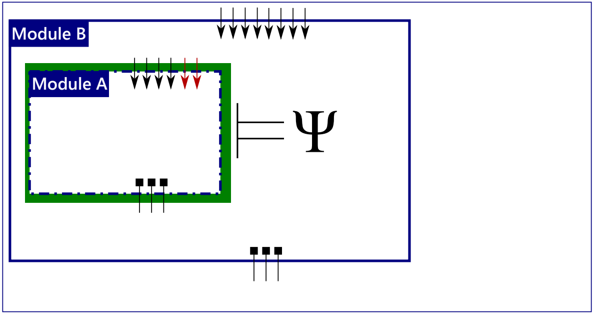
[GeSc13a]

M. Gesell and K. Schneider
Modular Verification of Synchronous Programs, ACSD 2013

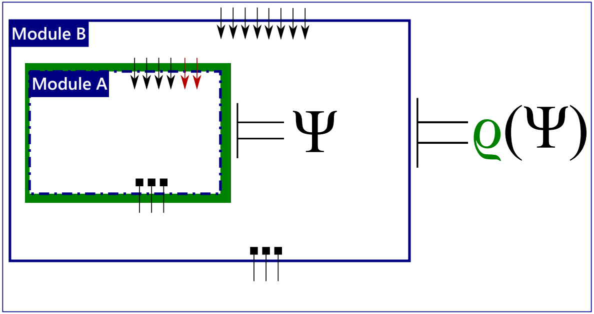
Already Published Approach



Already Published Approach



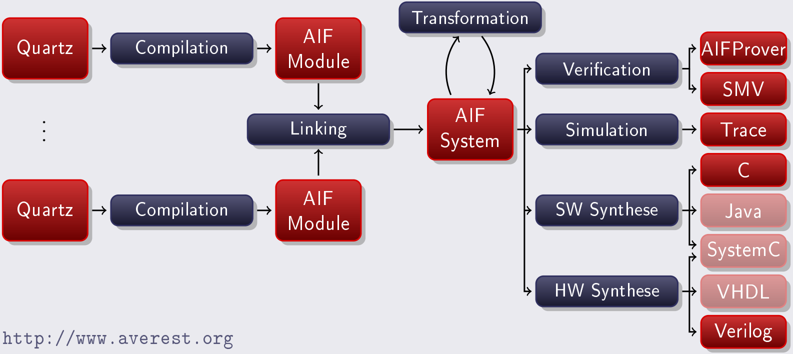
Already Published Approach



Advantages and Disadvantages

+ reuse of verification results
– manual transition step
– additional verification task
○ no necessary relation to the original specification

Averest Design Flow



http://www.averest.org