A Model-driven development and verification framework for embedded software (3)

Yunja Choi

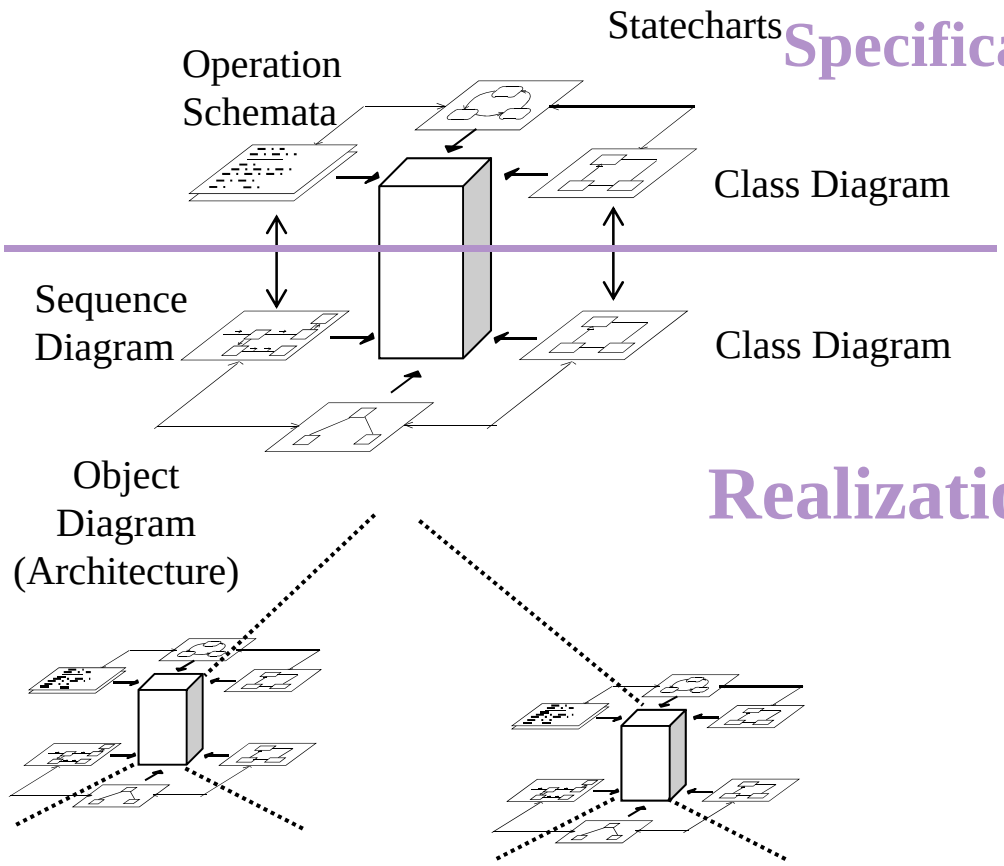
Software Safety Engineering Laboratory
School of Electrical Engineering and Computer Science
Kyungpook National University

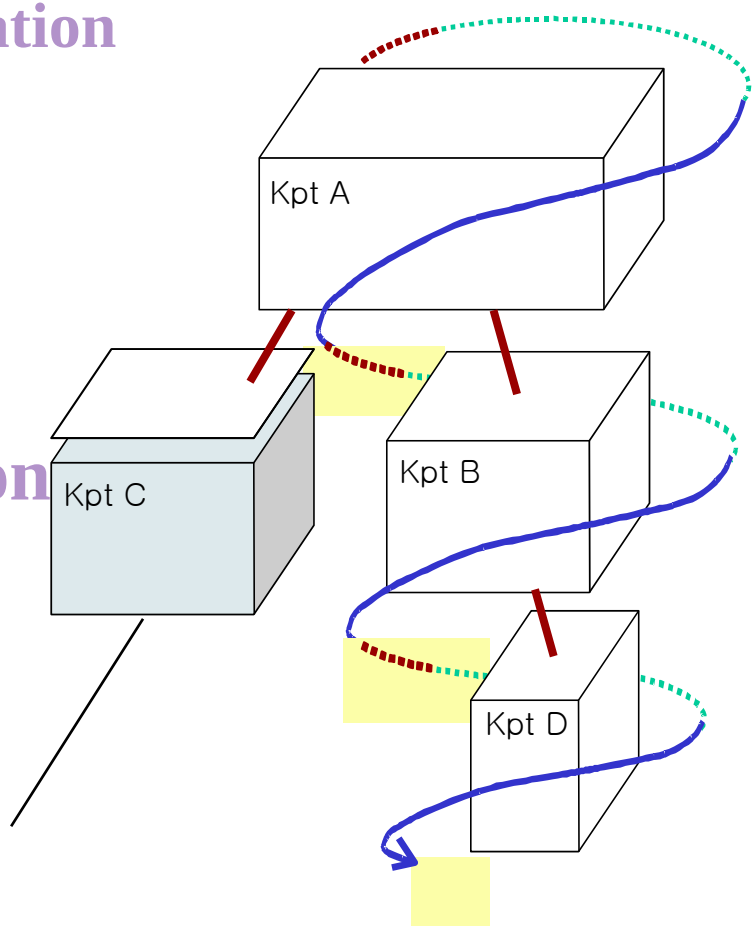
Work in progress

- Model-driven and component-based design verification
  - Supporting tool for design verification
  - Performance improvement using component control model
- Safety analysis of OSEK/VDX for automobile software

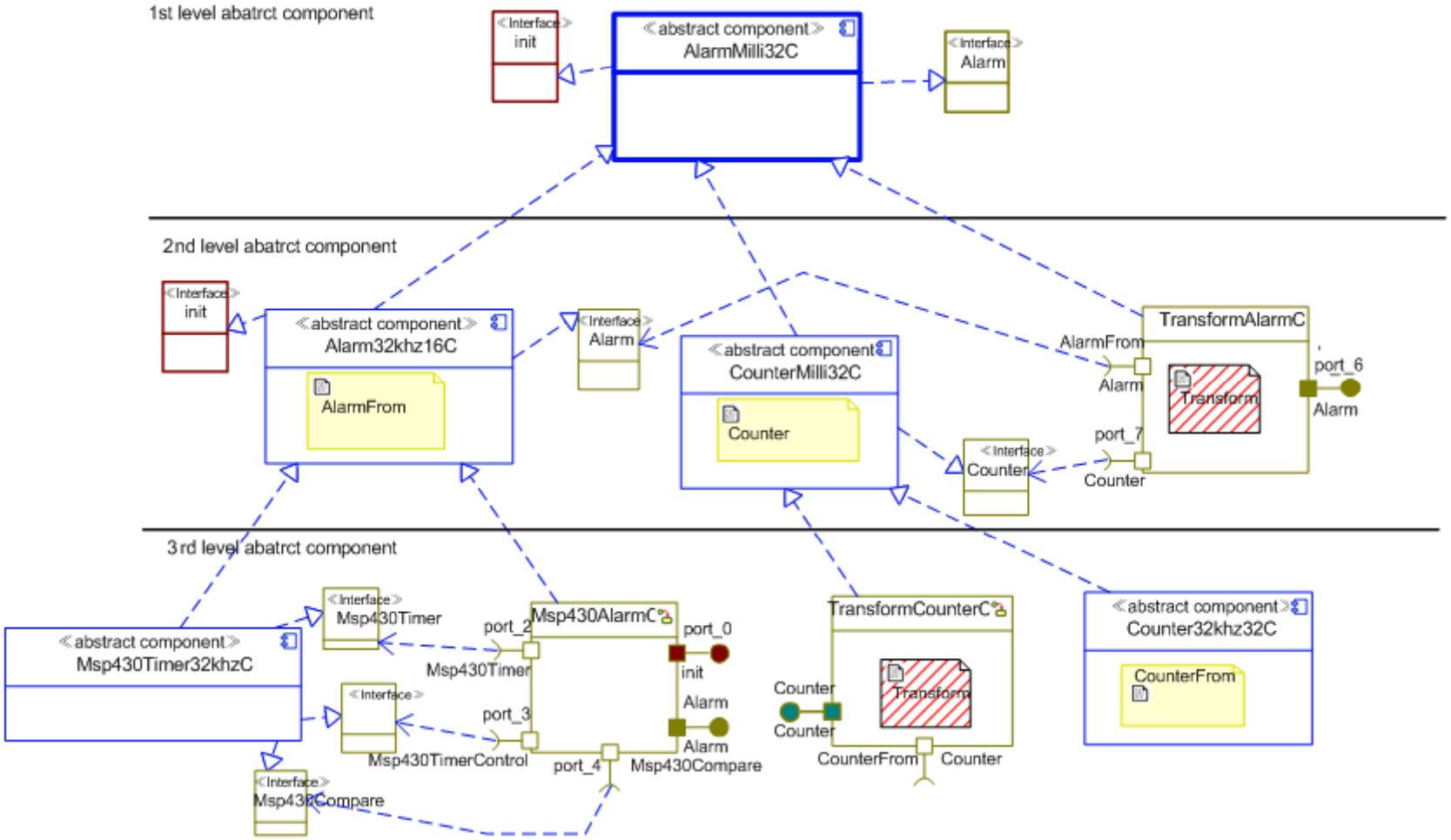
Model-driven and component-based design verification

Model-driven and component-based design verification

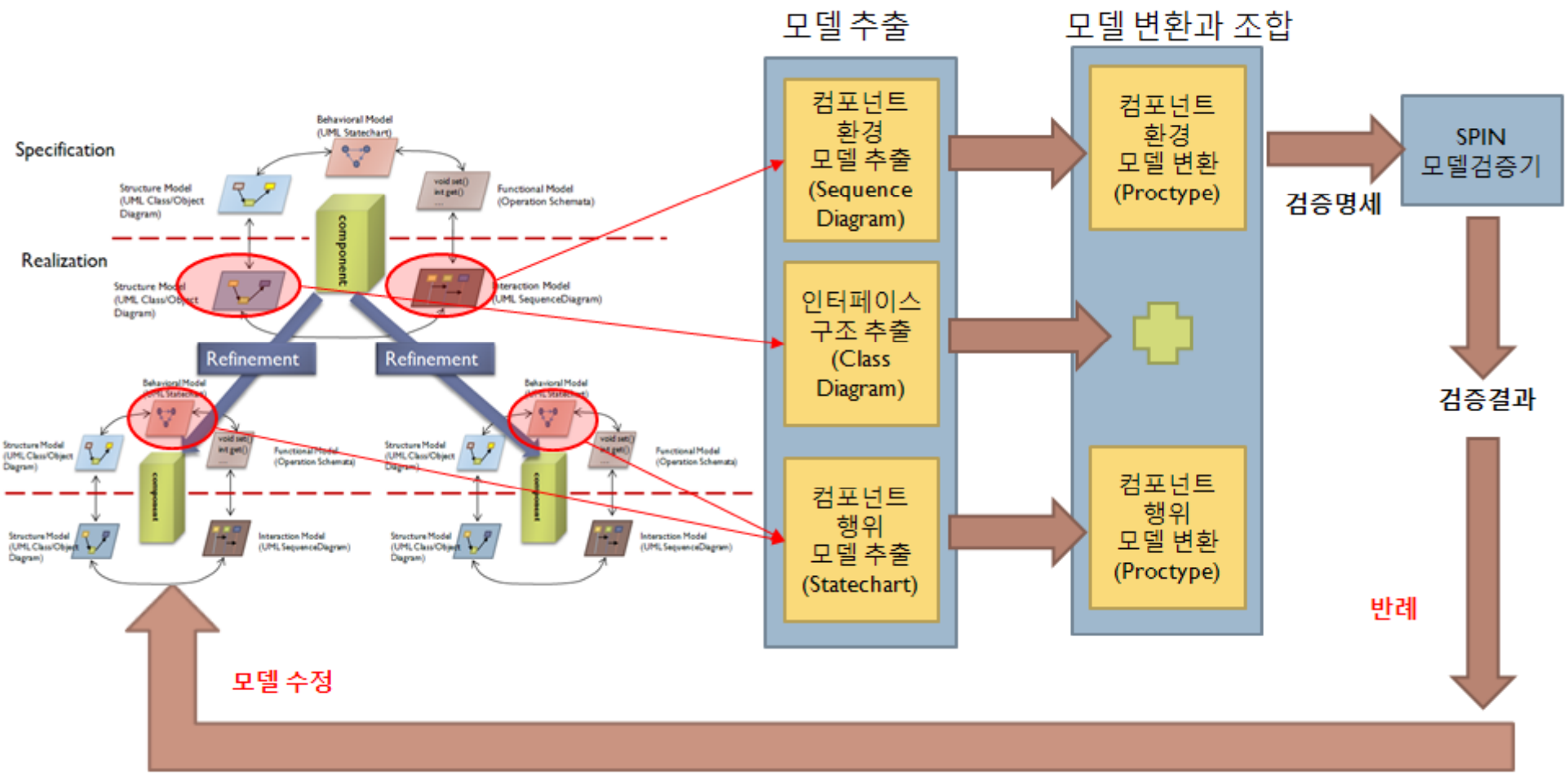




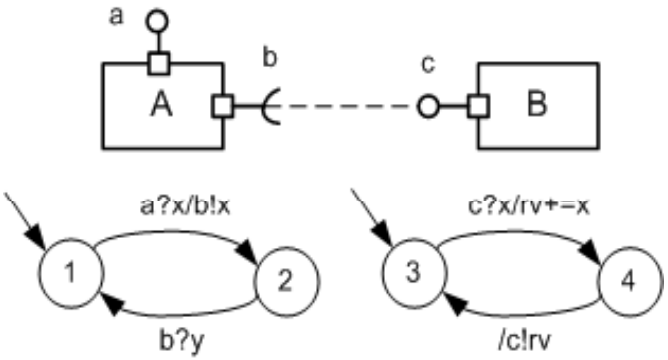
Example


1st level abatrct component
«Interface» init
«abstract component» AlarmMilli32C
«Interface» Alarm
2nd level abatrct component
«Interface» init
«abstract component» Alarm32khz16C
AlarmFrom
«Interface» Alarm
«abstract component» CounterMilli32C
Counter
«Interface» Counter
TransformAlarmC
AlarmFrom
Alarm
Transform
port_6
Alarm
port_7
Counter
3rd level abatrct component
«abstract component» Msp430Timer32khzC
«Interface» Msp430Timer
«Interface» Msp430TimerControl
«Interface» Msp430Compare
Msp430AlarmC
port_2
Msp430Timer
port_3
port_4
port_0
init
Alarm
Alarm
Msp430Compare
TransformCounterC
Counter
Counter
Transform
CounterFrom
Counter
«abstract component» Counter32khz32C
CounterFrom

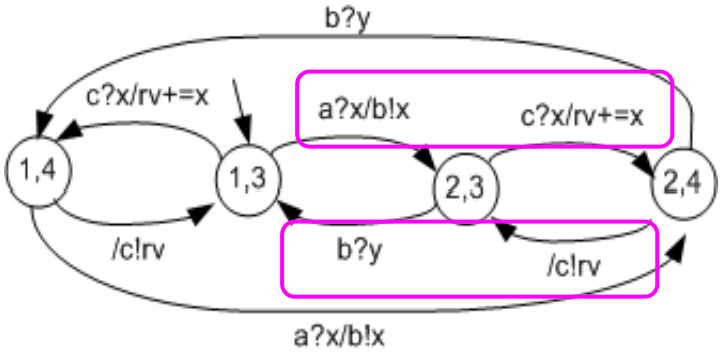
Supporting tools



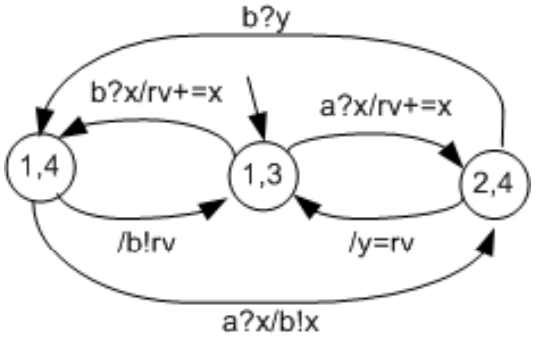
Construction of abstract behavior (1)



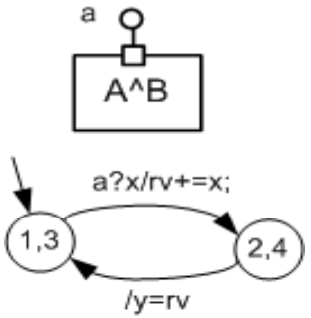
(a) two dependent abstract components



(b) free composition

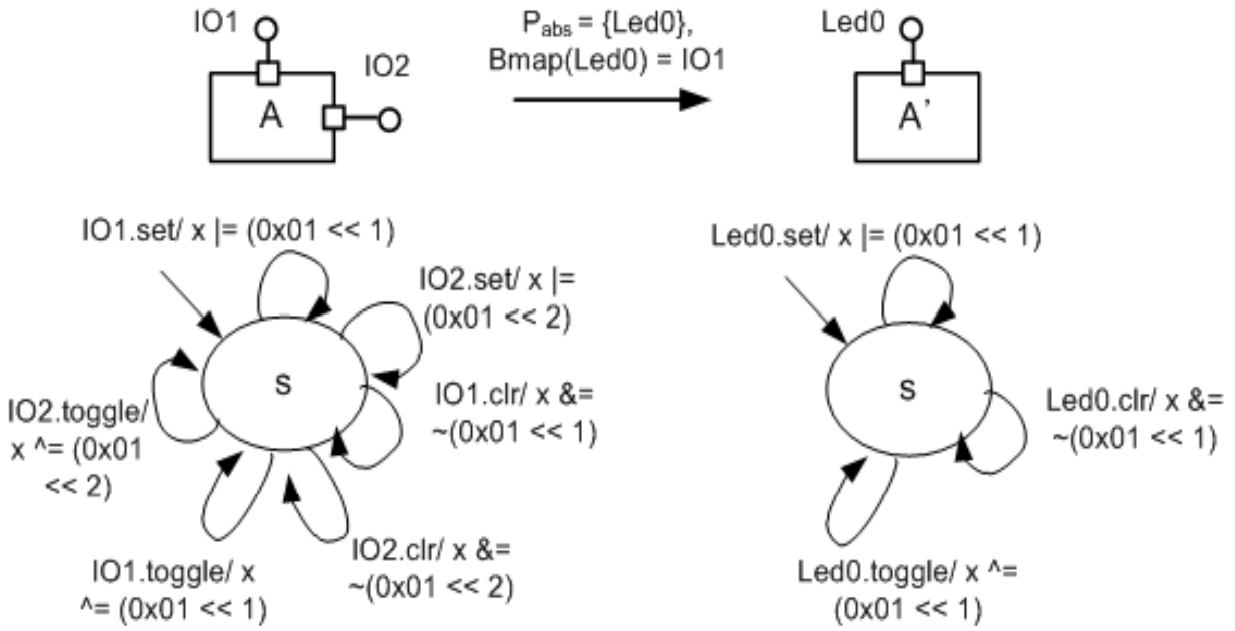


(c) synchronized reduction

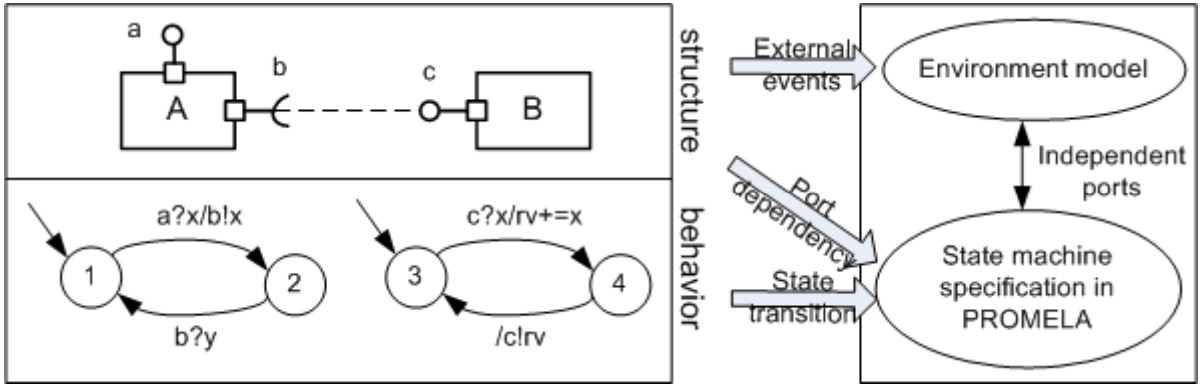


(d) abstraction

Construction of abstract behavior (2)



Checking communication consistency

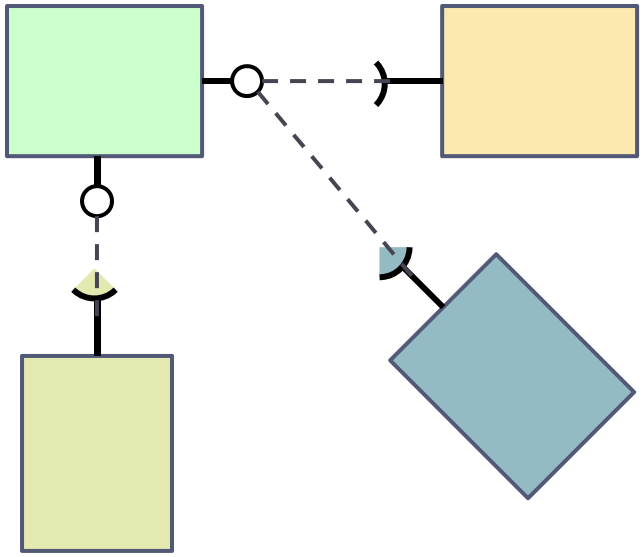


Performance

name	type	depth	states	transitions	Memory (M bytes)	Time (Seconds)
Msp430Timer32khzC	realization	1,494,716	2.4e+07	1.22e+08	5,034.0 (997.5)	1.69e+03 (3.46e+03)
	specification	392,826	8e+06	2.92e+07	843.5 (300.4)	224 (649)
Alarm32khz16C	realization	9,047	4.3e+07	2.02e+08	14,492.5 (1,488.9)	3.97e+03 (7.56e+03)
	specification	761	283,461	438,704	15.5 (6.7)	2.44 (7.8)

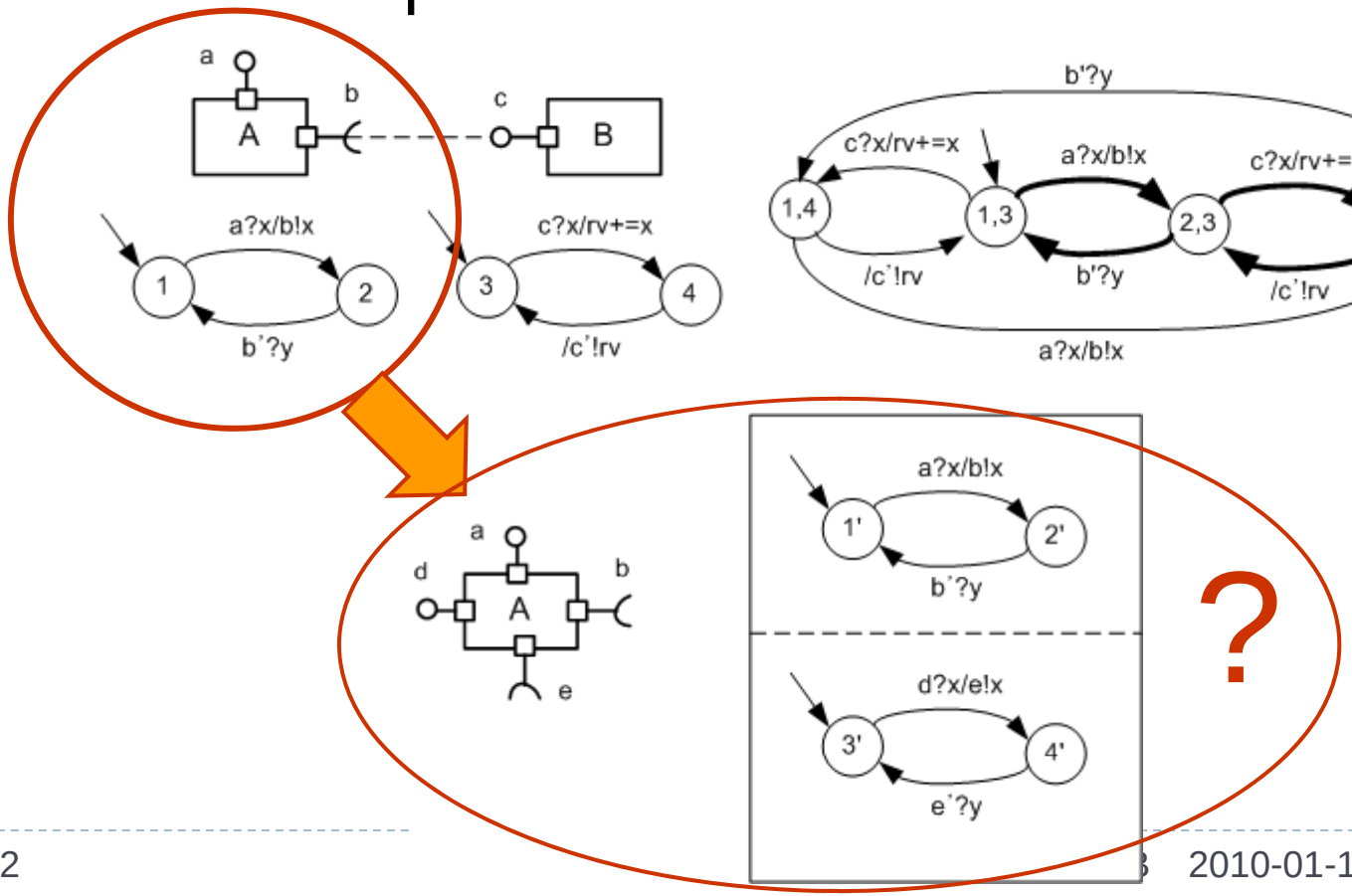
Issues

- Abstraction is abstraction
  - Timing and communication-related issues are ignored
- Still need compositions of components in the same abstraction level
  - A systematic method for the reuse of components is desired



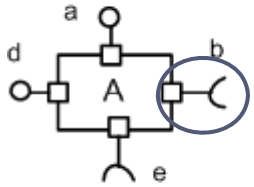
Control models for synchronous call

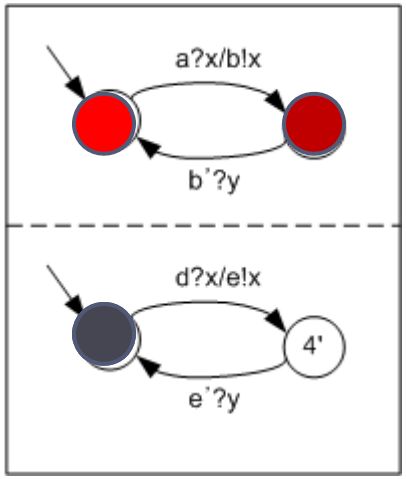
- Explicit modeling for synchronous calls can reduce verification complexity while maintaining the behavior of individual components

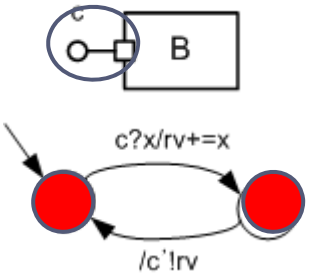


Control models for synchronous call

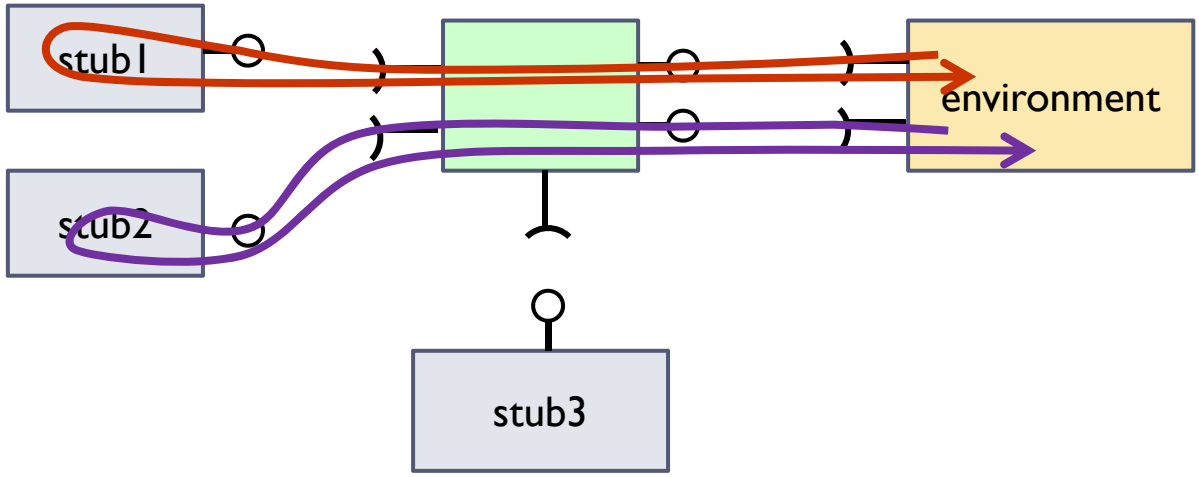
- Synchronous call to external services activates external components and deactivates itself until its return message arrives
- Assumption : one processor model



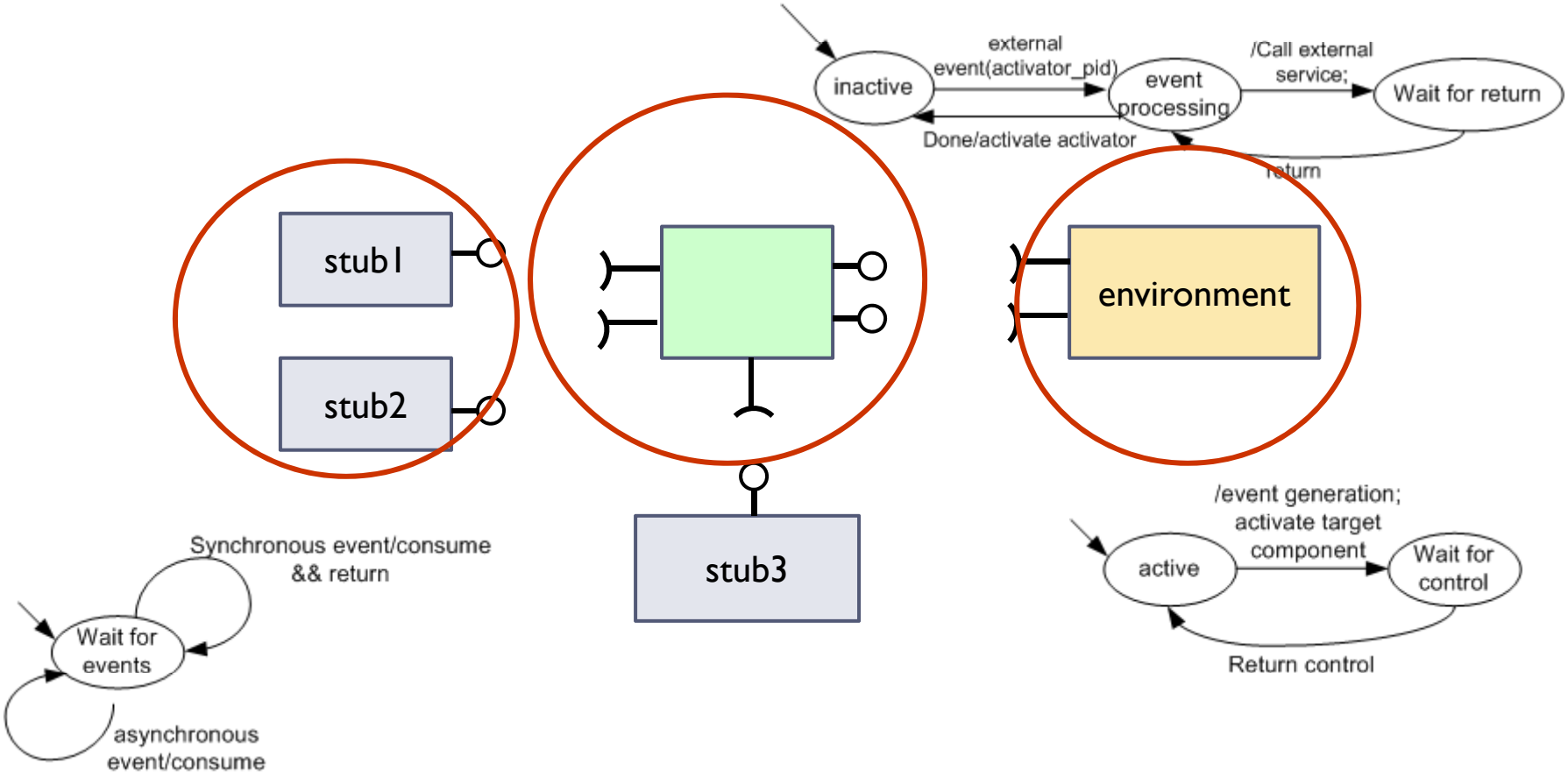




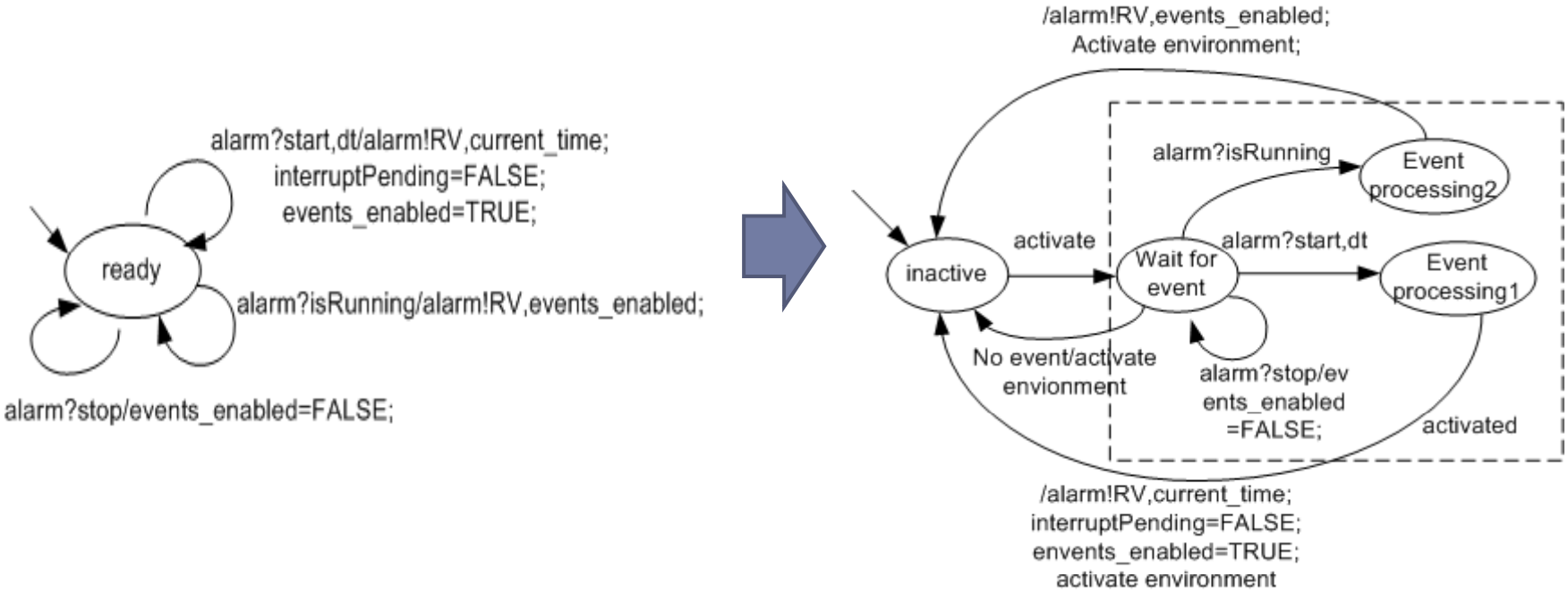
Control models for synchronous call



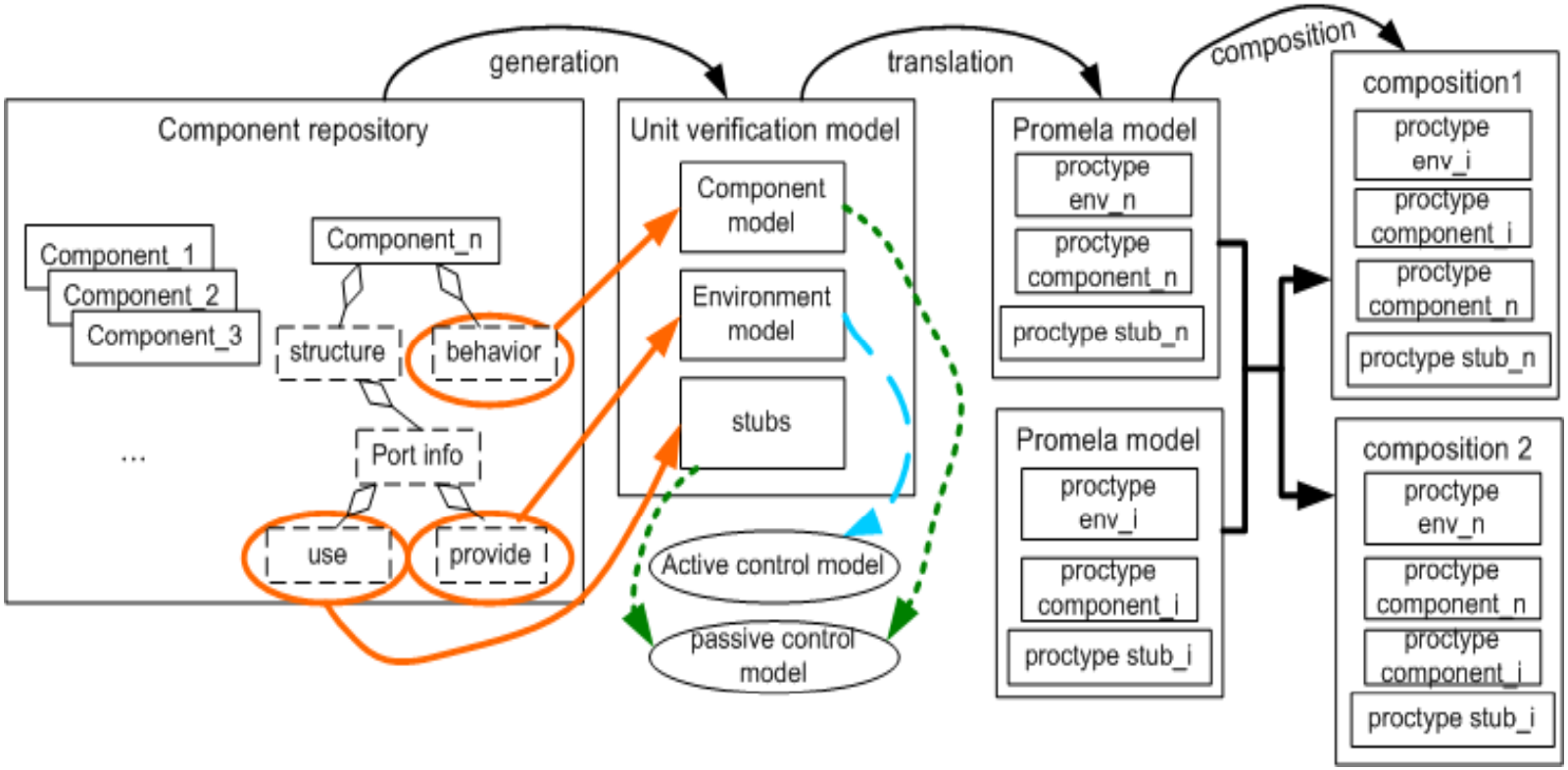
Control models for synchronous call



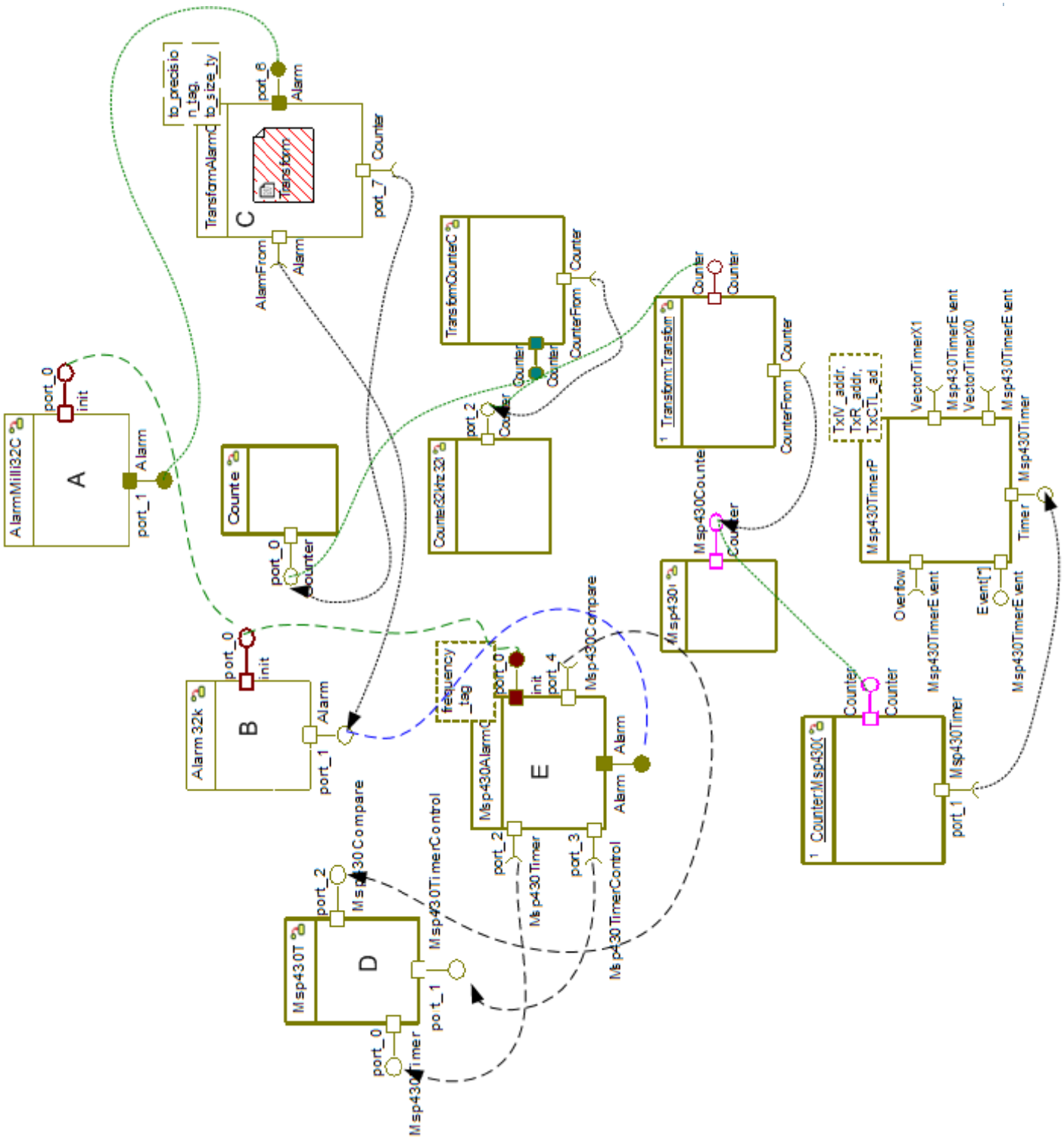
Control models for synchronous call



Verification framework



Experiments



Experiments

name	type	depth	states	transitions	Memory	Time
E	unit	1,302	1,891,489	2,366,210	112	11
D	unit	8,361	162,844	2.39E+05	8.85	0.17
B	composition	214,157	8,605,951	1.33E+07	819.5	16.3
A	composition	N/A	N/A	N/A	N/A	N/A

<models without consideration of synchronous calls >

name	type	Depth	states	transitions	Memory	Time
E	Unit	2,715	101,694	135,115	95.7	0.35
D	Unit	91	758	1.14E+03	2.5	0.001
B	Composition	763	12,227	1.94E+04	3.6	0.018
A	Composition	43,436	268,819	388.931	83.0	0.43

< models composed with the control models for synchronous calls>

Next step

- Need to update the translator for full automation
- more extensive experiments with TinyOS
- Application to OSEK/VDX
  - Model-based safety analysis using model checking