MODELING AND SYNTHESIS OF A DYNAMIC AND PARTIAL RECONFIGURATION CONTROLLER

S. Guillet, F. de Lamotte*, N. Le Griguer*, É. Rutten**, J.-P. Diguet*, G. Gogniat**

**Lab-STICC, Université de Bretagne Sud, France, {sebastien.guillet,florent.lamotte, nicolas.le-griguer,jean-philippe.diguet,guy.gogniat}@univ-ubs.fr*

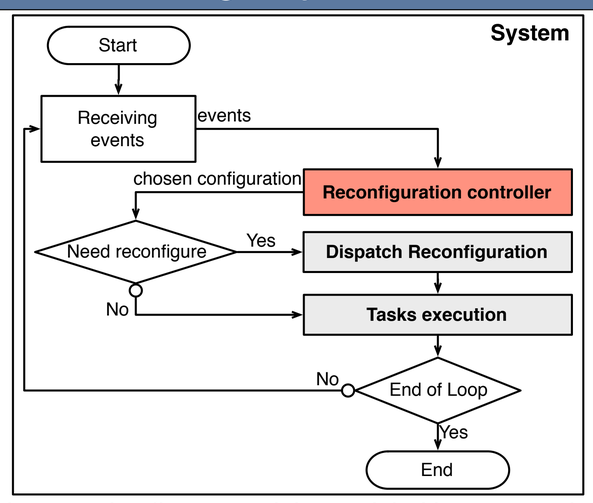
***LIG / INRIA Rhône-Alpes, France, eric.rutten@inria.fr*

ANR FAMOUS

Introduction

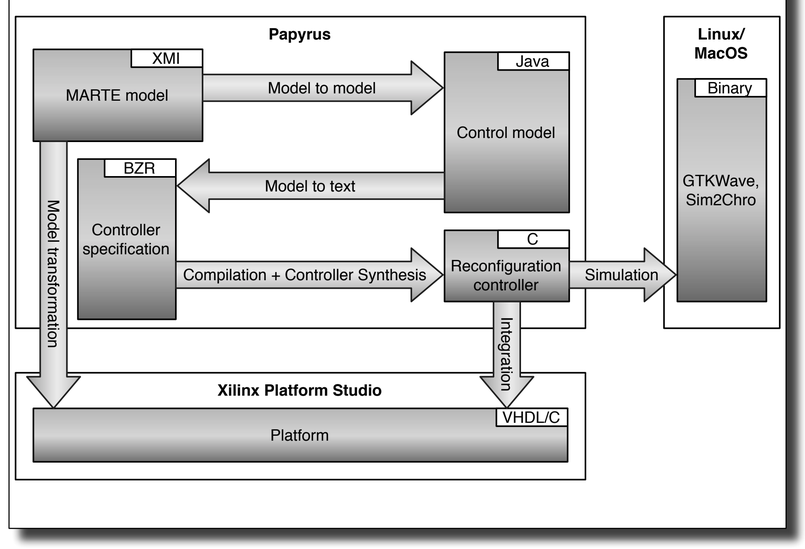
- Control specification : a transversal problem in reconfigurable SoC design
- How to make it easier to specify, and how to make it safe ?
  - Definition of a Model Driven Engineering based approach
  - Usage of formal methods, and integration into the design flow

Target systems



Generic execution flow, containing a global loop managing the system configuration upon the reception of events.

Methodology



MARTE modeling

Weights valuations

«configuration» HighMode
«Configuration» mode=[High]
«nfp» : qos = {value=5}
«nfp» : power = {value=8}

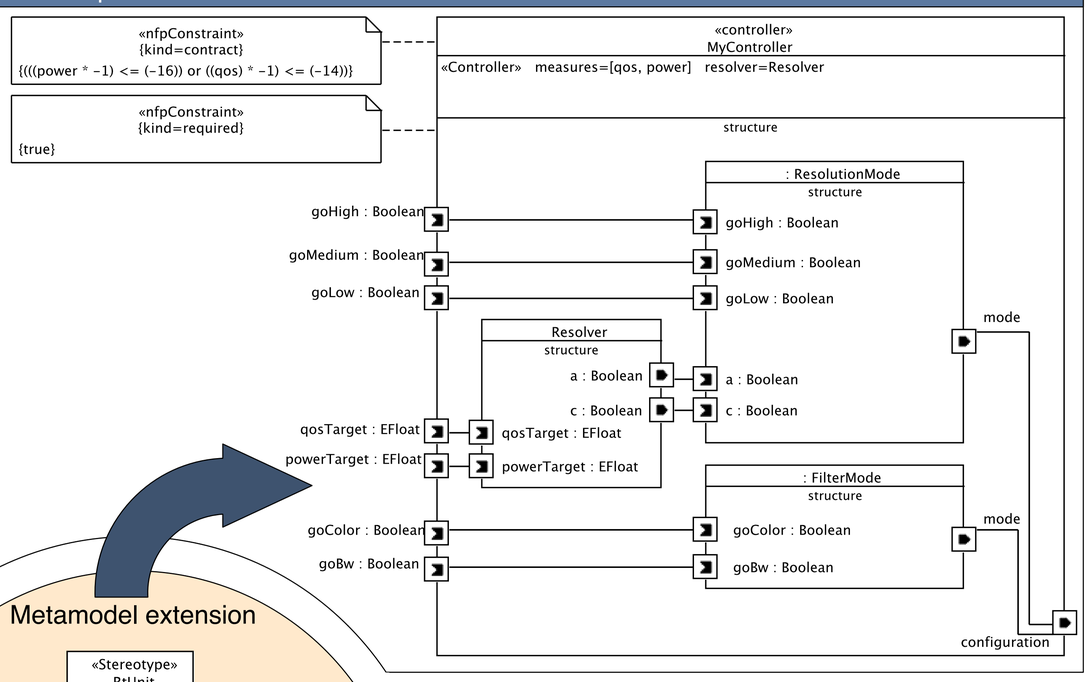
«configuration» MediumMode
«Configuration» mode=[Medium]
«nfp» : qos = {value=10}
«nfp» : power = {value=20}

«configuration» LowMode
«Configuration» mode=[Low]
«nfp» : qos = {value=11}
«nfp» : power = {value=17}

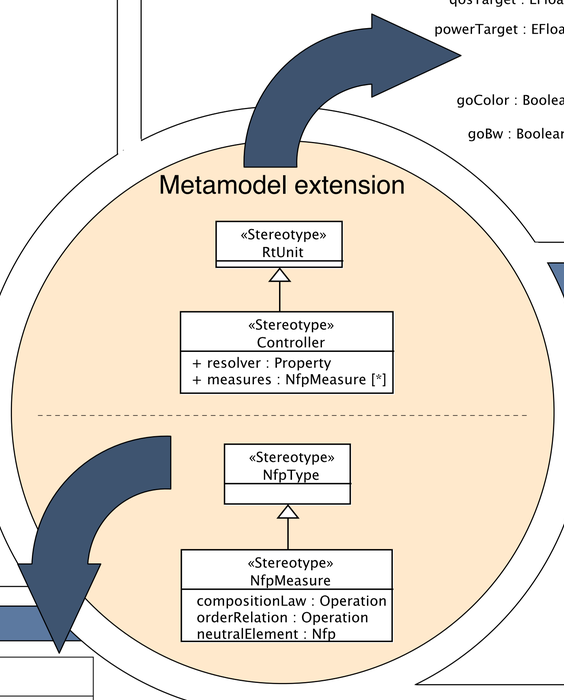
«configuration» ColorMode
«Configuration» mode=[Color]
«nfp» : qos = {value=5}
«nfp» : power = {value=8}

«configuration» BwMode
«Configuration» mode=[Bw]
«nfp» : qos = {value=8}
«nfp» : power = {value=7}

Control specification

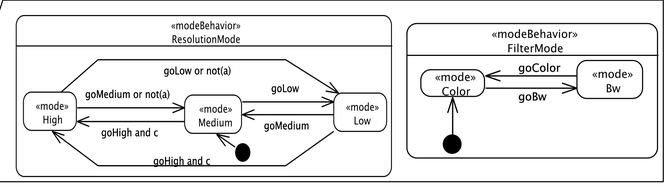


Metamodel extension

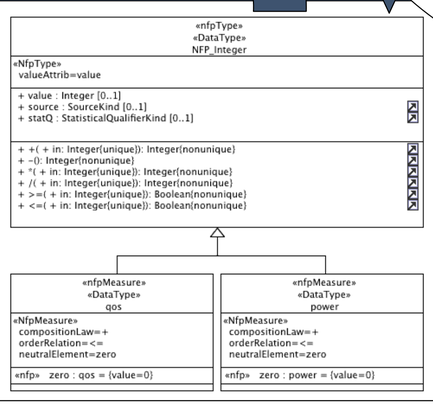


Application task graph

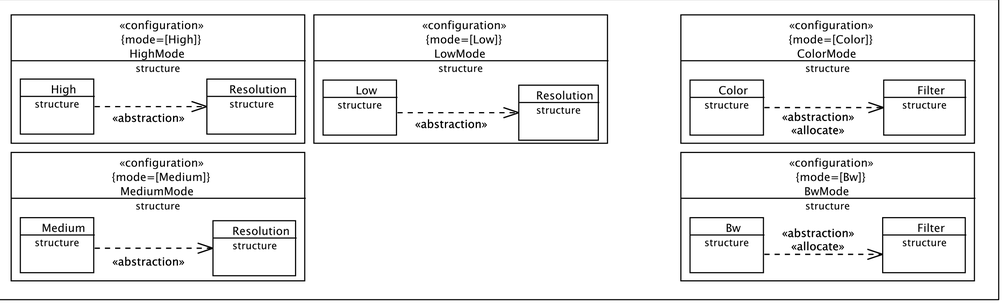
Behavior specification



Non-functional properties



Implementations mappings



Mapping from MARTE to BZR

Controller(okHighColor, okHighBw, okMediumColor, okMediumBw, okLowColor, okLowBw, goHigh, goMedium, goLow, goColor, goBw)
returns(canHighColor, canHighBw, canMediumColor, canMediumBw, canLowColor, canLowBw)

Taking decision inputs and returning accessible configurations

atLeastOne = (((false -> pre(canHighColor)) ∧ okHighColor) ∧
((true -> pre(canMediumColor)) ∧ okMediumColor) ∧
((false -> pre(canLowColor)) ∧ okLowColor) ∧
((false -> pre(canHighBW)) ∧ okHighBW) ∧
((false -> pre(canMediumBW)) ∧ okMediumBW) ∧
((false -> pre(canLowBW)) ∧ okLowBW))
atMostOne = (¬(
(okHighColor ∧
(okMediumColor ∨ okLowColor ∨
okHighBW ∨ okMediumBW ∨ okLowBW)) ∧
(okMediumColor ∧ (okLowColor ∨ okHighBW ∨
okMediumBW ∨ okLowBW)) ∧
(okLowColor ∧ (okHighBW ∨ okMediumBW ∨
okLowBW)) ∧
(okHighBW ∧ (okMediumBW ∨ okLowBW)) ∧
(okMediumBW ∧ okLowBW)))
assume(atLeastOne ∧ atMostOne)

One accessible configuration...
... and only one...
... is supposed to be selected by a decision system

enforce((canHighColor ∨ canMediumColor ∨ canLowColor ∨
canHighBW ∨ canMediumBW ∨ canLowBW) ∧
((power * -1) ≤ -16) ∨ **((qos * -1) ≤ -14)**))

Two control objectives :
- there is always one accessible configuration
- power > 16 or qos > 14

with(a,c)

Two controllables : a, c

canHigh = (medium ∧ (goHigh ∧ c)) ∨ (low ∧ (goHigh ∧ c)) ∨
(high ∧ ¬((goMedium ∨ ¬(a)) ∨ (goLow ∨ ¬(a))))
canMedium = (high ∧ (goMedium ∨ (¬(a))) ∨ (low ∧ (goMedium)) ∨
(medium ∧ ¬((goHigh ∧ c) ∨ (goLow)))
canLow = (medium ∧ (goLow)) ∨ (high ∧ (goLow)) ∨
(low ∧ ¬((goMedium) ∨ (goHigh ∧ c)))
canColor = (bw ∧ (goColor)) ∨ (color ∧ ¬(goBW))
canBW = (color ∧ (goBW)) ∨ (bw ∧ ¬(goColor))

States accessibility definition

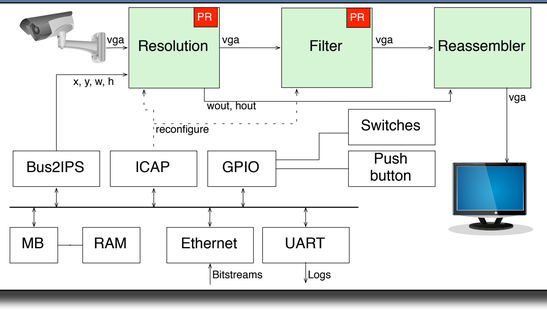
canHighColor = canHigh ∧ canColor
canHighBW = canHigh ∧ canBW
canMediumColor = canMedium ∧ canColor
canMediumBW = canMedium ∧ canBW
canLowColor = canLow ∧ canColor
canLowBW = canLow ∧ canBW

Configurations accessibility definition

(qos,power) =
if(okMediumBw) then (18,27) else (
if(okMediumColor) then (15,28) else (
if(okLowBw) then (19,24) else (
if(okLowColor) then (16,25) else (
if(okHighBw) then (13,15) else (
if(okHighColor) then (10,16) else (
(0,0)))))))

Weights computations

Execution platform



Simulation (two first steps)

Conclusions

- Brings Discrete Controller Synthesis (DCS) to SoC modeling
  - MDE oriented methodology, with MARTE extension for control specification
- Makes DCS more usable in practice
  - Enable dynamic (on-line) decision
  - Decision relies on accessible (ie. *allowed*) configurations

References

- P. Ramadge, W. Wonham. "Supervisory control of a class of discrete event processes". *Analysis and Optimization of Systems*. 1984
- S. Kent. "Model Driven Engineering". *IFM*. 2002
- INRIA Sardes Team. "BZR synchronous language". http://bzr.inria.fr/
- S. Guillet et al. "Designing formal reconfiguration control using UML/ MARTE". *ReCoSoC*. 2012