Dataflow models of MPSoCs with NoCs

ESWEEK 2012 tutorial

Marco Bekooij
marco.bekooij@nxp.com

October 7, 2012
Outline

- Relevance of dataflow modeling
- Basics of dataflow analysis
- Network-on-chip modeling
  - NoC models at different abstraction levels based on the-earlier-the-better refinement relation
- Summary
Relevance dataflow modeling

- Guarantees: checking whether real-time constraints will be satisfied after design and programming, even for cyclic graphs
- Hardware dimensioning: dimension hardware such that real-time constraints will be satisfied after programming
- Optimization:
  - quantify the consequences on the performance of (manual) changes in the software and hardware
  - explore trade-offs
- Simplify manual system design:
  - design rules ensure that the (abstraction of the) system is temporally monotonic, such that lower execution times improve the throughput
Comparison with simulation

Cons:
- No guarantees, instead limited coverage
- Evaluation becomes possible after detailed design
- Trade-offs are implicit
- Analytical system properties are implicit

Pros:
- Does not require the creation of a dataflow model
  - circumvent issues with correctness and accuracy of dataflow models
Message of this talk

- Dataflow models are used to compute the worst-case temporal behavior:
  - such that the sources (ADCs) and sinks (DACs) can run periodically
  - the execution of the tasks is data driven and is not periodic
  - for systems that make use of back-pressure and flow-control
  - for systems that contain a NoC
  - the analysis does not rely on characterization of traffic

- Dataflow models have very attractive analytical properties:
  - temporally monotonic and linear self-timed execution
  - a number of formal semantic models: MaxPlus, set of linear inequalities, labeled transition system
  - the-earlier-the-better-refinement relation allows to make a trade-off between accuracy and compactness of a (NoC) model
Dataflow analysis models

A number of dataflow models exist with a different trade-off between expressivity and analysability

- will explain the basics of dataflow analysis using the least expressive dataflow model (HSDF)
Formal semantic models of timed dataflow

- Labeled transition system (simulate worst-case behavior, model checking)
- MaxPlus Algebra (symbolic evaluation, closed-form expressions)
- System of inequalities (convex optimization in polynomial time)
- Trace algebra (the earlier the better refinement relation)
HSDF model elements

- Be aware that it is a *model* with elements without a physical counterpart.
Firing rule of HSDF actor: 1 token in each input queue
  - no check for space in output queue
Firing duration $\rho$: time between start and finish
  - consumption one token from each input queue at the start of a firing
  - production in each output queue at the finish of a firing
Self-timed execution

▶ An actor is enabled if there are a sufficient number of tokens in its input queues

▶ During self-timed execution actors fire as soon as they are enabled
  • token production time: \( c(i) = \max(a(i), b(i)) + \rho(v_0) \)
  • multiple firings of the same actor can occur simultaneously
Monotonicity

- An earlier production of a token cannot result in a later firing of actors during self-timed execution
  - An earlier production can be a result of:
    - lower firing durations
    - an earlier start of an actor
    - more initial tokens

- In other words, the self-timed execution of a dataflow model is scheduling anomaly free
Implication of monotonicity

- It is sufficient to show that an admissible schedule exists given worst-case firing durations
  - a schedule is *admissible* if the start-times of the actors in the schedule are not before their enabling times
Maximum cycle mean (1/throughput)

Nodes on a simple cycle are passed once

The set $O(G)$ contains all simple cycles in the graph $G(V, E)$

The maximum cycle mean (MCM) of an HSDF graph $G$ is:

$$
\mu(G) = \max_{o \in O(G)} \text{CM}(o), \text{ with}
$$

$$
\text{CM}(o) = \frac{\sum_{v \in V(o)} \rho(v)}{\sum_{e \in E(o)} \delta(e)}
$$
Maximum cycle mean (1/throughput)

\[ \rho(v_0) = 2 \quad \rho(v_1) = 1 \]

\[ \delta = 2 \]

\[ \mu = \max\left( \frac{1}{1}, \frac{2}{1}, \frac{1+2}{2} \right) = 2 \text{ second/token} \]
Maximum cycle mean (1/throughput)

$\rho(v_0) = 2$
$\rho(v_1) = 1$
$\delta = 2$

Corresponding schedule
Constraint corresponding with a dataflow edge with tokens

\[ \sigma(v_j, k) \geq \sigma(v_i, k - 1) + \rho_i \]

\[ \sigma(v_i, k + n) = \sigma(v_i, k) + n \cdot \mu \]

after substitution: \[ \sigma(v_j, k) \geq \sigma(v_i, k) + \rho_i - \mu \]
Rate optimal static periodic schedule (ROSPS)

An edge $e_{ij}$ with $\delta_{ij}$ initial tokens imposes the constraint:

$$\sigma(v_j, k + \delta_{ij}) \geq \sigma(v_i, k) + \rho_i$$

(1)

Given that the start times of an ROSPS are described by $\sigma(v_i, k) = \sigma(v_i, 0) + \mu k$ we can rewrite (1) in:

$$\sigma(v_j, 0) + \mu (k + \delta_{ij}) \geq \sigma(v_i, 0) + \mu k + \rho_i$$

(2)

After simplification and reordering we obtain:

$$\sigma(v_i, 0) - \sigma(v_j, 0) \leq \mu \delta_{ij} - \rho_i = c_{ij}$$

(3)

For all edges in the graph we obtain the following system in linear inequalities:

$$A\sigma \leq c$$

(4)
Schedule computation

- The rate optimal static periodic schedule can be computed with an LP-solver
  - convex problem, polynomial time-complexity.
  - remark: task schedule in the implementation is *not* periodic
Temporal refinement of components [GTW11]

Refinement iff:
\[
(\forall j, \hat{a}'(j) \leq \hat{a}(j)) \Rightarrow (\forall i \geq 0, \hat{b}'(i) \leq \hat{b}(i))
\]

- A sufficient condition for refinement of component $A$ by $A'$ is that if tokens for component $A'$ do not arrive later than for $A$, then $A'$ will not produce tokens later than $A$. 

\[
A' \subseteq A
\]
Refinement of a graph

\[ \forall j, \ X'_j \subseteq X_j \Rightarrow G' \subseteq G \]
\[ G' \subseteq G \Rightarrow G \text{ can be replaced by } G'' \text{ because the worst-case temporal behavior of } G' \text{ is better (not worse) than that of } G \text{ i.e.} \]
\[ (\forall j, \hat{d}'(j) \leq \hat{d}(j)) \Rightarrow (\forall i \geq 0, \hat{e}'(i) \leq \hat{e}(i)) \]
Conformance relation between hardware and the dataflow model

\[ a(i) \xrightarrow{\text{actor A}} b(i) \]
\[ A' \sqsubseteq A \]
\[ a'(i) \xrightarrow{\text{task A'}} b'(i) \]

- Arrival of a token represent an event, e.g.
  - \( a(i) \): arrival time of the i-th token in a queue
  - \( b(i) \): production moment of the i-th token in a queue
  - \( a'(i) \): event indicating the arrival of the i-th full container in a FIFO
  - \( b'(i) \): event indicating the arrival of the i-th empty container in the same FIFO
Transitivity of the refinement relation

- Transitivity of $\sqsubseteq$:
  - $(C''' \sqsubseteq C' \sqsubseteq C) \Rightarrow (C''' \sqsubseteq C)$
  - $(G''' \sqsubseteq G' \sqsubseteq G) \Rightarrow (G''' \sqsubseteq G)$

- Transitivity enables to create dataflow models at different abstraction levels
  - e.g. to make a trade-off between accuracy and compactness of a NoC model
Dataflow graph abstraction

\[ G \]

\[ d(i) \]

\[ \rightarrow \]

\[ v_0 \]

\[ \rightarrow \]

\[ g(i) \]

\[ G' \]

\[ d'(i) \]

\[ \rightarrow \]

\[ v_1 \]

\[ \rightarrow \]

\[ e'(i) \]

\[ \rightarrow \]

\[ v_2 \]

\[ \rightarrow \]

\[ f'(i) \]

\[ \rightarrow \]

\[ v_3 \]

\[ \rightarrow \]

\[ g'(i) \]

\[ T/3 \]

\[ T/3 \]

\[ T/3 \]

\[ T/3 \]

\[ G' \sqsubseteq G \]
Dataflow graph abstraction

\[ g(i) = \max(d(i), g(i - 1)) + T \]
\[ g'(i) = f'(i) + T/3 \]
\[ f'(i) = e'(i) + T/3 \]
\[ e'(i) = \max(d'(i), f'(i - 1)) + T/3 \]
\[ \text{after substitution: } g'(i) = \max(d'(i), g'(i - 1) - T/3) + T \]
Dataflow graph abstraction

- $G' \subseteq G$ because:
  - given $(\forall i, d'(i) \leq d(i) \land g'(-1) = g(-1)) \Rightarrow (\forall i \geq 0, g'(i) \leq g(i))$
  - the reason is that given $(\forall i, d'(i) \leq d(i) \land g'(-1) = g(-1))$ we can prove with induction that:
    $$\max(d'(i), g'(i-1) - T/3) + T = g'(i) \leq g(i) = \max(d(i), g(i-1)) + T$$
Conservative HSDF dataflow model of task scheduled by a starvation-free scheduler

The dataflow model is conservative iff:
\[(\forall j,\ a(j) \leq \hat{a}(j)) \Rightarrow (\forall i \geq 0, \ b(i) \leq \hat{b}(i))\]

- Most servers belong to the class of starvation-free schedulers
Firing duration given time-division multiplex

\[ \rho(i) \leq x(i) + (P - B) \left\lceil \frac{x(i)}{B} \right\rceil \]

- \( x(i) \): execution time of the \( i \)-th execution, \( P \): period, \( B \): slice length
Latency-rate dataflow model [WBS07, WBS09]

\[ \Theta = P - B \]
\[ \rho(i) = \frac{P}{B} x(i) \]

▶ Improved accuracy: one task is modeled with 2 actors!

- \[ \hat{f}(i) = \max(\hat{e}(i) + P - B, \hat{f}(i - 1)) + \frac{P}{B} x(i) \]
- worst-case for any trace if: \[ \hat{\rho} = \max_i \rho(i) = \frac{P}{B} \hat{x}(i) \]
- \[ 1/\hat{\rho} \] is called the rate
Alternative hard real-time analysis techniques (1)

- Classical Real-Time calculus [CKT03], SymTA/S [SE09], Network calculus [Cru91], Holistic analysis [TC94]
  - not restricted to starvation-free schedulers
  - make use of a worst-case and best-case traffic characterization in the time interval domain
    - infinite representation $\hat{\alpha}(\Delta t)$ or finite representation $(P, J, d)$
  - support of cyclic data-dependencies is problematic: correlation between event streams is lost as a result of time-interval traffic characterization
  - exponential worst-case computational complexity
    - iterative fixed point computation
    - no closed form expression for throughput like the MCM expression
Alternative hard real-time analysis techniques (2)

- Latest incarnation of Real-Time calculus [TS09]
  - suitable for arbitrary cyclic HSDF graphs
  - not restricted to starvation-free schedulers
  - makes use of a worst-case and best-case traffic characterization in the time domain
    - correlation between event streams is maintained
  - exponential worst-case computational complexity
    - iterative fixed point computation
    - no closed form expression for throughput like the MCM expression
    - requires conversion of an SDF into a potentially huge HSDF
Alternative hard real-time analysis techniques (3)

- Compositional temporal analysis (CTA) model [HGWB12]
  - suitable for arbitrary cyclic SDF graphs
  - analysis algorithm has a polynomial computational complexity
    - convex problem
  - makes use of periodic event traces in the time domain to characterize traffic
  - unlike dataflow it has no firing rules, nor tokens
  - can be used as an over-approximation of dataflow
  - like dataflow analysis, over-approximation is based on the earlier the better refinement theory
  - is restricted to starvation-free schedulers
  - associative composition operator: enables incremental design
NoC modeling [MBvM04, HWM+09]

- Will gradually introduce a dataflow model for an (Aethereal) guaranteed throughput network connection
- Connection:
  - one channel for request-messages from master to slave
  - another channel for response-messages from slave to master
Channel model

Channel has a FIFO like behavior with a limited bandwidth and a latency:

- first-in-first-out arrival order of data
- lossless
- bandwidth reservation with starvation free schedulers
- read and writes are blocking

The dataflow model for one channel will be introduced step-wise
FIFO communication between processors

Hardware architecture (one task per processor):

Corresponding HSDF:
Minimum buffer capacity

\[ \mu = \max \left( \frac{\rho_p}{1}, \frac{\rho_c}{1}, \frac{\rho_p + \rho_c}{n} \right) \] (5)

- Conclusion:
  - more than 2 tokens does not result in a lower \( \mu \) (higher throughput)
  - thus a FIFO capacity of more than two words does not result in a higher throughput
Communication through router between processors

Hardware architecture:

![Hardware architecture diagram]

Corresponding HSDF:

![Corresponding HSDF diagram]

- Router can be seen as a processor that copies data
- Effects of sharing can be included in $\rho_r$
TDM arbitration in router

Budgets of 2 slots (max 2 tokens) every $T$ time units

$T = 5$ slots

2 slots

$T = 5$

Corresponding HSDF

$\rho_r = T$

- Very conservative model with respect to the throughput: 1 token every $T$
  - a 1 or a 2 tokens consumption/production per firing cannot be represented in an HSDF graph
  - $\rho_r = \frac{2}{5}T$ does not correspond with the maximum latency
Latency-rate model for the router

\[ \rho_{r0} = (1 - \frac{2}{5})T \quad \rho_{r1} = \frac{2}{5}T \]

- Latency \((T)\) can be larger than the rate \((\frac{2}{5}T)\)
- Latency does not include the time that tokens stay in a queue
Router modeled as a latency-rate server

- More than 2 tokens can be needed for the maximum throughput
  \[ \frac{\rho_p + \rho_{r0} + \rho_{r1}}{n_1} \geq \rho_{r1}, \text{i.e.} \quad \frac{\frac{2}{5}T + (1 - \frac{2}{5})T + \frac{2}{5}T}{n_1} \leq \frac{2}{5}T \Rightarrow n_1 \geq 3 \]
- result valid for any \( \rho_p \)
Communication through a network connection

Hardware architecture:

![Diagram of network communication architecture](image)

Corresponding HSDF:

![Diagram of HSDF](image)
Communication through a network connection

Hardware architecture:

- Processor 1
  - FIFO
  - stall
- FIFO
  - network data
  - NI
  - data credits
  - one additional token space available
- FIFO
- Processor 2
  - stall

Corresponding HSDF:

- Independently adjustable latency and rate for data and credits
Abstraction

Hardware architecture:

Processor 1
stall
FIFO
network
data
FIFO
 NI
 credits
one additional token
space available
Processor 2
stall

More abstract HSDF model:
Summary

- Dataflow models are used to derive the worst-case temporal behavior of applications executed on an MPSoC with a NoC
  - NoCs must use starvation free schedulers
  - flow-control does not complicate dataflow analysis
  - monotonicity makes it sufficient to show that a schedule exists that satisfies the temporal constraints
  - such a schedule can be computed in polynomial time
    - HSDF exact
    - (SDF, CSDF, VRDF, and VPDF after over-approximation)

- The-earlier-the-better refinement relation enables creation of dataflow models at different levels of abstraction
Questions?

- Consequence of applying backpressure 😊
References I


References II


References III

Modular performance analysis of cyclic dataflow graphs.  

Modelling run-time arbitration by latency-rate servers in dataflow graphs.  

Monotonicity and run-time scheduling.  