

A Domain-specific Approach for Software Development on Manycore Platforms

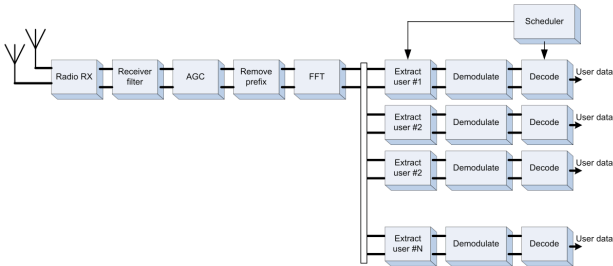
Jerker Bengtsson and Bertil Svensson
{Jerker.Bengtsson, Bertil.Svensson}@hh.se

MCC-08 Ronneby, November 27-28, 2008



CENTRE FOR RESEARCH ON EMBEDDED SYSTEMS

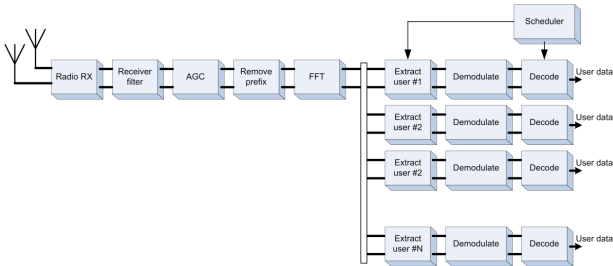
What Domain?



Ref. H. Sahlin, "Introduction and overview of LTE Baseband Algorithms", Baseband research group, Ericsson AB

- High-performance signal processing (e.g. Radio Basestations or Radar Systems)
 - time constrained processing of infinite streams of data
 - large amounts of heterogeneous parallelism
- **Problem:** How to describe and best map **time constrained** computation graphs on **abstract** manycore targets?

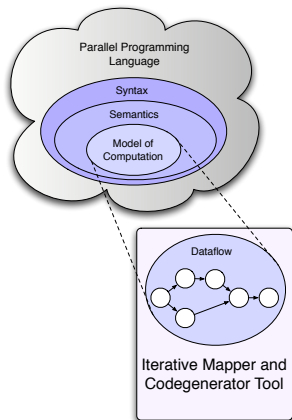
What Domain?



Ref. H. Sahlin, "Introduction and overview of LTE Baseband Algorithms", Baseband research group, Ericsson AB

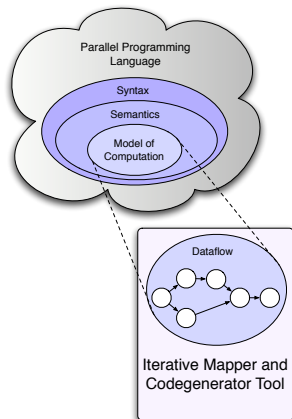
- High-performance signal processing (e.g. Radio Basestations or Radar Systems)
 - time constrained processing of infinite streams of data
 - large amounts of heterogeneous parallelism
- **Problem:** How to describe and best map **time constrained** computation graphs on **abstract** manycore targets?

Research Focus



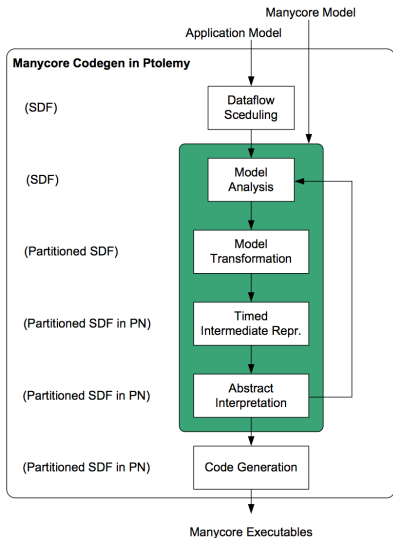
- We argue that *domain-specific* parallel models of computation (MoC) are needed to
 - be able to develop efficient parallelization and mapping tools
 - enable portability of tools and design methodologies
 - raise the parallel programming abstraction level
- Dataflow MoC's are *naturally parallel* and provides an excellent match to
 - signal processing applications
 - manycore targets

Research Focus



- We argue that *domain-specific* parallel models of computation (MoC) are needed to
 - be able to develop efficient parallelization and mapping tools
 - enable portability of tools and design methodologies
 - raise the parallel programming abstraction level
- Dataflow MoC's are *naturally parallel* and provides an excellent match to
 - signal processing applications
 - manycore targets

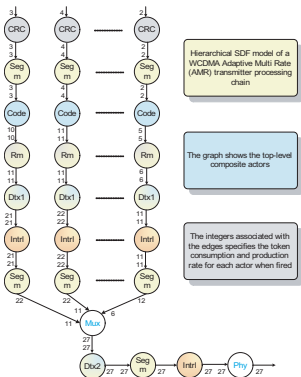
A Tool for Iterative Mapping and Codegeneration CERES



- We are building an iterative mapping and code generation tool
- The idea is to iteratively tune parallel mapping by
 - user-specified constraints in the application model
 - feed back dynamic performance measurements
- We use the Ptolemy modeling environment (UC Berkeley)
- ptolemy.berkeley.edu/

The Application Model

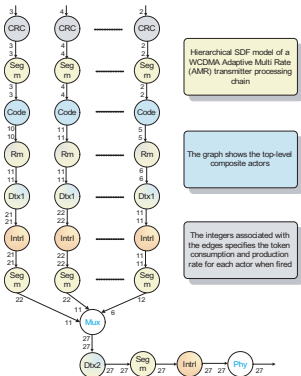
CERES



- We describe an application using Synchronous Dataflow (SDF)
- With each actor we associate a tuple $\langle r_p, r_m, R_r, R_s \rangle$ where
 - r_p is the worst case computation time, in number of operations.
 - r_m is the requirement on data allocation, in words.
 - $R_s = \{r_{s_1}, r_{s_2}, \dots, r_{s_n}\}$ and r_{s_i} is the number of words produced on channel i each firing.
 - $R_r = \{r_{r_1}, r_{r_2}, \dots, r_{r_m}\}$ and r_{r_j} is the number of words consumed on channel j each firing.

The Application Model

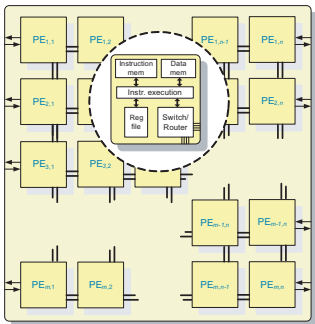
CERES



- We describe an application using Synchronous Dataflow (SDF)
- With each actor we associate a tuple $\langle r_p, r_m, R_r, R_s \rangle$ where
 - r_p is the worst case computation time, in number of operations.
 - r_m is the requirement on data allocation, in words.
 - $R_s = \{r_{s_1}, r_{s_2}, \dots, r_{s_n}\}$ and r_{s_i} is the number of words produced on channel i each firing.
 - $R_r = \{r_{r_1}, r_{r_2}, \dots, r_{r_m}\}$ and r_{r_j} is the number of words consumed on channel j each firing.

What manycore targets?

CERES



- We are interested in manycores with
 - core individual instruction sequencing
 - tightly coupled interconnectivity
 - distributed on-chip memory (core private)
 - software controlled caching
- We have derived a parallel machine model for such targets

Machine Model

- A machine is described by two tuples M and F
- The computational resources are described by

$$M = \langle (x, y), p, b_g, g_w, g_r, o, s_o, s_l, n_b, c, h_l, r_l, r_o \rangle$$

which are parameters

- The computational performance is described by

$$F(M) = \langle t_p, t_s, t_r, t_c, t_{gw}, t_{gr} \rangle$$

which are functions of M determining the cost for

- process the fire code of an actor (t_p)
- core send and receive (t_s, t_r)
- core to core propagation time (t_c)
- reading and writing to global memory (t_{gw}, t_{gr})

Machine Model

- A machine is described by two tuples M and F
- The computational resources are described by

$$M = \langle (x, y), p, b_g, g_w, g_r, o, s_o, s_l, n_b, c, h_l, r_l, r_o \rangle$$

which are parameters

- The computational performance is described by

$$F(M) = \langle t_p, t_s, t_r, t_c, t_{gw}, t_{gr} \rangle$$

which are functions of M determining the cost for

- process the fire code of an actor (t_p)
- core send and receive (t_s, t_r)
- core to core propagation time (t_c)
- reading and writing to global memory (t_{gw}, t_{gr})

Performance Functions

- To model a machine we first set parameters of M
- Then we define how to evaluate the functions in F
 - $t_p(r_p, p) = \left\lceil \frac{r_p}{p} \right\rceil$
 - $t_s(R_s, o, s_o) = \left\lceil \frac{R_s}{\text{framesize}} \right\rceil \times o + R_s \times s_o$
 - $t_r(R_r, o, r_o) = \left\lceil \frac{R_r}{\text{framesize}} \right\rceil \times o + R_r \times r_o$
 - $t_c(R_s, d, s_l, c, h_l, r_l) = s_l + d \times h_l + \left\lceil (R_s - 1) \times \frac{1}{c} \right\rceil + r_l$
 - $t_{g_w}(R_s, d, s_l, c, h_l, b_g, g_w) = \dots$
 - $t_{g_r}(R_s, d, s_l, c, h_l, b_g, g_r, r_l) = \dots$
- With this approach we can tune F in detail for a certain target to obtain higher accuracy

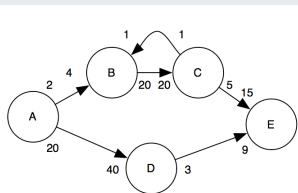
Performance Functions

- To model a machine we first set parameters of M
- Then we define how to evaluate the functions in F
 - $t_p(r_p, p) = \left\lceil \frac{r_p}{p} \right\rceil$
 - $t_s(R_s, o, s_o) = \left\lceil \frac{R_s}{framesize} \right\rceil \times o + R_s \times s_o$
 - $t_r(R_r, o, r_o) = \left\lceil \frac{R_r}{framesize} \right\rceil \times o + R_r \times r_o$
 - $t_c(R_s, d, s_l, c, h_l, r_l) = s_l + d \times h_l + \left\lceil (R_s - 1) \times \frac{1}{c} \right\rceil + r_l$
 - $t_{g_w}(R_s, d, s_l, c, h_l, b_g, g_w) = \dots$
 - $t_{g_r}(R_s, d, s_l, c, h_l, b_g, g_r, r_l) = \dots$
- With this approach we can tune F in detail for a certain target to obtain higher accuracy

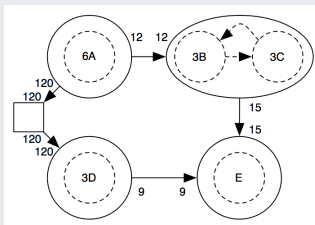
Performance Functions

- To model a machine we first set parameters of M
- Then we define how to evaluate the functions in F
 - $t_p(r_p, p) = \left\lceil \frac{r_p}{p} \right\rceil$
 - $t_s(R_s, o, s_o) = \left\lceil \frac{R_s}{\text{framesize}} \right\rceil \times o + R_s \times s_o$
 - $t_r(R_r, o, r_o) = \left\lceil \frac{R_r}{\text{framesize}} \right\rceil \times o + R_r \times r_o$
 - $t_c(R_s, d, s_l, c, h_l, r_l) = s_l + d \times h_l + \left[(R_s - 1) \times \frac{1}{c} \right] + r_l$
 - $t_{g_w}(R_s, d, s_l, c, h_l, b_g, g_w) = \dots$
 - $t_{g_r}(R_s, d, s_l, c, h_l, b_g, g_r, r_l) = \dots$
- With this approach we can tune F in detail for a certain target to obtain higher accuracy

Executable IR: Timed Configuration Graph



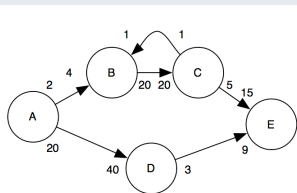
The application (SDF)



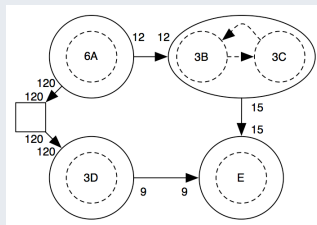
The IR (SDF in PN)

- The Configuration Graph is a dataflow process network (PN)
- The PN model is annotated with the functions of F
 - edges are weighted with one of t_c, t_{gw}, t_{gr}
 - vertices has a list of operations t_p, t_s, t_r
- The usage of the IR is two-fold, we can:
 - use it to generate code for cores and the network
 - do abstract interpretation to evaluate performance

Executable IR: Timed Configuration Graph



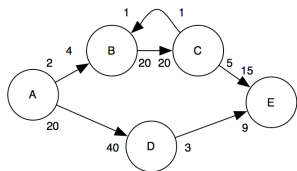
The application (SDF)



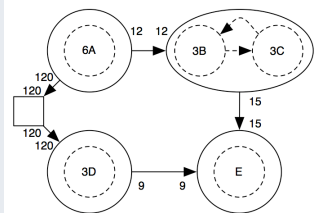
The IR (SDF in PN)

- The Configuration Graph is a dataflow process network (PN)
- The PN model is annotated with the functions of F
 - edges are weighted with one of t_c, t_{gw}, t_{gr}
 - vertices has a list of operations t_p, t_s, t_r
- The usage of the IR is two-fold, we can:
 - use it to generate code for cores and the network
 - do abstract interpretation to evaluate performance

Executable IR: Timed Configuration Graph



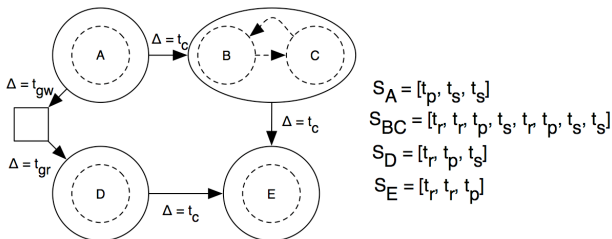
The application (SDF)



The IR (SDF in PN)

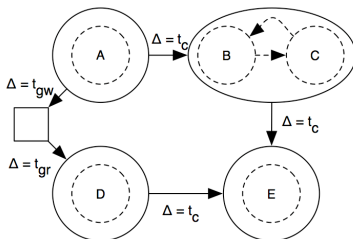
- The Configuration Graph is a dataflow process network (PN)
- The PN model is annotated with the functions of F
 - edges are weighted with one of t_c, t_{gw}, t_{gr}
 - vertices has a list of operations t_p, t_s, t_r
- The usage of the IR is two-fold, we can:
 - use it to generate code for cores and the network
 - do abstract interpretation to evaluate performance

Contributions in the paper



- We have
 - proposed a fine-grained a machine model
 - developed a multipurpose intermediate representation
 - shown how to do analysis on the IR using abstract interpretation
- What we describe in the paper is implemented in Ptolemy

Contributions in the paper



$$S_A = [t_p, t_s, t_s]$$

$$S_{BC} = [t_r, t_r, t_p, t_s, t_r, t_p, t_s, t_s]$$

$$S_D = [t_r, t_p, t_s]$$

$$S_E = [t_r, t_r, t_p]$$

- We have
 - proposed a fine-grained a machine model
 - developed a multipurpose intermediate representation
 - shown how to do analysis on the IR using abstract interpretation
- What we describe in the paper is implemented in Ptolemy

Summary and future work

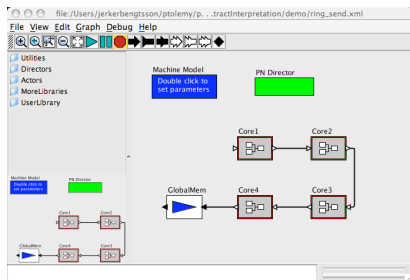
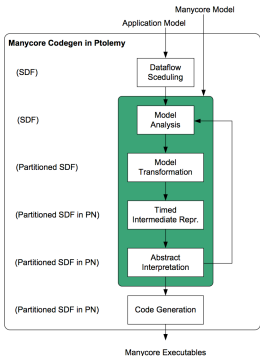
- We aim for a tool chain offering flexible mapping strategies
 - The goal is to evaluate and use non functional properties for iterative tuning of parallel mappings
-
- To explore how efficiently this can be done, we further need to
 - model and compare with concrete target (e.g. RAW)
 - calibrate the model so that IR calculations closely match dynamic measurements on target hardware
 - To investigate practical usefulness, we plan to
 - model an application (e.g. LTE uplink) and study the consequences of parallel mapping choices

Summary and future work

- We aim for a tool chain offering flexible mapping strategies
 - The goal is to evaluate and use non functional properties for iterative tuning of parallel mappings
-
- To explore how efficiently this can be done, we further need to
 - model and compare with concrete target (e.g. RAW)
 - calibrate the model so that IR calculations closely match dynamic measurements on target hardware
 - To investigate practical usefulness, we plan to
 - model an application (e.g. LTE uplink) and study the consequences of parallel mapping choices

The End

CERES



- Thank you!
- e-mail: Jerker.Bengtsson@hh.se