Methodologies and Tools for Development of Signal Processing Software on Manycore Platforms.

Jerker Bengtsson
Jerker.Bengtsson@hh.se

EPC meeting at Halmstad, October 1, 2008

Centre for Research on Embedded Systems

EPC Thread 1

CERES

- The "multicore problem" is a software development problem
- We argue that *domain-specific* parallel programming models and tools are the solution to
  - raise the programming abstraction level
  - be able to develop efficient parallelization tools
  - enable portability of tools and design methodologies
- Our proposal: **a tool chain for iteratively tuned code generation** based on
  - a parallel model of computation - to describe the application
  - a parallel machine abstraction - to describe and model a manycore
  - a parallel IR - allowing performance analysis through abstract interpretation

EPC Thread 1

CERES

- The "multicore problem" is a software development problem
- We argue that *domain-specific* parallel programming models and tools are the solution to
  - raise the programming abstraction level
  - be able to develop efficient parallelization tools
  - enable portability of tools and design methodologies
- Our proposal: **a tool chain for iteratively tuned code generation** based on
  - a parallel model of computation - to describe the application
  - a parallel machine abstraction - to describe and model a manycore
  - a parallel IR - allowing performance analysis through abstract interpretation

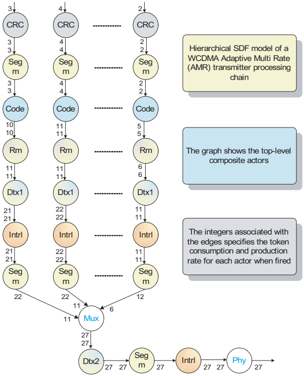
EPC Thread 1

CERES

- The "multicore problem" is a software development problem
- We argue that *domain-specific* parallel programming models and tools are the solution to
  - raise the programming abstraction level
  - be able to develop efficient parallelization tools
  - enable portability of tools and design methodologies
- Our proposal: **a tool chain for iteratively tuned code generation** based on
  - a parallel model of computation - to describe the application
  - a parallel machine abstraction - to describe and model a manycore
  - a parallel IR - allowing performance analysis through abstract interpretation

The Application Model

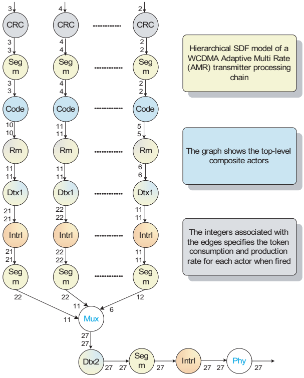
CERES



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

The Application Model

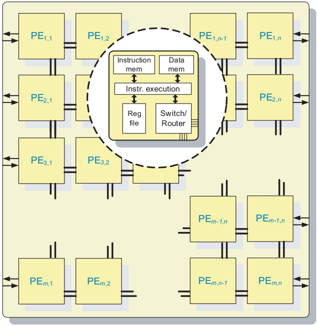
CERES



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

What manycore targets?

CERES



- We are interested in manycores where the cores
  - are many to the number
  - have individual instruction execution ability
  - have local private memory
  - are tightly coupled to the interconnection network (mesh)
  - allow the programmer to orchestrate the transactions between local and global memories
- We have derived a parallel machine abstraction for such targets

Machine Abstraction

CERES

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

$$M = < (x, y), p, b_g, g_w, g_r, o, s_o, s_I, n_b, c, h_I, r_I, r_o >$$

which are parameters

- The computational performance is described by

$$F = < t_p(M), t_s(M), t_r(M), t_c(M), t_{gw}(M), t_{gr}(M) >$$

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})

HÖGSKOLAN HALMSTAD

Machine Abstraction

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

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

which are parameters

- The computational performance is described by

$$F =< t_p(M), t_s(M), t_r(M), t_c(M), t_{gw}(M), t_{gr}(M) >$$

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_{qr})

Machine Parameters

CERES

- (x, y) is the number of rows and columns of cores.
- p is the processing power of each core
- b_g is global memory bandwidth
- g_w is the global memory write latency
- g_r is the penalty global memory read latency
- l_g is the global memory access latency
- o is software overhead for initiation of a network transfer
- s_o is core send occupancy
- s_l is the latency for a sent message to reach the network
- n_b is the network input- and output buffer capacity
- c is the bandwidth of each interconnection link
- h_l is network hop latency
- r_l is the latency from network to receiving core
- r_o is core receive occupancy when receiving a message

Performance Functions

CERES

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

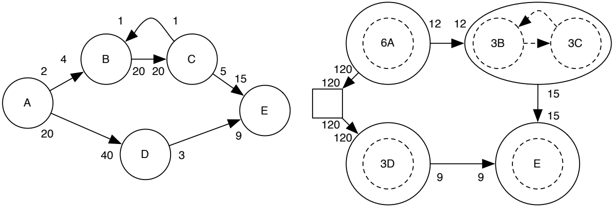
Performance Functions

CERES

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

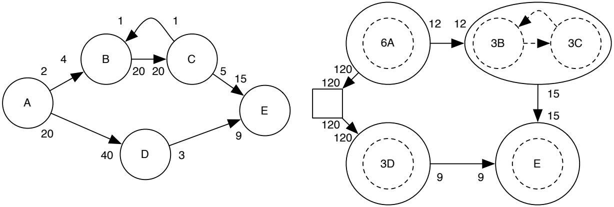
The IR: Timed Configuration Graphs

CERES



- 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 perfromance

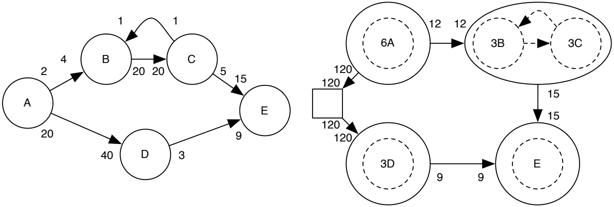
The IR: Timed Configuration Graphs CERES



- 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 perfromance

The IR: Timed Configuration Graphs

CERES



- 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 perfromance

Performance Feedback for Iterative Tuning CERES

- We aim for a tool chain offering flexible mapping strategies
- The goal is to evaluate non functional properties for iterative tuning of parallel mappings

- To show that this can be done, we further need to
  - model a concrete target (e.g. RAW)
  - calibrate the model so that IR calculations match target implementation
- To investigate practical usefulness, we plan to
  - model an application (e.g. LTE uplink) and study the consequences of parallel— mapping choices
- Timeplan for this is end of December 2008

Performance Feedback for Iterative Tuning

CERES

- We aim for a tool chain offering flexible mapping strategies
- The goal is to evaluate non functional properties for iterative tuning of parallel mappings

- To show that this can be done, we further need to
  - model a concrete target (e.g. RAW)
  - calibrate the model so that IR calculations match target implementation
- To investigate practical usefulness, we plan to
  - model an application (e.g. LTE uplink) and study the consequences of parallel— mapping choices
- Timeplan for this is end of December 2008

Work reports since last meeting CERES

- Current reports related to this presentation
  - [1] "**A Set of Models for Manycore Performance Evaluation Through Abstract Interpretation of Configuration Graphs**", Technical report IDE0856
  - [2] "**Methodologies and Tools for Development of Signal Processing Software on Multicores**", Workshop on Streaming Systems, Lake Como, Italy, Nov 8-9, 2008
  - [3] "**A Domain-specific Approach for Software Development on Multicore Platforms**", submitted to the MCC workshop in Ronneby