Multi-Objective Intrinsic Evolution of Embedded Systems (MOVES)

Paul Kaufmann, Marco Platzner

Computer Engineering Group

University of Paderborn

{paulk, platzner}@uni-paderborn.de

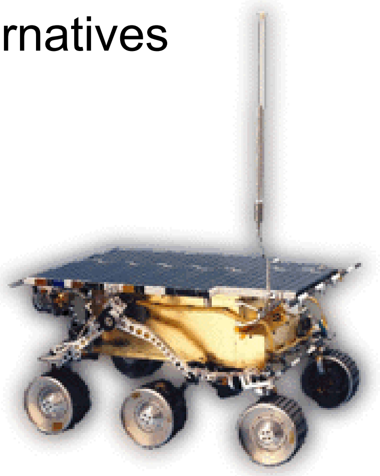
Outline

- motivation/vision/approach
- work done
  - hardware representation model
  - evaluation framework
  - evolving circuits with multi-objective optimizers
  - results so far
- ongoing work
  - improving scalability
  - implementing resource-aware EA on Virtex-4
  - cooperation in evolvable hardware

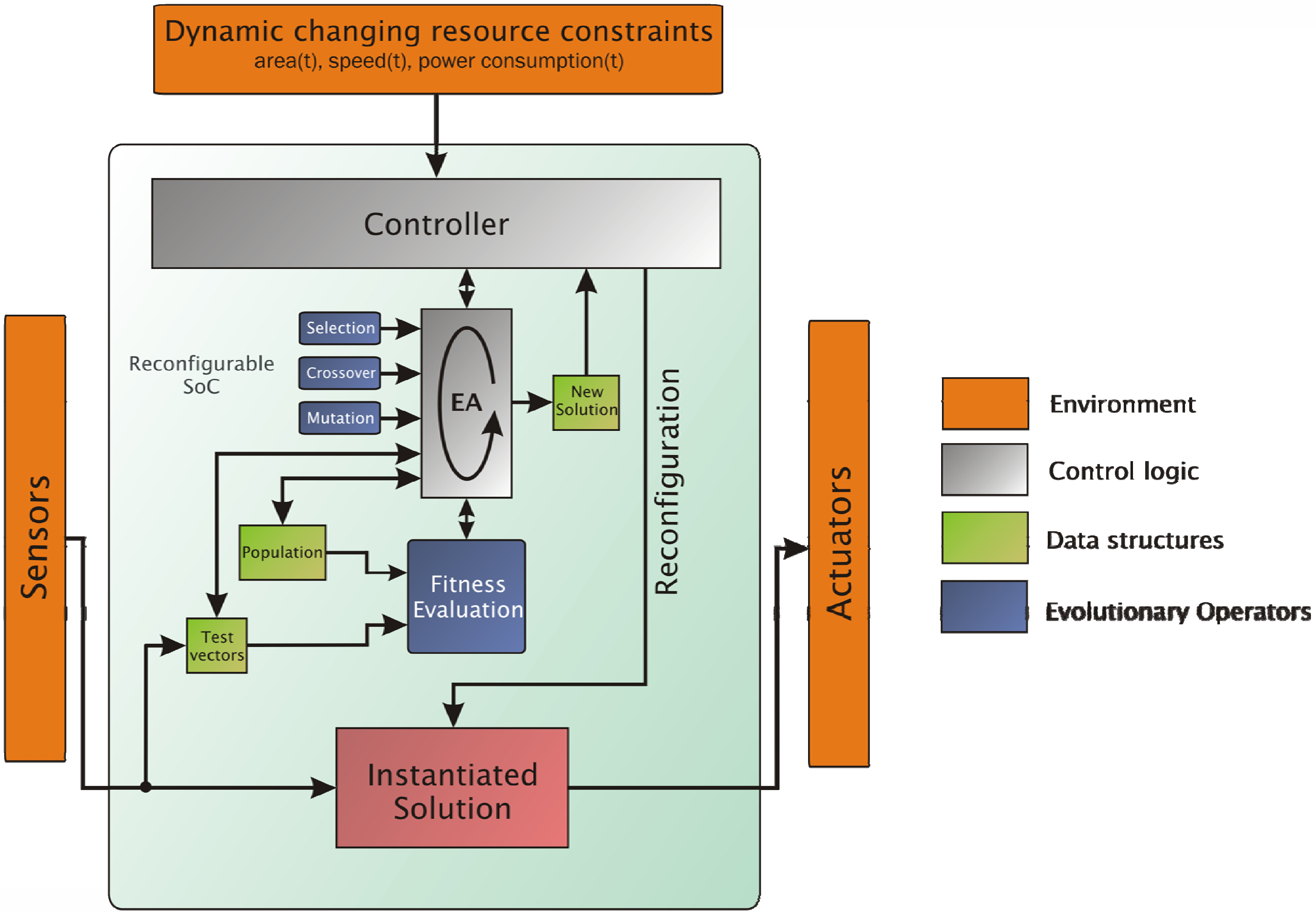
Motivation / Vision

- investigate intrinsic evolution as a mechanism to achieve self-adaptation and –optimization for autonomous embedded systems

- an embedded system ...
  - adapts to slow changes by simulated evolution
    - typically, change of environment
  - adapts to radical changes by switching to pre-evolved alternatives
    - typically, change in computational resources
  - requires intrinsic evolution for autonomous operation



Architecture



Dynamic changing resource constraints
area(t), speed(t), power consumption(t)

Controller

Reconfigurable SoC

Selection

Crossover

Mutation

EA

New Solution

Population

Test vectors

Fitness Evaluation

Reconfiguration

Instantiated Solution

Sensors

Actuators

- 🟧 Environment
- ⬜ Control logic
- 🟩 Data structures
- 🟦 Evolutionary Operators

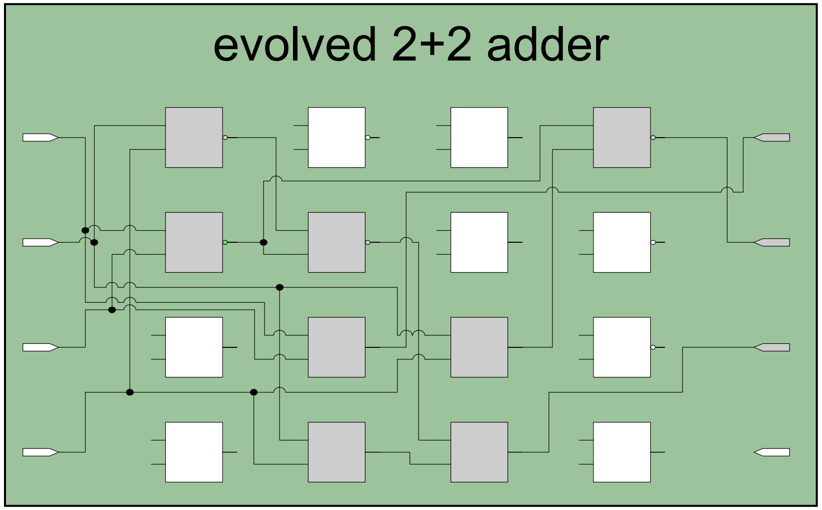
Outline

- ✓ motivation/vision/approach
- work done
  - hardware representation model
  - evaluation framework
  - evolving circuits with multi-objective optimizers
  - results so far
- ongoing work
  - improving scalability
  - implementing resource-aware EA on Virtex-4
  - cooperation in evolvable hardware

Hardware Representation Model

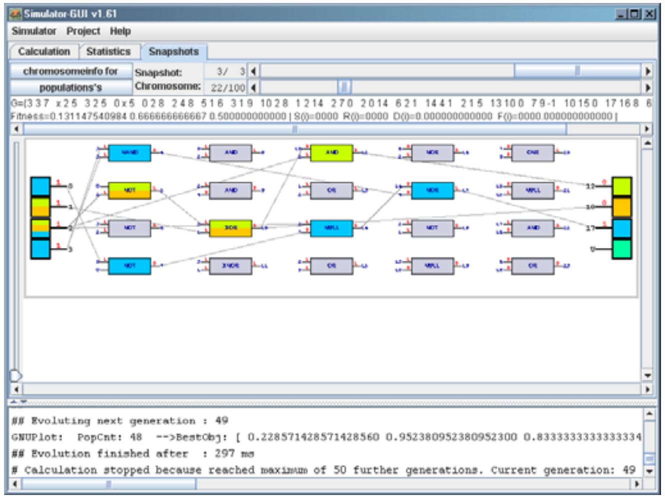
- adapted Cartesian Genetic Program (CGP) model
  - array of combinational blocks connected by feed-forward wires
  - chromosome defines configuration of the array
  - model similar to [Miller & Thomson, '96]
  - advantage: matches well FPGA architectures
  - challenge: scalability

- adapted embedded CGP (ECGP) model [Miller & Walker, ’04]
  - automatically identify larger building blocks
  - reuse of building blocks
  - implemented, experiments to be done



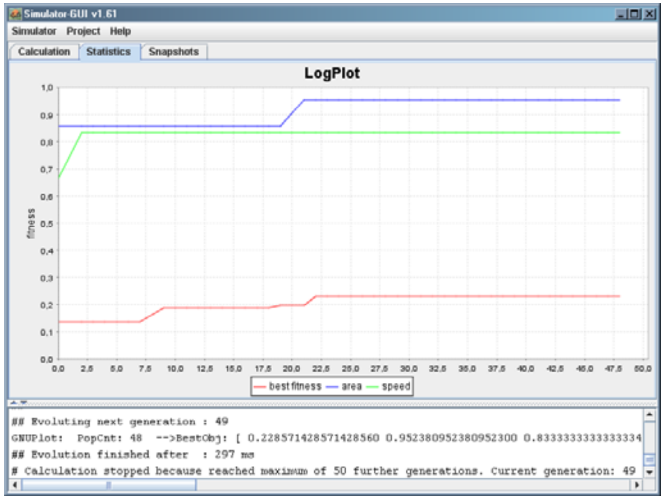
Evaluation Framework

- implemented toolbox for experimenting
  - interactive and batch modes
  - stop/resume evolutionary process
  - graphical array representation and analysis tools, e.g. fitness plots, statistics, ...
- GCP model parametrizable
  - array dimensions
  - block complexity and functions
  - interconnect
- modular and extensible to different EAs and representation models
  - currently: variants of GA, SPEA2, TSPEA2 on CGP and ECGP models



```
G=[3 3 7 x 2 5 3 2 5 0 x 5 0 2 8 2 4 8 5 1 6 3 1 9 10 2 8 1 2 14 2 7 0 20 14 6 2 1 14 4 1 2 1 5 13 10 0 7 9 -1 10 15 0 17 16 8 6
Fitness=0.131147540984 0.666666666667 0.500000000000 | S()=0000 R()=0000 D()=0.000000000000 F()=0000.000000000000 |
```

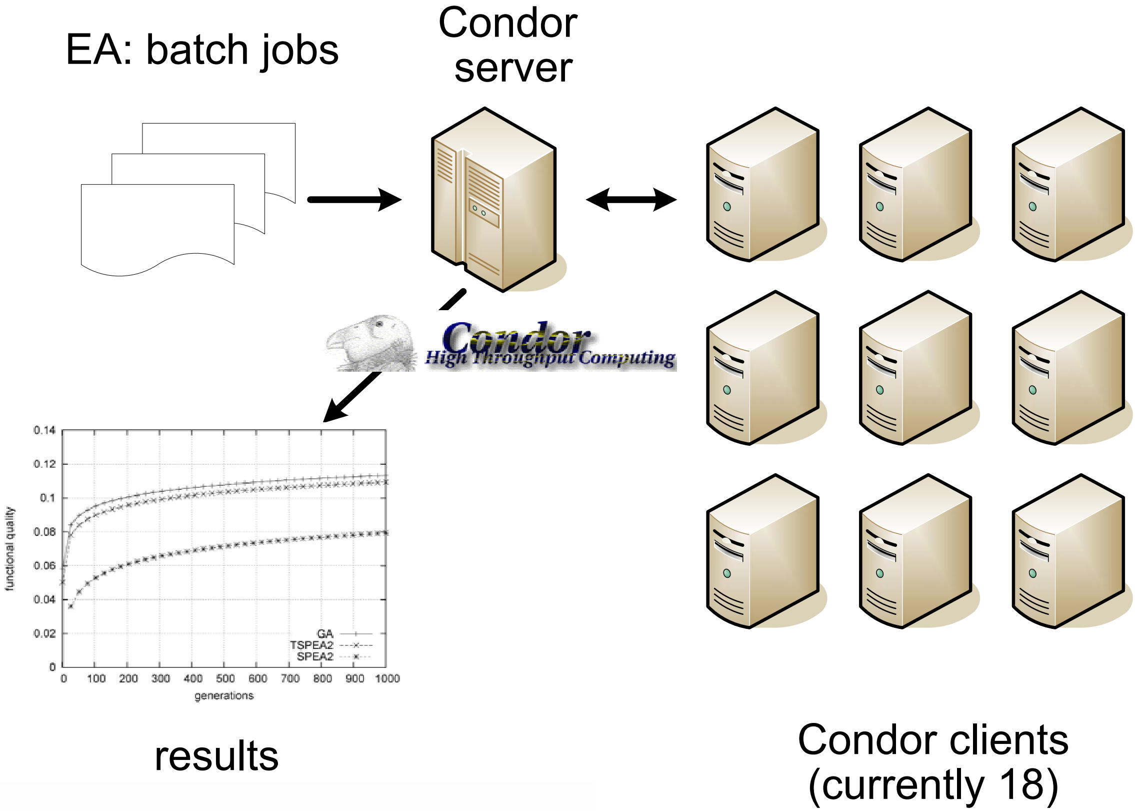
```
## Evoluting next generation : 49
GNUPlot:  PopCnt: 48  -->BestObj: [ 0.228571428571428560 0.952380952380952300 0.8333333333333334
## Evolution finished after  : 297 ms
# Calculation stopped because reached maximum of 50 further generations. Current generation: 49
```



```
## Evoluting next generation : 49
GNUPlot:  PopCnt: 48  -->BestObj: [ 0.228571428571428560 0.952380952380952300 0.8333333333333334
## Evolution finished after  : 297 ms
# Calculation stopped because reached maximum of 50 further generations. Current generation: 49
```

Evaluation Framework (2)

- distributed computing environment to speed up experiments
  - installed Condor grid
  - implemented extensions for improved client control and Java support



EA: batch jobs

Condor server

results

Condor clients (currently 18)

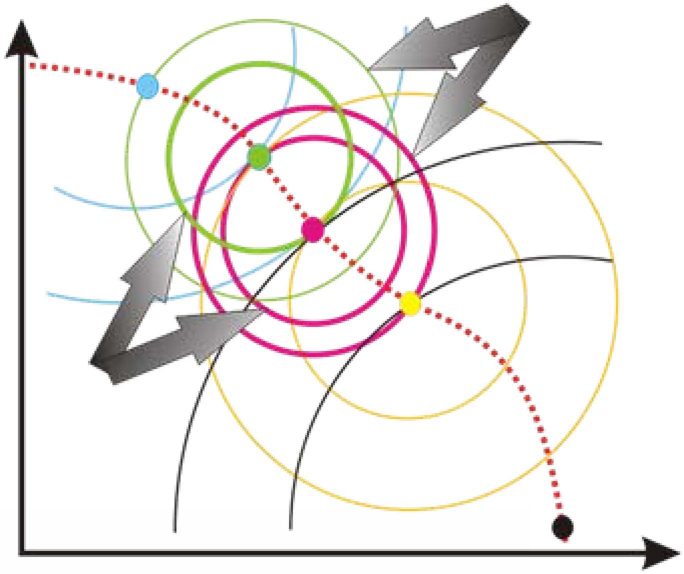
EAs for Multi-objective Optimization

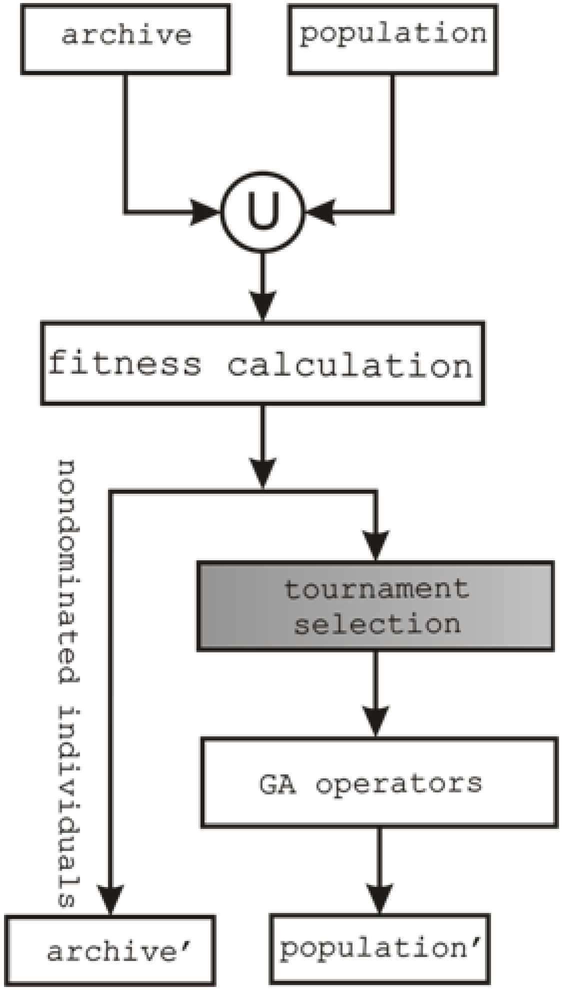
- implemented and experimented with different variants of EAs
  - multiple objectives: functional quality, circuit area, circuit speed
  - experimented with rather simple test functions: adders, multipliers, parity-function, hash functions, etc.

- evolutionary algorithms
  - GA (reference algorithm)
    - conventional, single-objective genetic algorithm
    - uses elitism, tournament selection, uniform crossover, mutation
  - SPEA2
  - TSPEA2

TSPEA2

- Turtle Strength Pareto Evolutionary Algorithm
  - based on SPEA2 [Zitzler et al., '01], which maintains diversity by dissolving Pareto front clusters with k-th nearest neighbor density estimation
  - prioritize a main objective to speed up convergence
    - selection scheme similar to MO-Turtle GA [Trefzer et al., ‘05]

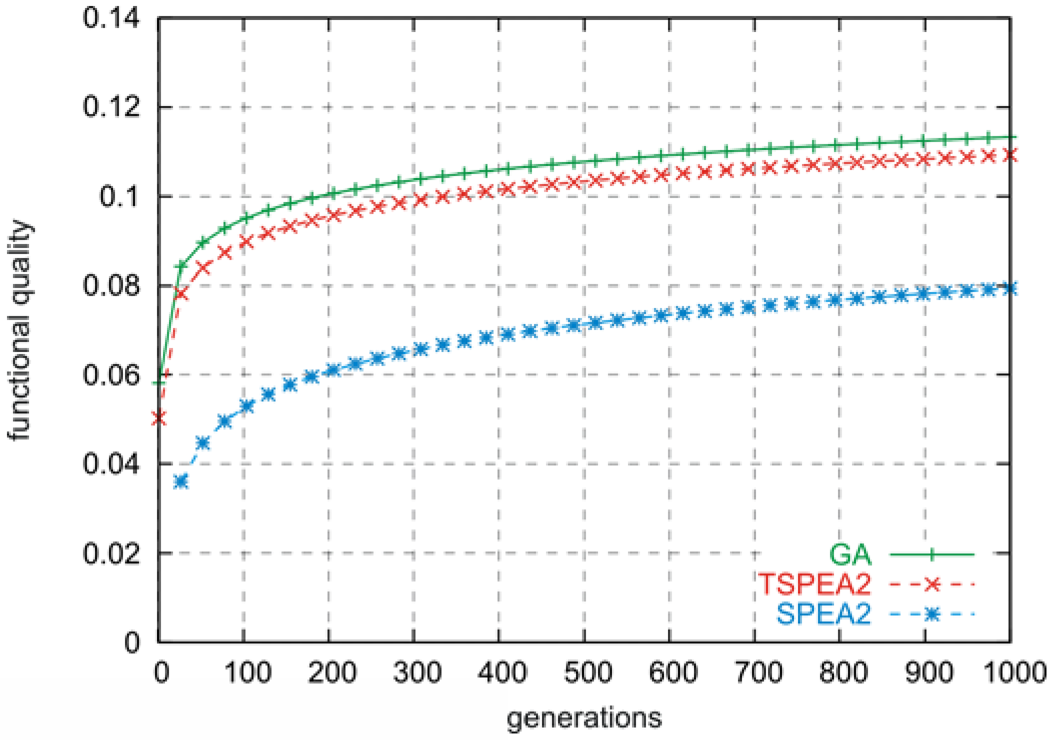




Example: Hashing Function

- test function used by [Tettamanzi et al.,'98]
  - find a function that maps 4096 16-bit keys to 256 indices
  - no correctness property, just a functional "quality"
  - CGP model: 8x8 4-LUT array, levels back parameter l=8

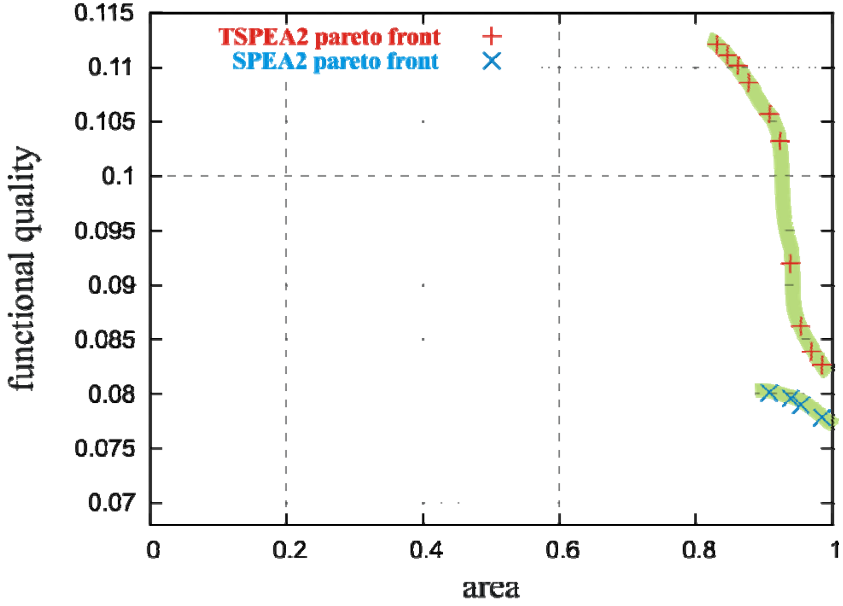
average functional quality



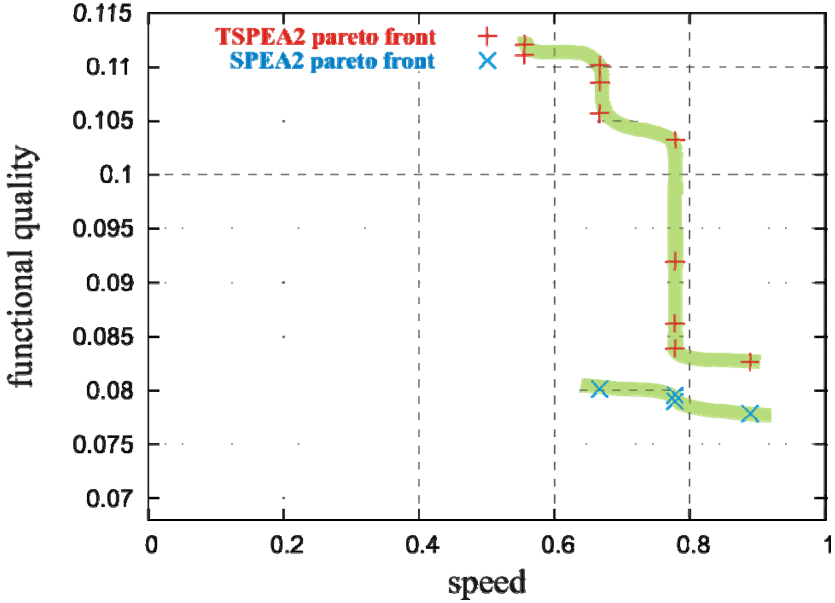
Example: Hashing Function (2)

- Pareto front approximations: SPEA2 vs. TSPEA2

quality vs. area



quality vs. speed



- surprisingly, TSPEA2 outperforms SPEA2 in quality and area and speed

Results so far ...

- for functions with a correctness property (eg. arithmetic functions)
  - functional quality is a constraint (100% required) and needs to be preferred during selection to achieve acceptable convergence times
  - we would expect that this comes at the price of a degraded Pareto front → SPEA2 and TSPEA2 generate roughly the same Pareto front

- for functions without correctness property (eg. hashing function)
  - functional quality is an objective, but TSPEA2 beats SPEA2 in both, runtime and Pareto front: possible explanations
    - our objectives are not necessarily conflicting (!)
    - we have experimented with rather small examples
    - there are not too many possible area/speed values

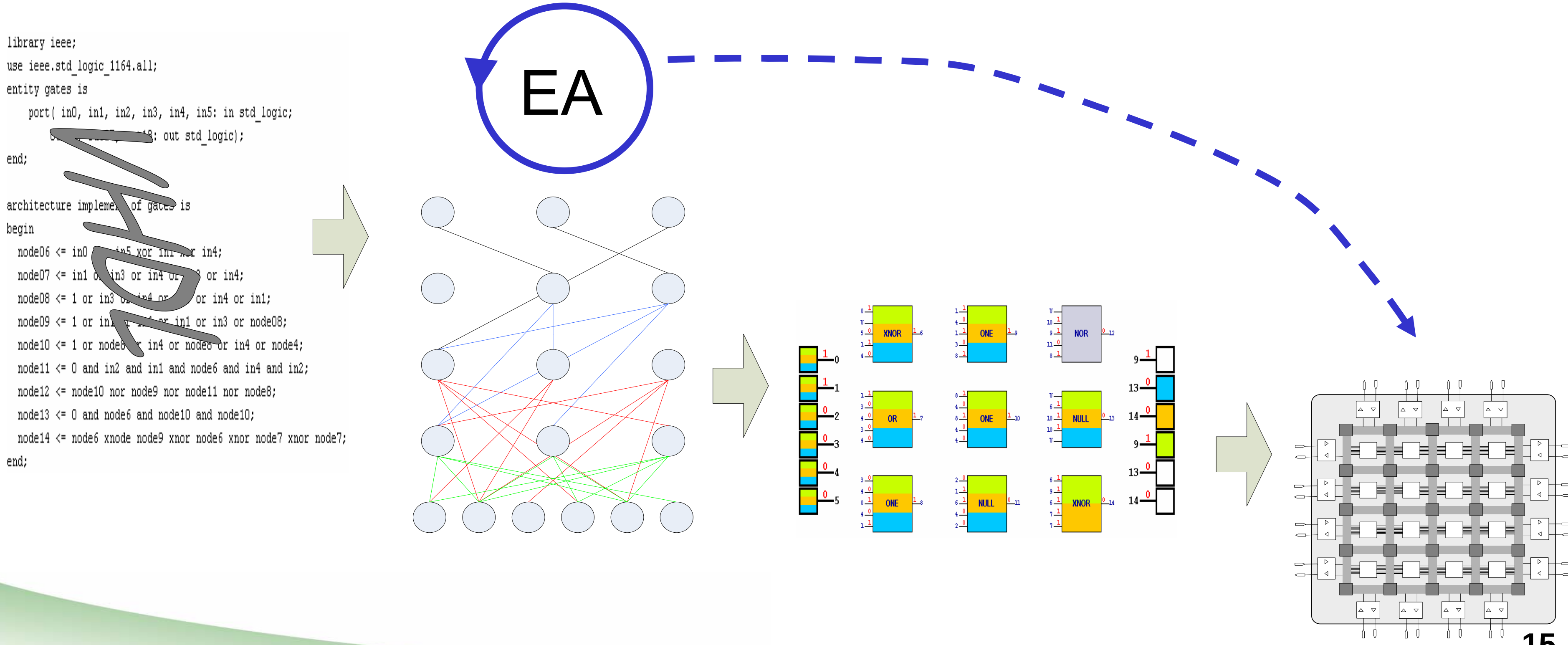
- to evolve more complex circuits, we have to ...
  - move to higher-level hardware representations, automatically identify larger building blocks, utilize domain-specific knowledge, ...

Outline

- ✓ motivation/vision/approach
- ✓ work done
  - hardware representation model
  - evaluation framework
  - evolving circuits with multi-objective optimizers
  - results so far
- ongoing work
  - improving scalability
  - implementing resource-aware EA on Virtex-4
  - cooperation in evolvable hardware

Scalability

- native CGP model too flat for complex problems
- evolving higher-level models is more efficient but increases the effort to map to actual hardware
  - balance the computational resources between evolutionary search and FPGA mapping



```
library ieee;
use ieee.std_logic_1164.all;
entity gates is
    port( in0, in1, in2, in3, in4, in5: in std_logic;
        [illegible]: out std_logic);
end;

architecture implement of gates is
begin
  node06 <= in0 [illegible] in5 xor in1 xor in4;
  node07 <= in1 or in3 or in4 or [illegible] or in4;
  node08 <= 1 or in3 or in4 or [illegible] or in4 or in1;
  node09 <= 1 or in1 or in4 or in1 or in3 or node08;
  node10 <= 1 or node8 or in4 or node8 or in4 or node4;
  node11 <= 0 and in2 and in1 and node6 and in4 and in2;
  node12 <= node10 nor node9 nor node11 nor node8;
  node13 <= 0 and node6 and node10 and node10;
  node14 <= node6 xnode node9 xnor node6 xnor node7 xnor node7;
end;
```

Resource-constrained EA

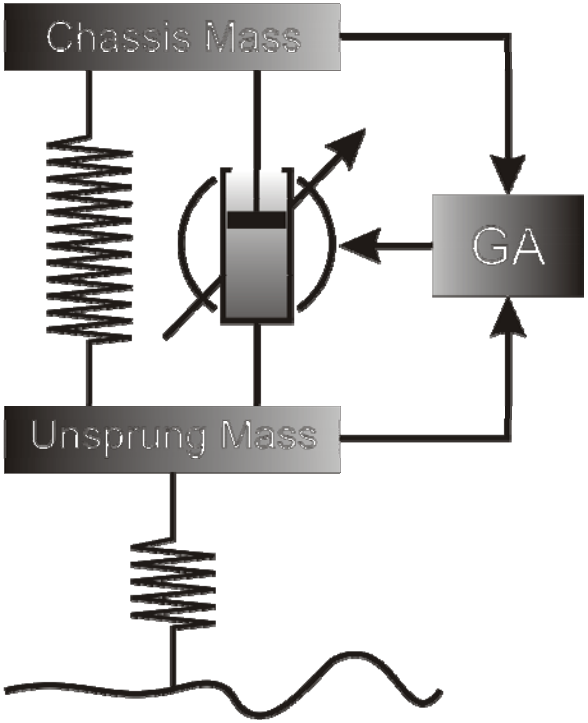
- implement EA on an embedded platform
  - Virtex4 platform FPGA with Power PC core (ML403 board)
  - EA and data structures in software
  - operational circuit and time-consuming computations in hardware
    - fitness evaluation amounts to 75% of EA runtime, density estimation to 12.5%
- challenges
  - what can we achieve under stringent resource-constraints?
    - eg. with population size of only 10, single-parent crossover, ...
  - mapping to FPGA at runtime
    - utilize dynamic reconfiguration vs. virtual FPGA on FPGA

Cooperation in Evolvable Hardware

- with Prof. Jim Tørresen, University of Oslo

- improving scalability of hardware evolution
  - automatic localization of “building blocks”
  - adding scalability-driven objectives

- investigate potential applications
  - prosthetic hand controller
  - active suspension control



Thank you for your attention!