Evolvable Hardware: Tutorial

Lukáš Sekanina

Faculty of Information Technology
Brno University of Technology
Brno, Czech Republic
http://www.fit.vutbr.cz/~sekanina
sekanina@fit.vutbr.cz

GECCO 2007
London, July 7, 2007

Outline

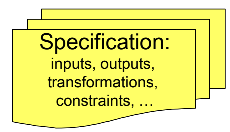
- Principles of evolvable hardware
- Applications of evolvable hardware
- Properties of evolved circuits when considered as computing devices
- Discussion and conclusions

2

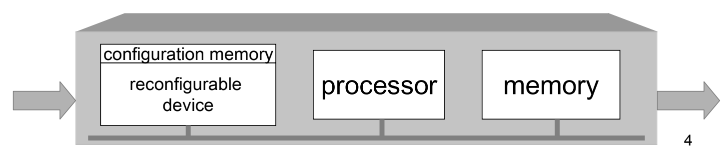
Part I.

Principles of evolvable hardware

3



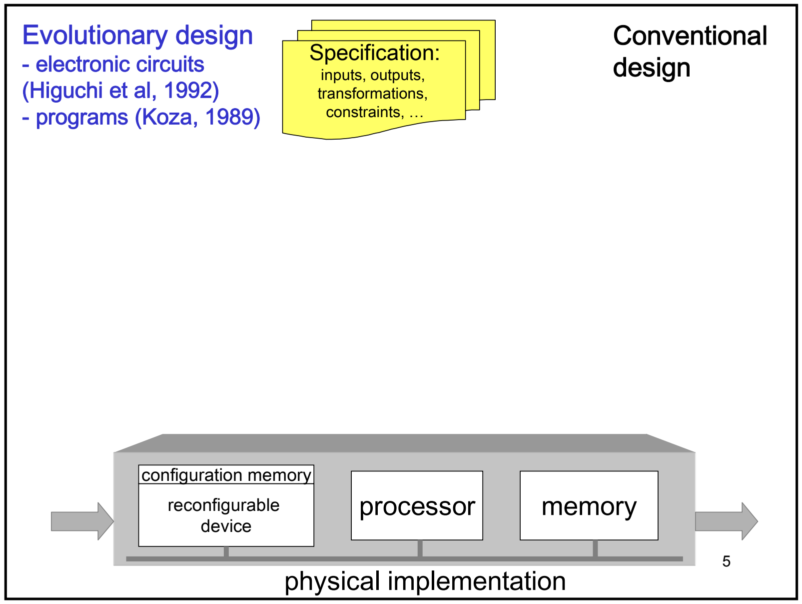
Evolutionary vs conventional design: Example: A computing device

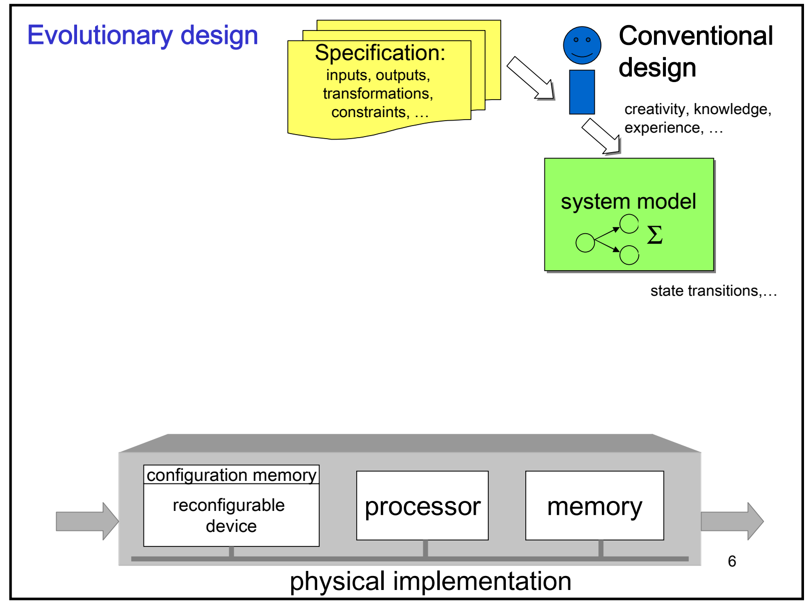


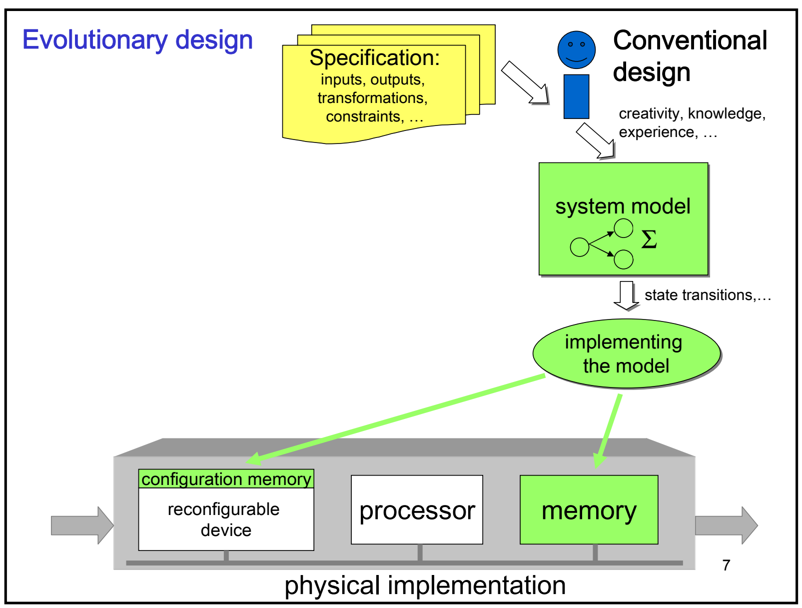
physical implementation

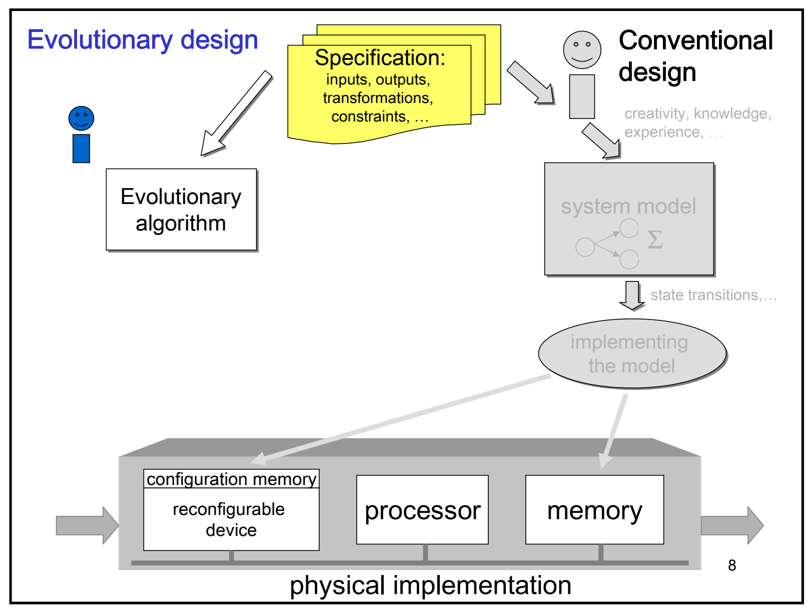
4

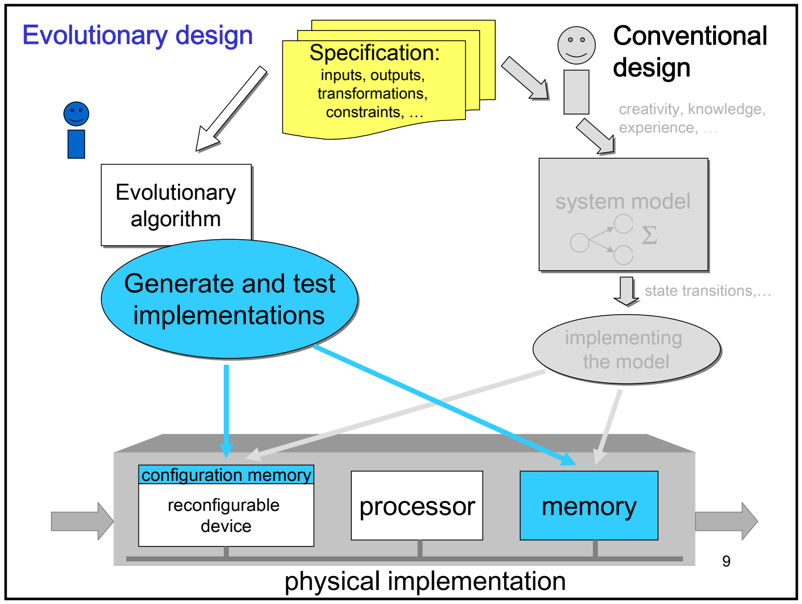
GECCO'07, July 7–11, 2007, London, England, United Kingdom.
ACM 978-1-59593-698-1/07/0007.


Evolutionary design
- electronic circuits
(Higuchi et al, 1992)
- programs (Koza, 1989)
Specification:
inputs, outputs, transformations, constraints, …
Conventional design
configuration memory
reconfigurable device
processor
memory
5
physical implementation


Evolutionary design
Specification:
inputs, outputs, transformations, constraints, …
Conventional design
creativity, knowledge, experience, …
system model
Σ
state transitions,…
configuration memory
reconfigurable device
processor
memory
6
physical implementation


Evolutionary design
Specification:
inputs, outputs, transformations, constraints, …
Conventional design
creativity, knowledge, experience, …
system model
Σ
state transitions,…
implementing the model
configuration memory
reconfigurable device
processor
memory
7
physical implementation


Evolutionary design
Specification:
inputs, outputs, transformations, constraints, …
Evolutionary algorithm
Conventional design
creativity, knowledge, experience, …
system model
Σ
state transitions,…
implementing the model
configuration memory
reconfigurable device
processor
memory
8
physical implementation



Advantages of evolutionary design (1)

- Radically new designs (unreachable by conventional techniques) can be discovered by means of EA.

10

Advantages of evolutionary design (2)

- "The challenge of conventional design is replaced with that of designing an evolutionary process that automatically performs the design in our place. *This may be harder than doing the design directly*, but makes autonomy possible." (A. Stoica)

11

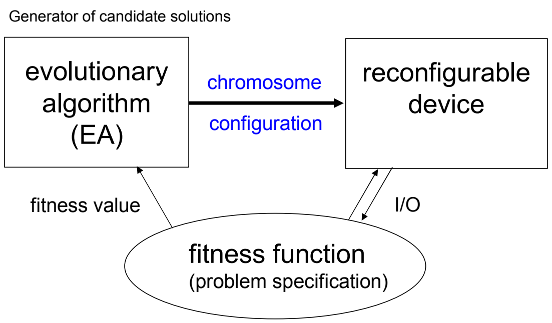
Two views

- Practical viewpoint: How to obtain the required behavior by means of EA?
  - Assumptions, current limitations, achievements …?
- Theoretical viewpoint: Is there any impact of the evolutionary circuit design on theoretical computer science and AI?
  - What is the relationship between formal computational structures and evolved physical systems?

12

Evolvable Hardware =

Evolutionary Algorithm + Reconfigurable Device



- generate test vectors
- obtain responses
- compare the responses with the required values
- calculate the fitness value

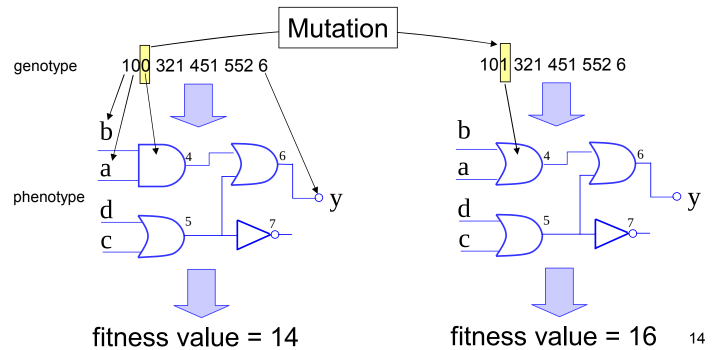
13

Creating new circuits

Example: the goal is to obtain *y(a, b, c d) = a OR b OR c OR d*
Max. fitness is 16 when calculated as the number of correct output bits for all possible input combinations.

Encoding of inputs: 0 – a, 1 – b, 2 – c, 3 – d

Encoding of functions: 0 – AND, 1 – OR, 2 – NOT



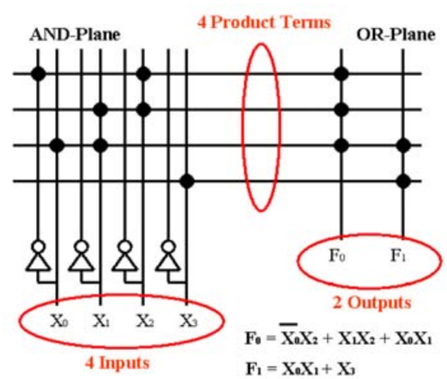
14

Reconfigurable devices

- The functionality of hardware is defined by a configuration bitstream.
- Examples:
  - Field programmable gate array (FPGA)
  - Field programmable transistor array (FPTA)
  - Field programmable analog array (FPAA)
  - Reconfigurable antenna array
  - Reconfigurable optics (e.g. deformable mirrors)
  - Reconfigurable molecular array (e.g. NanoCell)
  - Reconfigurable liquid crystals
  - MEMS
  - Reconfigurable multiprocessors (e.g. PicoChip)

15

Programmable Logic Array



$F_0 = \overline{X_0}X_2 + X_1X_2 + X_0X_1$

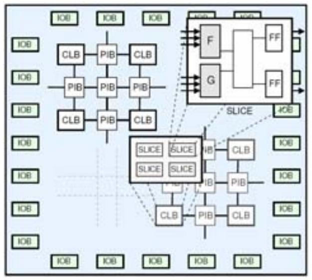
$F_1 = X_0X_1 + X_3$

First evolvable hardware experiments were performed with PLA by Higuchi in 1992. A 6-input multiplexer was evolved.

16

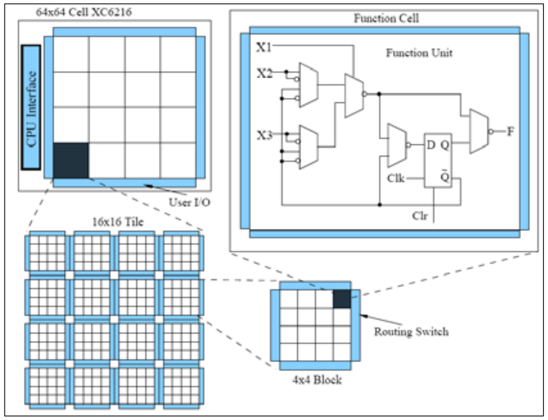
Field Programmable Gate Arrays

- FPGA consists of
  - an array of configurable logic blocks
  - a configurable interconnecting system
  - a configurable I/O ports
- Modern FPGAs contain integrated hard cores
  - SRAMs, multipliers, processors etc.
- Reconfiguration
  - Parallel, serial, standard boundary scan mode, etc.
  - Full – all FPGA resources must be reconfigured
  - Dynamic partial reconfiguration – a part of FPGA can be reconfigured while the remaining circuits work unchanged – important for evolvable hardware
  - External – an external circuit reconfigures FPGA
  - Internal reconfiguration – a circuit inside FPGA can change the configuration of FPGA (supported by Xilinx and Atmel FPGAs)
- The format of the configuration bitstream is typically unknown for users – which is problematic for evolvable hardware



17

FPGA Xilinx XC6216



32 configuration bits per Function Unit (configuration time ~ 40 ns)

This is the first FPGA used for evolvable hardware (A. Thompson, 1995). Not available nowadays. Partial reconfiguration supported. The format of configuration bitstream known to the public.

Xilinx Virtex 2 Pro

www.xilinx.com

- The FPGA currently used.
- Virtex 4 and 5 are also available; however no experience within the EHW community
- an array of CLBs
- embedded 18b multipliers, 16kb SRAMs
- rocket IO ports (max. 3.125 Gb/s)
- 2 PowerPC processors (400 MHz)
- partial reconfiguration
- ICAP – internal configuration access port

19

Reconfiguration of Virtex FPGAs

- A frame is the smallest reconfigurable unit - it refers to a one bit wide column of the configuration memory.
- 48 frames have to be reconfigured to modify one CLB column.
- Configuration modes: serial, parallel and standard boundary scan mode.
- A circuit inside the FPGA (such as MicroBlaze processor core) can change the configuration of the FPGA through ICAP
- The format of the configuration bitstream is not completely documented.
- How to perform runtime partial reconfiguration:
  - modules (with well-defined interfaces) are exchanged between configurations
  - it is possible to change some bits of selected parts of the configuration bitstream (e.g. using JBits – Java API)
  - difficult to perform evolution directly at the bitstream level

20

Atmel AT94K FPSLIC

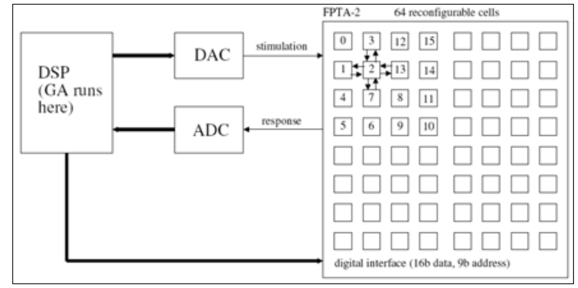
http://www.atmel.com/products/FPSLIC/

- AT94K FPSLIC
  - AVR microcontroller
  - AT90K FPGA
- a dynamic partial reconfiguration supported!
- it is possible to obtain the format of the configuration bitstream
- AVR can configure FPGA
- a much smaller device than Virtex!

21

Field Programmable Transistor Array: FPTA-2

Stoica et al., http://ehw.jpl.nasa.gov/



One cell consists of 14 transistors connected through 44 switches (yellow circles in the figure). The cell is able to implement different building blocks for analog processing, such as two- and three-stage operational amplifiers, logarithmic photo detectors etc. Each cell is configured using five 16bit words. The FPTA-2 uses 96 inputs (a cell has got 1-2 external inputs) and 64 outputs (one for each cell).

Speedup: several orders of magnitude against PSPICE simulations on a Pentium processor (for a circuit with approximately 100 transistors).

Applications:
- Evolution of analog, digital and mixed circuits.
- Extreme environment electronics

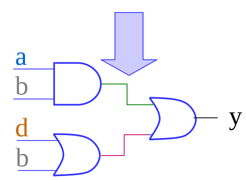
22

Evolutionary algorithm

- a population-based search algorithm
- variants: GA, GP, EP, ES, …
- new populations are created using biologically inspired operators (crossover, mutation, …)
- chromosome (genotype) – a string representing a candidate solution
  - directly
  - indirectly (e.g. a program which describes how to create the solution)
- phenotype – a candidate solution
- fitness function
  - evaluates candidate solutions
  - reflects the problem specification

```
set time t = 0
create initial population P(t)
evaluate P(t)
WHILE (not termination condition) DO
  t = t + 1
  P(t) = create new population using P(t-1)
  evaluate P(t)
END WHILE
```

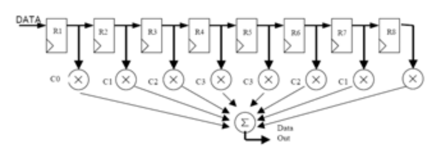
000001011001100101110



23

Domain knowledge

- determines
  - the complexity of a problem for EA
  - the level of innovation that we can obtain
- Example: Digital filter design
  - Much knowledge: An eight tap FIR filter structure is assumed. EA is used to find 8 coefficients of FIR filter. A chromosome contains values of C0-C7. Chromosome size: 8x16 = 128 bits. No innovative solutions.
  - Little knowledge: A set of building blocks is specified. EA is used to put them together to perform the filtering task. A chromosome contains a complete description of the filter. Chromosome size is approx. 1000 bits. Novel solutions can be discovered. However, the problem is hard.

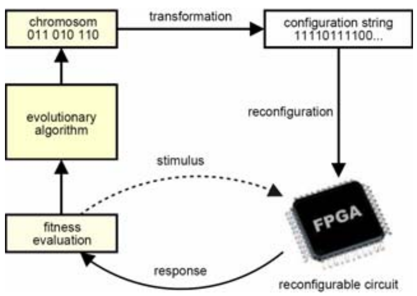


24

Fitness calculation: The main bottleneck of evolvable hardware

$t_{eval} = t_{bitgen} + t_{rec} + N.(t_{vect\_eval} + t_{cmp}) + t_{store}$

t_{eval} – evaluation time of a candidate circuit
t_{bitgen} – time to generate a configuration bitstream from a chromosome
t_{rec} – reconfiguration time
N – the number of test vectors
t_{vect_eval} – evaluation time of a single test vector
t_{cmp} – comparison time of the result with a target value (error update etc.)
t_{store} – time to store the fitness value



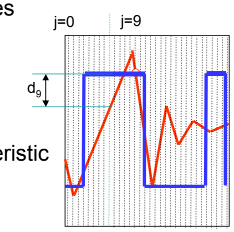
Time of evolution:
$t = G\,(P{*}t_{eval} + t_{np})$

P is the population size, G is the number of generations, t_{np} is time to create a new population.

25

Typical fitness calculations

- Small digital circuits
  - Fitness = Hamming distance (T_c, T_r)
    - T_c is the truth table of a candidate circuit
    - T_r is the required truth table
  - other requirements can be specified: minimize delay, area etc. (⇒ multicriteria optimization)
- Large digital circuits
  - use a training set, estimate the fitness values
- Analog circuits
  - Minimize $d = \Sigma|M_j - R_j|$ (j = 1..K samples)
    - M is measured voltage
    - R is required voltage
  - Many other alternatives: frequency characteristic analysis, power consumption analysis etc.



Where EA is implemented

- PC/cluster
  - the most common case
- DSP (digital signal processor)
  - example: JPL's SABLES
- FPGA
  - as a special circuit
  - as a program for a processor (e.g. MicroBlaze) created using the reconfigurable logic of FPGA
  - as a program for an internal processor available in the FPGA (PowerPC, AVR)
- ASIC (application specific integrated circuit)
  - e.g. some commercial evolvable hardware chips from AITS, Japan

- If EA and reconfigurable device are implemented on the same FPGA (or ASIC) then a very fast internal configuration interface can be established.

27

Typical scenarios

- Evolutionary circuit design
  - EA is used in the design phase only (EA is not a part of the target system).
  - We are interested in the result of evolution.
  - Novel and innovative solutions are sought.
- Evolvable systems
  - EA is typically a part of the target system.
  - EA is used during the operational time to ensure
    - adaptation and/or
    - functional recovery

    when the specification is redefined, HW is corrupted or the environment is changed.
  - Examples: adaptive image compression, adaptive prosthetic hand controller, …

28

Extrinsic vs Intrinsic Evolution

- Extrinsic evolution
  - candidate solutions are evaluated using a SW simulator (PSpice, digital circuit simulators, quantum circuit simulators, …)
- Intrinsic evolution (a crucial feature of evolvable HW)
  - candidate solutions are evaluated in a physical hardware
  - evolution can utilize various resources (features of a given piece of materio, temperature, external signals, …) to build the solution
    - initially demonstrated by A. Thompson in 1995
    - mixtrinsic evolution introduced to eliminate these effects (Stoica et al, 2000)
    - exploited by Harding and Miller (to evolve digital circuits in liquid crystals) and others

29

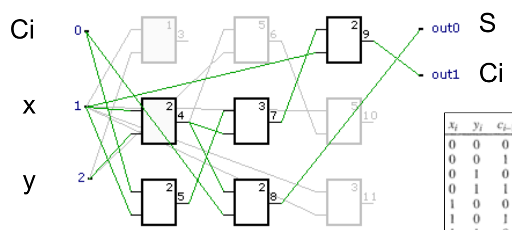
Part II.

Applications of evolvable hardware

We will follow the path:
simulations ⇒ FPGA ⇒ FPTA ⇒ nanoelectronics ⇒ evolvable systems on a chip

30

Extrinsic evolutionary circuit design at the gate level



1-bit full adder

x_i	y_i	c_{i-1}	c_i	s_i
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

- **Method:** Cartesian Genetic Programming – CGP (Miller&Tompson, 1999)
- **Array of PEs:** rows = 3, columns = 3, inputs = 3, outputs = 2
- **PEs**: NAND (0), NOR (1), XOR (2), AND (3), OR (4), NOT (5)
- **Chromosome**: (1,2,1) (1,2,2) (0,1,2) (4,2,5) (5,4,3) (4,0,2) (7,1,2) (1,6,5) (1,1,3) (8,9)
- **EA**: evolution strategy ES(1 + 5), 1 gene mutated
- **Fitness value**: the number of bits correctly calculated for all possible input combinations (max. 16 in this example)

31

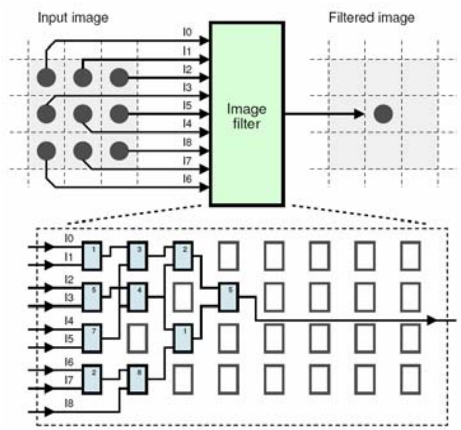
Extrinsic evolutionary circuit design: Results

Conventional multipliers of 2x2b – 4x4b improved!
- approx. 18% reduction in gate count.
- time of evolution grows exponentially with the growing number of inputs
- J. Miller et al, University of York, UK:
http://www.elec.york.ac.uk/intsys/users/jfm7/

32

Functional-level image filter evolution in FPGA

http://www.fit.vutbr.cz/~sekanina/pubs.php.en



	function	description
0	$x \vee y$	binary or
1	$x \wedge y$	binary and
2	$x \oplus y$	binary xor
3	x + y	addition
4	$x +^{S} y$	addition with saturation
5	(x + y) >> 1	average
6	Max(x,y)	maximum
7	Min(x,y)	minimum

$$fitness = \sum_{i=1}^{K-2} \sum_{j=1}^{K-2} v(i,j) - w(i,j)$$

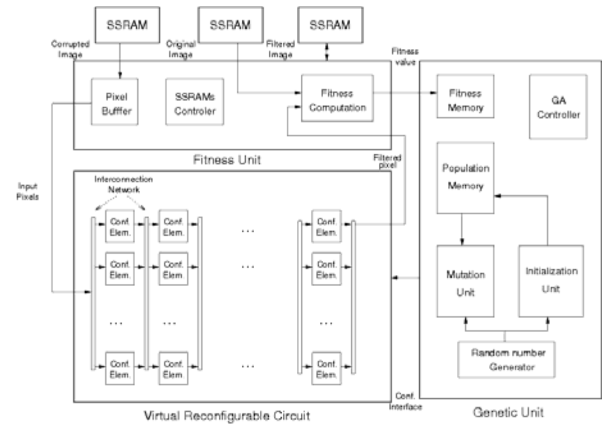
The goal is to minimize the difference between filtered image *v(i,j)* and original image *w(i,j)*. *K = 128*

33

ehw@FIT

Functional-level image filter evolution in FPGA

http://www.fit.vutbr.cz/~sekanina/pubs.php.en



f = 50MHz in Virtex XC2V3000

Speedup: 27x over Celeron 2.4GHz! (15 sec. to evolve a filter)

34

ehw@FIT

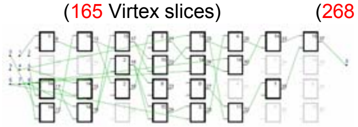
Evolved filter vs conventional median filter (for 5% salt-and-pepper noise)

5% noise

evolved filter applied (165 Virtex slices)

median – conventional filter (268 Virtex slices)

ehw@FIT



35

Bank of evolved filters

original image

40% noise

adaptive median (2220 slices)

evolved bank (500 slices)

ehw@FIT

Intrinsic evolution in FPGA (A. Thompson)

http://www.cogs.susx.ac.uk/users/adrianth/ade.html

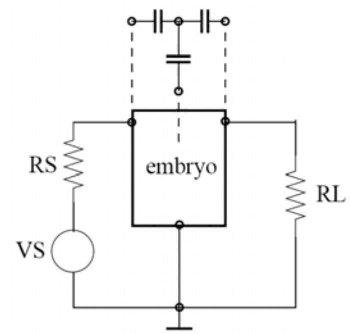
- Task: In FPGA XC6216, find a circuit which gives log. 1 when f_{input} = 10kHz and log. 0 when f_{input} = 1kHz.
- Thompson evolved a very strange solution which is completely outside the space of conventional designs.
- The resulting configuration works correctly only for the FPGA where it was evolved.
- Impossible to fully understand how the evolved circuit works.
- EA can exploit physical properties of a given platform!

37

Extrinsic evolution of analog circuits (J. Koza)

http://www.genetic-programming.org

- GP-based system – a program is evolved which modifies the embryo.
- development of an embryonic circuit
- candidate circuits evaluated using PSpice
- 1,000-Pentium parallel cluster computer used
- GP reinvented more than 30 previously patented inventions in the area of analog circuit design



38

Polymorphic gates

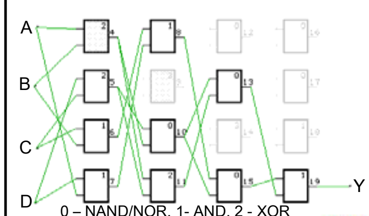
Stoica et al., http://ehw.jpl.nasa.gov

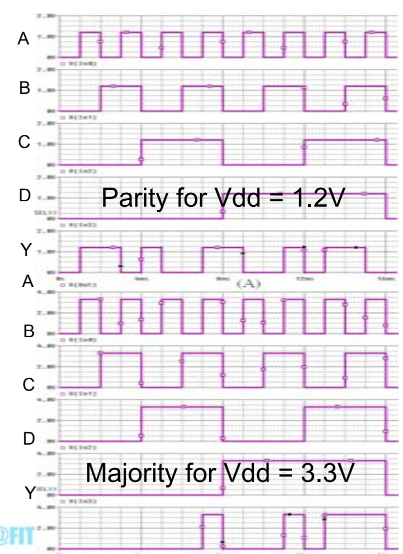
- Polymorphic gates perform two or more logic functions.
- Logic functionality of a polymorphic gate is determined in a non-traditional way: by the level of Vdd (power supply voltage), temperature, light etc.
- Example: evolved NAND/NOR gate operates as NAND for Vdd = 3.3V and as NOR for Vdd = 1.8V. (only 6 transistors, fabricated using CMOS HP 0.5u)
- Stoica, A. et al. : Taking Evolutionary Circuit Design From Experimentation to Implementation. IEE Proc.-Comp. Digit. Tech. Vol. 151(4) (2004) 295-300

39

Transistor-level simulation of a polymorphic circuit evolved at the gate level

Evolved 4-bit parity (Vdd = 1.2V) vs. majority (Vdd = 3.3V)





ehw@FIT

Sekanina, L., Martinek, T., Gajda, Z.: Extrinsic and Intrinsic Evolution of Multifunctional Combinational Modules. 2006 IEEE World Congress on Computational Intelligence, Vancouver, IEEE CIS, 2006, p. 9676-9683

Evolution in materio

(Miller & Harding, www.evolutioninmaterio.com)

- EA used to find values and positions of configuration voltages in order to manipulate with a suitable materio (liquid crystals used).
- The orientation of liquid crystal molecules can be controlled using electric field.
- Changing the orientation of molecules alters optical and electrical properties of liquid crystals.
- Tone discriminator, robot controller and other circuits evolved directly in liquid crystals.

41

NanoCell

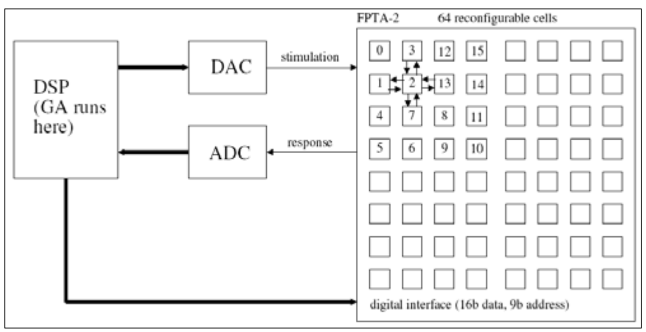
http://www.jmtour.com

- The nanocell is a 2D network of self-assembled metallic particles connected by molecular switches.
- The nanocell is surrounded by a small number of lithographically defined access leads.
- The nanocell is not constructed as a specific logic gate - the logic is created in the nanocell by training it postfabrication by changing the states of the molecular switches.
- In particular, nitroaniline molecules are switched through voltage pulses.
- The training algorithm does not know the connections within the nanocell or the locations of the switches. However, the configuration can be changed by voltage pulses applied to the I/O pins.
- Various molecular logic circuits evolved.

42

Evolutionary functional recovery in FPTA

Stoica et al., http://ehw.jpl.nasa.gov/



Circuit evolution in extreme environments:
- high temperatures (300°C)
- low temperatures (-196°C)
- cumulative radiation (250 krad)

43

Evolvable hardware

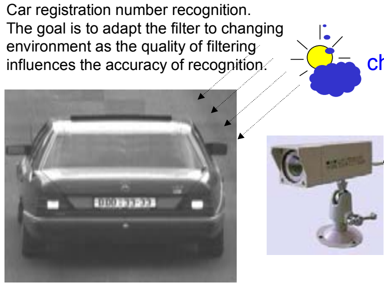
- EA is used during the <u>operational time</u> of a system to ensure
  - adaptation and/or
  - functional recovery

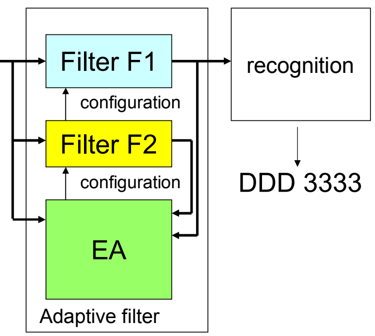
    when the specification is redefined, HW is corrupted or the environment is changed.
- EA is typically a part of the target system.
- Much more difficult than evolutionary design of a single circuit.
- Current practice: EA is used to tune a couple of parameters of a circuit on a chip.

44

Motivation: Online adaptation using EHW

Example of a real-world application:
Car registration number recognition.
The goal is to adapt the filter to changing environment as the quality of filtering influences the accuracy of recognition.





EA is a part of the target system.
EA is running forever - it has to improve the image filter F2 continually. When F2 is better than F1 then F1 is replaced by F2.

45

Problems in adaptive hardware

- The speed of evolution >> the speed of changes in the environment
  - In the example: EA is able to find a filter in 15 sec; however, several runs are needed to find a good filter. The environment is changed in order of minutes.
- EA as a stochastic algorithm can fail in a particular run.
- The design of fitness function is difficult
  - The fitness function usually requires a reference signal which is difficult to obtain.
    - In the example: We do not know the perfect image in each moment.
  - A possible solution is to monitor some global system characteristics (such as performance etc.) and use them in the fitness function.
    - In the example: We could optimize with the aim of keeping the average pixel intensity at a certain level. Not necessarily sufficient.

46

Application classes
(according to changes in the specification/environment)

- Embedded evolutionary design
  - spec. is not often changed
  - EA is started (e.g. manually) when it is needed
  - example: functional recovery, post-fabrication calibration
- Self-triggered evolution
  - spec. is changed (and EA is started) in the well-defined time points
  - example: image compression
- Adaptive systems
  - spec. is often changed
  - EA runs continuously with the goal to supply as good solution as possible for a given situation
  - two reconfigurable devices can be utilized
  - example: adaptive signal processing
- Online evolution
  - population of “individuals” (e.g. robots) coevolves

47

Real-world applications of EHW from AIST, Japan
http://unit.aist.go.jp/asrc/asrc-5/index_en.html

- image compression chip
- prosthetic hand controller
- postfabrication calibration of integrated circuits a communication systems
- adaptive clock timing adjustment
- etc.

48

Evolvable hardware is successful at the Human-Competitive Results Competitions (The Hummies) at GECCO

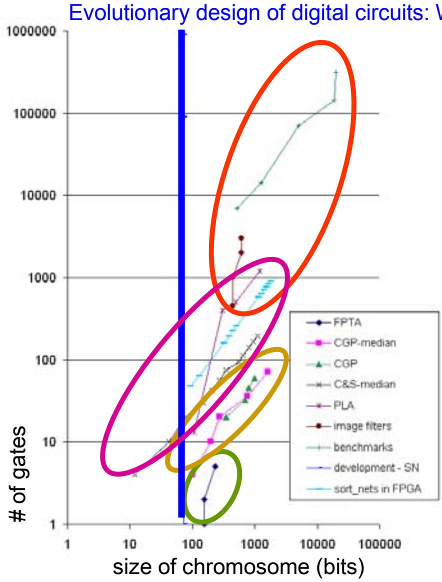
J. Koza: We say that an automatically created result is "human-competitive" if it satisfies one or more of the eight criteria below.

- **(A)** The result was patented as an invention in the past, is an improvement over a patented invention, or would qualify today as a patentable new invention.
- **(B)** The result is equal to or better than a result that was accepted as a new scientific result at the time when it was published in a peer-reviewed scientific journal.
- **(C)** The result is equal to or better than a result that was placed into a database or archive of results maintained by an internationally recognized panel of scientific experts.
- **(D)** The result is publishable in its own right as a new scientific result — *independent* of the fact that the result was mechanically created.
- **(E)** The result is equal to or better than the most recent human-created solution to a long-standing problem for which there has been a succession of increasingly better human-created solutions.
- **(F)** The result is equal to or better than a result that was considered an achievement in its field at the time it was first discovered.
- **(G)** The result solves a problem of indisputable difficulty in its field.
- **(H)** The result holds its own or wins a regulated competition involving human contestants (in the form of either live human players or human-written computer programs).

Human-competitive results awarded in areas
(from Koza's books and Hummies 2004 – 2006)

• Analog circuit design	26 results
• Quantum circuit design	8
• Physics	7
• Digital circuits/programs	7
• Classical optimization	5
• Game strategies	4
• Chemistry	4
• Antenna design	2
• Mathematics	1

Evolutionary design of digital circuits: Where are current limits?



development

functional level

PLA, spec. repres. (SN)

gate level

transistor level

In most cases the size of search space is approx. 2^{1000}.

Sekanina, L: Evolutionary Design of Digital Circuits: Where Are Current Limits? In: Proc. of the 1st NASA/ESA Conference on Adaptive Hardware and Systems, IEEE CS, 2006, p. 171-178

Main problems of EHW

- Scalability of representation: large systems ⇒ longer chromosomes ⇒ large search spaces ⇒ inefficient search
  - Solutions proposed: functional-level evolution, development, incremental evolution, …
- Evaluation time usually grows exponentially with the linearly increasing complexity of required circuits
  - Solution: use training set, estimate fitness, …
- No guarantee that evolved circuits will work correctly in other environments.
- A suitable result is not provided in each run of EA.
- It is not easy to find suitable applications!

Part III.

Properties of evolved circuits which are considered as computing devices

53

The world of abstract models of computing

The standard classical computation theory (including complexity and computability analysis) is philosophically and terminologically closely bounded to the theory of Turing machines.

Are computations physical processes or something very different: processes in abstract machines?

The implementation problem

Computation is not discovered in the physics, it is assigned to it. Certain physical phenomena are assigned or used or programmed or interpreted syntactically (J. Searle).

What computers can or cannot compute is determined by the laws of physics alone, and not by pure mathematic (D. Deutsch)

The world of physical computing devices

Physical limits: The ultimate laptop (1kg, 1 liter) could store and process 10^{31} bits of information and performi 10^{51} operations per second (Lloyd, Nature, Vol. 406, 2000)

54

Is evolutionary design different from conventional design within this context?

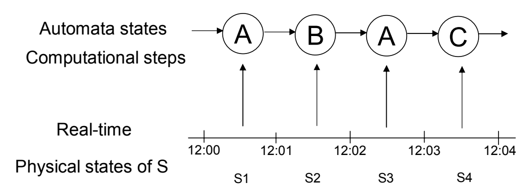
Computing mechanisms

- What is for a physical system to be a computing mechanism?
  - It is not clear! Many answers exist…

55

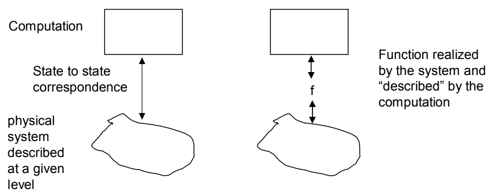
H. Putnam

- A system is a computing mechanism if and only if there is a mapping between a computational description and a physical description of the system.
- Putnam puts no constraints on how to find the mapping between the computational and the physical description, allowing any computationally identified state to map onto any physically identified state.
- Therefore, every ordinary open system is the realization of every abstract finite automaton.



56

M. Scheutz



- Scheutz's approach differs significantly from the Putnam's state-to-state correspondence view.
- The notion 'realization of a function', developed out of physical theories, is introduced as a replacement for the notional pair 'computation-implementation'.
- It does not require a notion of physical state, but determines directly the function which is realized by a physical system.
- It does not have to assume a particular computational formalism (to which the physical system exhibits a state-to-state correspondence), but can be related to computations described by any computational formalism via the function that these computations give rise.

Scheutz, M.: When Physical Systems Realize Functions ...Minds and Machines. 9(2), 161-196 (1999)

G. Piccicini

- A computing mechanism is a mechanism whose proper function is to obtain certain output strings of tokens from certain input strings (and internal states), according to a general rule that applies to all inputs and outputs.

An optimal account of computing mechanisms should satisfy the six desiderata:

1. Paradigmatic computing mechanisms compute.
2. Paradigmatic non-computing systems don't compute.
3. Computation is observer-independent.
4. Computations can go wrong.
5. Some computing mechanisms are not computers.
6. Program execution is explanatory.

- This account of computing mechanisms ``… allows us to formulate the question of whether a mechanism computes as an empirical hypothesis, to be decided by looking at the functional organization of the mechanism. It allows us to formulate a clear and useful taxonomy of computing mechanisms and compare their computing power".

58

Piccinini, G.: Computations and Computers in the Sciences of Mind and Brain. PhD thesis, University of Pittsburgh (2003) pp 323

C. Johnson

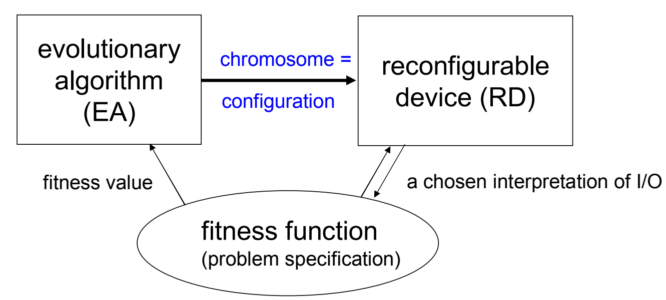
What Kinds of Natural Processes Can be Regarded as Computations?

- An important characteristic of computing is. . . "that symbols within the system have a consistent interpretation throughout the computation, or at least if they do not there is a component of the system which explains how the interpretation of the symbols changes as the computation progresses".
- That is, any external system which observes and/or initiates a computation must declare in advance how it is going to interpret those symbols.
- If there is not a consistent allocation of symbols then transformations are meaningless.
- It is important to make a distinction between consistent and deterministic here; this property does not exclude probabilistic actions being included in computations.

Johnson, C. G.: What Kinds of Natural Processes Can be Regarded as Computations? In: Computation in Cells and Tissues: Perspectives and Tools of Thought. Paton T. (ed) (Springer, Berlin Heidelberg New York 2004)

59

Evolve function *F* in a physical RD



- EA might find a solution (a configuration) *C*, which works correctly only in a given RD (*C* works neither in a circuit simulator nor in another RD of the same type) and only in the operational conditions (such as temperature,…), which have existed at the end of evolution (Thompson, 1996).
- In some cases we are not able to explain how RD implements *F*.

60

Is RD (configured by *C* and calculating *F*) a computing mechanism?

- Yes
  - If it is sufficient that
    - the interpretation of inputs and outputs is defined before EA is started and
    - nothing is known about the internal behavior of RD.
  - If RD is a computing mechanism then the functions computable by RD are also computable by the Turing machine.

61

Is RD (configured by *C* and calculating *F*) a computing mechanism?

- No
  - Symbols/physical phenomena inside RD must have a consistent interpretation which is established before EA is started.
  - However, it is impossible to define this interpretation before EA is started. EA can utilize "whatever" to implement *F*.
  - Unfortunately, it is also impossible (in general) to introduce this interpretation at the end of evolution, after an inspection of RD!
    - EA is able to utilize physical phenomena which have not been discovered by scientists so far.
    - Bartels et al.: Learning from Learning Algorithms: Applications to attosecond dynamics of high-harmonic generation. PHYSICAL REVIEW A 70(1):1-5, 2004

62

EA can generate behaviors that are not known to be physically possible

The winner of Humies 2005

- Bartels et al used EA to shape laser pulses to optimize a quantum process.
  - mJ-energy pulses from a 1 kHz repetition-rate laser system were focused into a 175 um diameter argon-filled hollow waveguide. This creates a phase-matched comb of harmonics containing the 23rd - 31st orders.
  - A deformable mirror pulse shaper was then used to selectively optimize the 27th harmonic, while simultaneously suppressing the 25th and 29th orders.
  - By optimizing the process of high-harmonic generation using shaped light pulses, a large data set was generated and statistically analyzed. This behavior was then compared with theoretical predictions (to verify understanding of the attosecond dynamics of high-harmonic generation) and an anomalous region of parameter space was uncovered.
- The results are completely unexpected and amazing from a physical point of view: behaviors are being evolved that were not known to be physically possible.
  - This includes anti-correlated attosecond harmonics in quantum systems.
  - The ability to move beyond the nanoscale to the attoscale is a major breakthrough, and the potential applications of controlling the behavior of materials at atomic level are enormous.
- Human can probe quantum systems, but are not capable of exploring different quantum behaviors in a fast automated fashion. In a sense, the results are beyond human competitive.

Bartels, R. et al.: Learning from Learning Algorithms: Applications to attosecond dynamics of high-harmonic generation. Physical Review A. 70(1):1-5 (2004)

63

A new class of computing devices

Property	Common computers	Brain	Evolved devices
I/O behavior can be interpreted as computing	YES	YES	YES
An abstract model exists before implementation	YES	NO	NO
The required behavior is specified beforehand	YES	NO	YES
Engineers can design&build	YES	NO	YES
Device is a computing mechanism	YES	?	?

- Physical realization is crucial for some evolved computing devices!!!
- We are not able to create an abstract model of their behaviors.
- See Sekanina, L.: Evolved Computing Devices and the Implementation Problem. Minds and Machines (to appear)

64

Conclusions

- Elementary concepts of evolutionary circuit design and evolvable hardware introduced in this tutorial.
- There are very nice applications of evolutionary circuit design and evolvable hardware!
- Some important topics omitted, for example, embryonics, developmental encodings etc.
- A computer science reflection of evolutionary circuit design presented – evolved systems exhibit a very unusual combination of properties when compared with computing devices designed conventionally.

65

References – evolvable hardware

- Gordon, T., Bentley, P.: On Evolvable Hardware. Soft Computing in Industrial Electronics, ed by Ovaska, S., Sztandera, L. (Physica-Verlag, Heidelberg 2001) pp 279—323
- Harding, S., Miller, J.: Evolution in materio: Initial experiments with liquid crystal. In: EH'04: Proc. of 2004 NASA/DoD Conference on Evolvable Hardware, ed by Zebulum, R. et al., Seattle, USA, 2004 (IEEE Computer Society, Los Alamitos 2004) pp 298—305
- Higuchi, T. et al.: Evolving Hardware with Genetic Learning: A First Step Towards Building a Darwin Machine. In: SAB'92: Proc. of the 2nd International Conference on Simulated Adaptive Behaviour (MIT Press, Cambridge MA 1993) pp 417-424
- Higuchi, T. et al.: Real-world applications of analog and digital evolvable hardware. IEEE Trans. on Evolutionary Computation. 3(3), 220-235 (1999)
- Koza, J. R. et al.: Genetic Programming III: Darwinian Invention and Problem Solving (Morgan Kaufmann Publishers, San Francisco CA 1999)
- Matoušek, R.: Reconfigurable Designs in FPGAs. Postgraduate Study Report FEL ČVUT Prague, DC-PSR-2003-10, 2003, 43 p.
- Miller, J., Job, D., Vassilev, V.: Principles in the evolutionary design of digital circuits - Part I. Genetic Programming and Evolvable Machines. 1 (1), 8-35 (2000)
- Sipper, M. et al: A phylogenetic, ontogenetic, and epigenetic view of bio-inspired hardware systems. IEEE Trans. on Evolutionary Computation. 1 (1), 83-93 (1997)

66

- Stoica, A., Keymeulen, D., Arslan, T., Duong, V., Zebulum, R., Ferguson I., Guo, X.: Circuit Self-Recovery Experiments in Extreme Environments. In: EH'04: Proc. of the 2004 NASA/DoD Workshop on Evolvable Hardware, ed by Zebulum, R. et al., Seattle USA, 2004 (IEEE Computer Society, Los Alamitos 2004) pp 142-145
- Stoica, A.: Evolvable Hardware for Autonomous Systems. Tutorial CEC, GECCO (2004)
- Thompson, A.: Hardware Evolution: Automatic Design of Electronic Circuits in Reconfigurable Hardware by Artificial Evolution. Distinguished Dissertation Series (Springer, London 1998)
- Thompson, A., Layzell, P., Zebulum, R. S.: Explorations in design space: unconventional electronics design through artificial evolution. IEEE Trans. on Evolutionary Computation. 3(3), 167-196 (1999)
- Tour J. M.: Molecular Electronics. World Scientific (2003)
- Yao, X., Higuchi, T.: Promises and Challenges of Evolvable Hardware. In: ICES'96: Proc. of the 1st International Conference on Evolvable Systems: From Biology to Hardware, ed by Higuchi, T. et al., Tsukuba, Japan, 1996. Lecture Notes in Computer Science, vol 1259 (Springer, Berlin Heidelberg New York 1997) pp 55-78
- Zebulum, R., Pacheco, M., Vellasco, M.: Evolutionary Electronics - Automatic Design of Electronic Circuits and Systems by Genetic Algorithms (CRC Press, Boca Raton 2002)

67

Other references

- Bartels, R. et al.: Learning from Learning Algorithms: Applications to attosecond dynamics of high-harmonic generation. Physical Review A, 70 (1):1-5 (2004)
- Copeland B. J.: What is Computation? Synthese. 108, 335-359 (1996)
- Deutsch, D.: Quantum theory, the Church-Turing principle and the universal quantum computer. Proceedings of the Royal Society London A, A400:97-117 (1985)
- Johnson, C. G.: What Kinds of Natural Processes Can be Regarded as Computations? In: Computation in Cells and Tissues: Perspectives and Tools of Thought. Paton T. (ed) (Springer, Berlin Heidelberg New York 2004)
- van Leeuwen, J., Wiedermann, J.: The Turing Machine Paradigm in Contemporary Computing. In: Mathematics Unlimited -- 2001 and Beyond (Springer, Berlin Heidelberg New York 2001) pp 1139—1155
- Piccinini, G.: Computations and Computers in the Sciences of Mind and Brain. PhD thesis, University of Pittsburgh (2003) pp 323
- Scheutz, M.: When Physical Systems Realize Functions …Minds and Machines. 9(2), 161-196 (1999)
- Searle, J.: Is the Brain a Digital Computer? Proceedings and Addresses of the American Philosophical Association 64, 21-37 (1990)
- Sekanina, L.: Evolvable Components: From Theory to Hardware Implementations. Natural Computing Series (Springer, Berlin Heidelberg New York 2004)
- Sekanina, L.: Evolutionary Approach to the Implementation problem. Habilitation thesis, Brno University of Technology, 2006
- Teuscher Ch. (ed.): Alan Turing: Life and Legacy of a Great Thinker (Springer, Berlin Heidelberg New York 2004)

68

References – Books on EHW

- Greenwood, G., Tyrrell, A.: Introduction to Evolvable Hardware. A Practical Guide for Designing Self-Adaptive Systems. IEEE Press Series on Computational Intelligence, 2006
- Higuchi, T., Liu, Y., Yao, X.: Evolvable Hardware. Springer Verlag, 2006
- Koza, J. R. et al.: Genetic Programming III: Darwinian Invention and Problem Solving (Morgan Kaufmann Publishers, San Francisco CA 1999)
- Koza, J. R. et al.: Genetic Programming IV: Routine Human-Competitive Machine Intelligence, Kluwer Academic Publishers, 2003
- Nedjah N., Mourelle, L. M.: Evolvable Machines. Studies in Fuzziness and Soft Computing. Springer Verlag, 2005
- Sekanina, L.: Evolvable Components - From Theory to Hardware Implementations, Berlin, DE, Springer, 2004, 194 p.
- Sipper, M.: Evolution of Parallel Cellular Machines - The Cellular Programming Approach. LNCS 1194, Springer, 1997, 198 p.
- Thompson, A.: Hardware Evolution: Automatic design of electronic circuits in reconfigurable hardware by artificial evolution, Distinguished dissertation series, Springer Verlag, 1998
- Zebulum, R., Pacheco, M., Vellasco, M.: Evolutionary Electronics - Automatic Design of Electronic Circuits and Systems by Genetic Algorithms. CRC Press, Boca Raton, 2002

69

Forthcoming conferences on EHW in 2007

- Evolvable Systems: From Biology to Hardware (ICES) – Wuhan, China
  - http://www.eccug.cn/ices2007.html
- NASA/ESA Adaptive Hardware and Systems Conference (AHS) – Edinburgh, UK
  - http://www.see.ed.ac.uk/ahs2007/AHS.htm
- Special tracks at GECCO, CEC, FPL

70

General Chair:
Greg Hornby
UC Santa Cruz/NASA Ames (USA)

Program Co-chairs:
Lukáš Sekanina
Brno University of Technology (CZ)
Pauline C. Haddow
Norwegian University of Science and Technology (N)

Publicity Chair:
Giovanni Squillero
Politecnico di Torino (I)

Local Organizing Committee:
R. Růžička; L. Sekanina
Faculty of Information Technology, Brno University of Technology
M. Zeithamlová – Action M Agency

Topics to be covered include, but are not limited to:

Evolutionary circuit diagnostics and testing
Self-reconfiguring/repairing and FT systems
Generative and developmental approaches
Hardware/software co-evolution
Real-world applications of evolvable hardware
MEMS and nanotechnology in evolvable hardware
Formal models for bio-inspired hardware systems
Novel devices/testbeds/tools for evolvable hardware
Evolutionary hardware design
Co-evolution of hybrid systems
Embryonic hardware
Intrinsic and extrinsic evolution
On-line hardware evolution
Evolutionary robotics
Adaptive computing

The ICES 2008 conference will be organized by the Faculty of Information Technology, Brno University of Technology. For further information please visit the ICES 2008 web site http://www.fit.vutbr.cz/events/ices2008 or contact Lukáš Sekanina at sekanina@fit.vutbr.cz

Important dates:
Deadline for submissions: **March 19, 2008**
Notification of acceptance: April 30, 2008
Camera-ready deadline: May 30, 2008

Thank you!

72