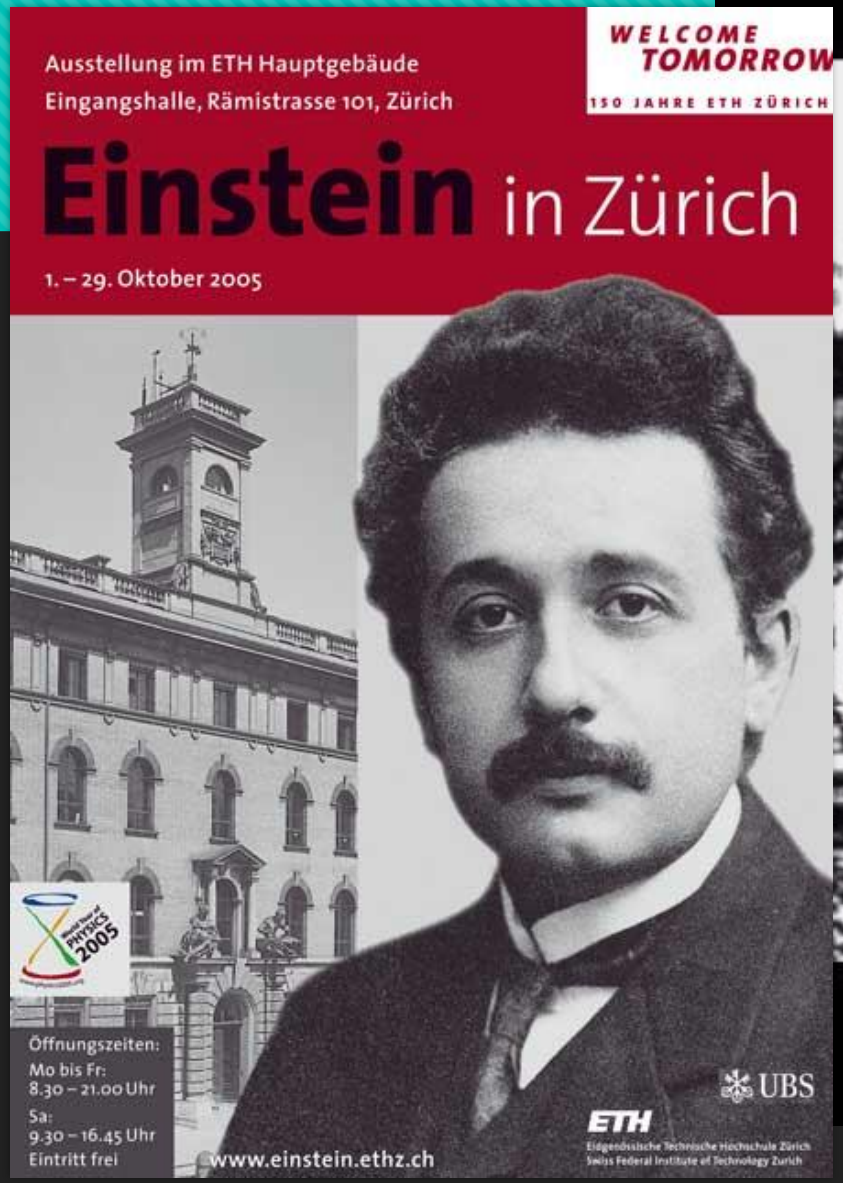
OBERON & Before and Beyond

Jürg Gutknecht, Russia, October 2018


Ausstellung im ETH Hauptgebäude
Eingangshalle, Rämistrasse 101, Zürich
WELCOME TOMORROW
150 JAHRE ETH ZÜRICH
Einstein in Zürich
1. – 29. Oktober 2005
World Year of PHYSICS 2005
Öffnungszeiten:
Mo bis Fr:
8.30 – 21.00 Uhr
Sa:
9.30 – 16.45 Uhr
Eintritt frei
www.einstein.ethz.ch
UBS
ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

ETH 1855

Programming Languages @ ETH

Year	Language	Concepts
1955	Algol/68 /W	Numerical math Procedures
1970	Pascal	Small scale programming Types
1980	Modula-2	Large scale programming Modules and Interfaces
1990	Oberon	Object-oriented programming Type extension
2000+	Active Cells	Hybrid programming Hardware description

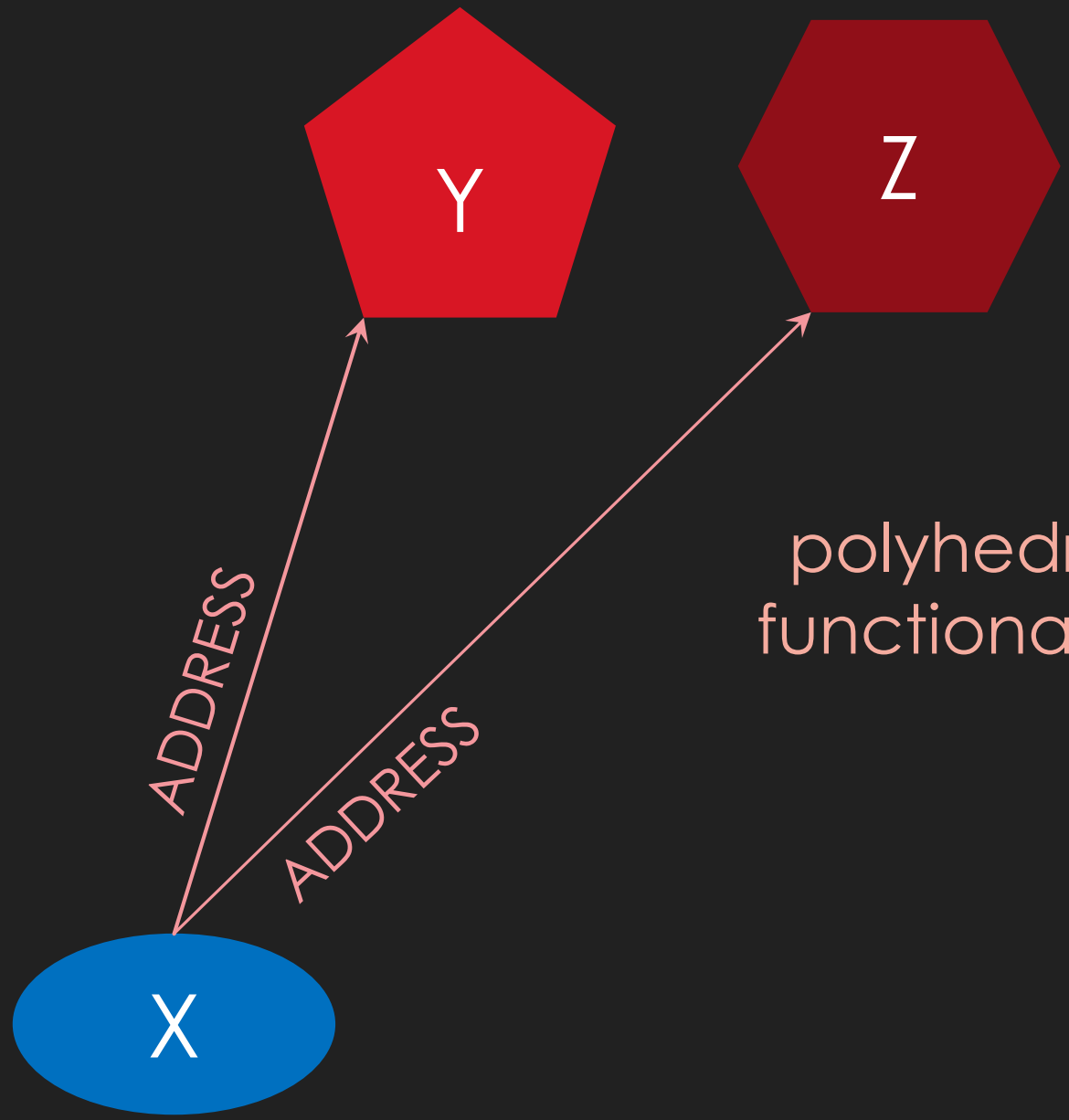
Guiding Design Principles

- Formally defined (BNF)
- Strongly type save
- Functionally orthogonal & closed
- Directly reflecting basic principles of construction
  - Stepwise refinement
  - Modular/hierarchical structures
  - Information hiding
  - Abstraction
- As simple as possible – but not simpler (quoting Albert Einstein)

Example for “simpler than possible”

Use of Modula-2’s type ADDRESS is too simple

- Contradicts type safety
- Breaks abstraction

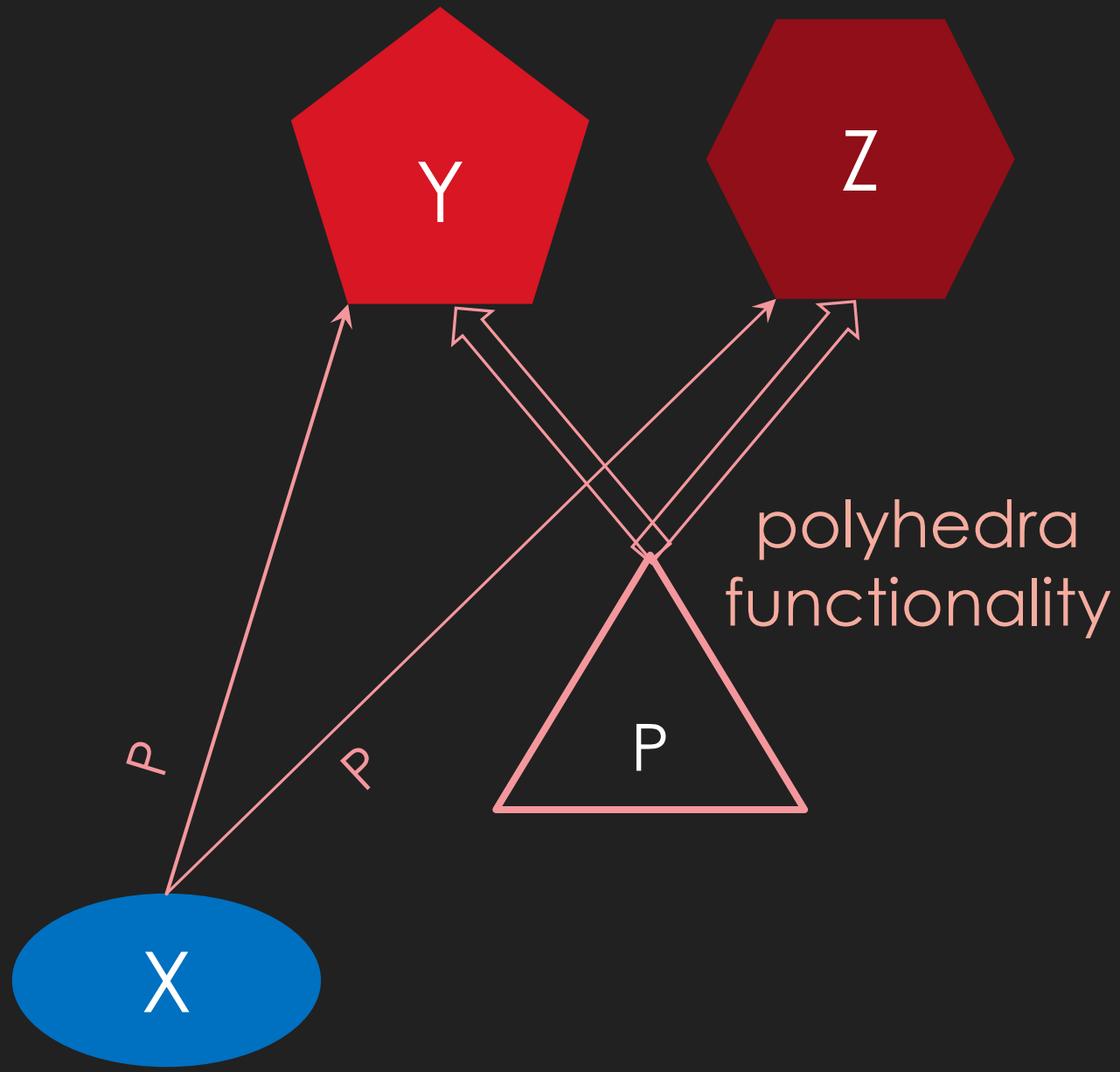


polyhedra functionality

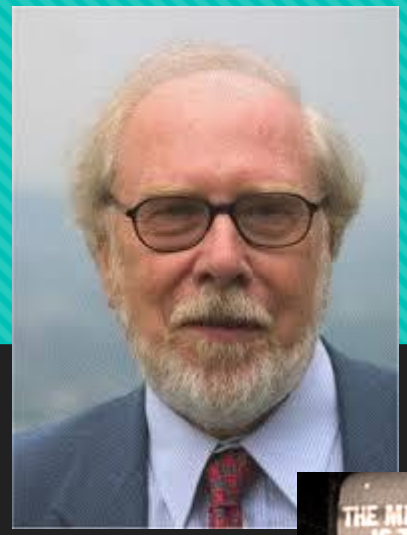
Example for "simpler than possible"

Adding type extension solves the pronlem

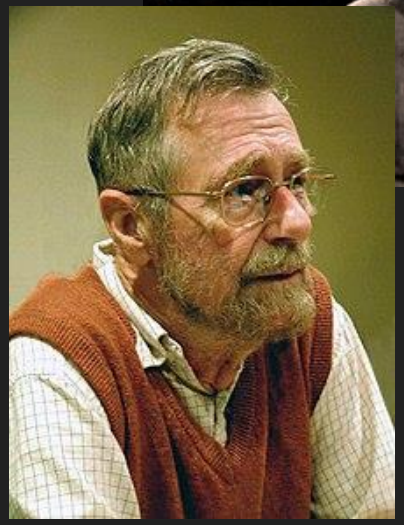
- Base type P
- Y "is-a" P
- Z "is-a" P



Why Does All this Matter?



- "With the first computer language, one not only learns a new vocabulary and grammar but one opens oneself to an new world of thought"
  Niklaus Wirth, Turing Award Laureate
- "We make our tools; then our tools make us"
  Marshall McLuhan, Canadian Philosopher
- "The tools we use have a profound influence on our thinking habits, and, therefore, on our thinking abilities"
  Edsger W. Dijkstra, Turing Award Winner



Codesign @ ETH: Strive for Coherence

Language/ Compiler

Operating System

Processor

Codesigned

Challenge: Expressive power of language

Von Neumann/ Harvard/ Multicore

ERMETH/ Algol

1955

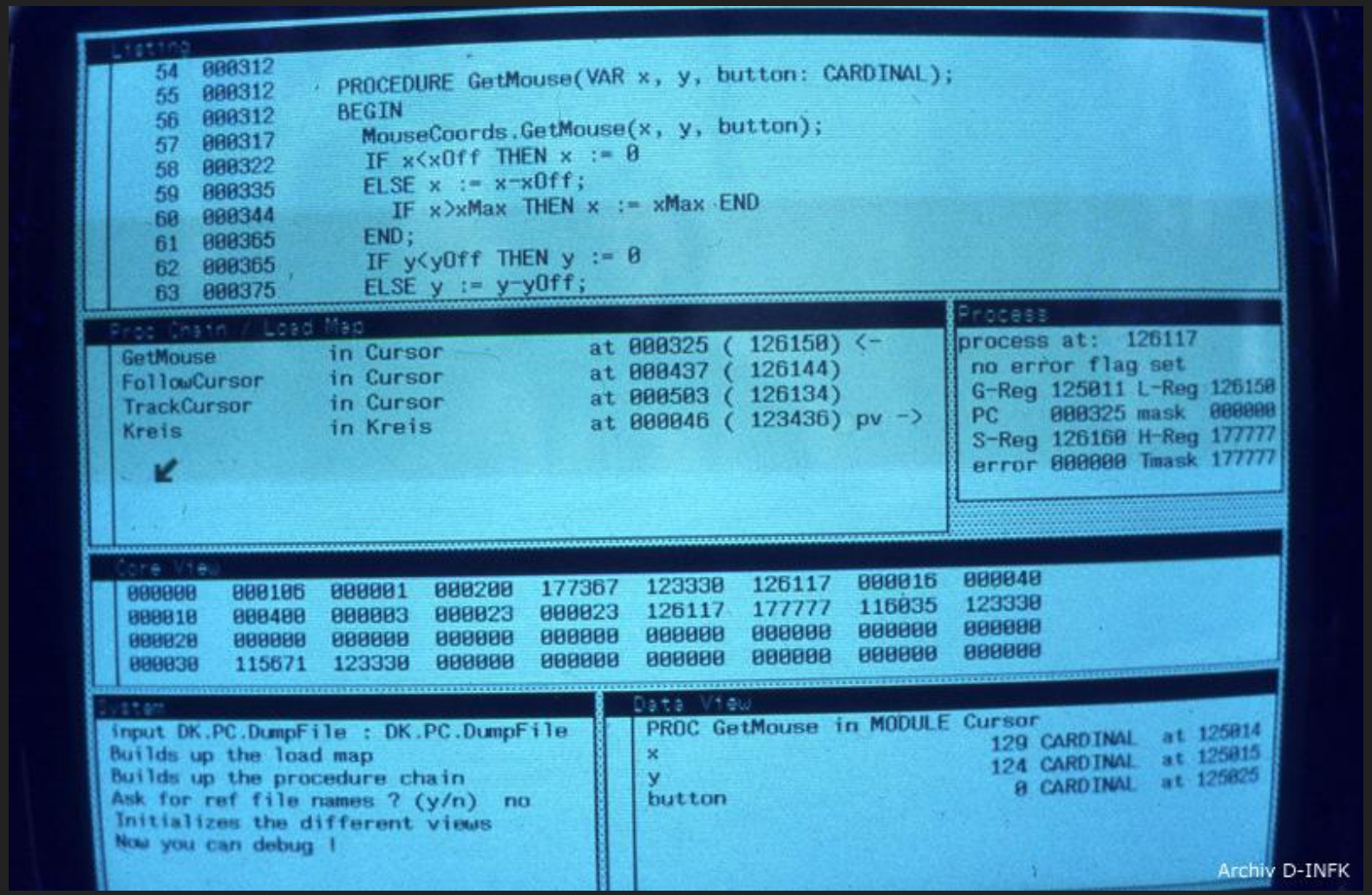
- Stiefel, Rutishauser
- Numerical Math

ERMETH

Lilith/ Modula-2/ Medos

1977

- Wirth and team
- Personal Computer
- Inspired by Xerox Alto



LILITH Screen with windows

Ceres/ Oberon

1984

- Wirth/ Gutknecht
- Workstation

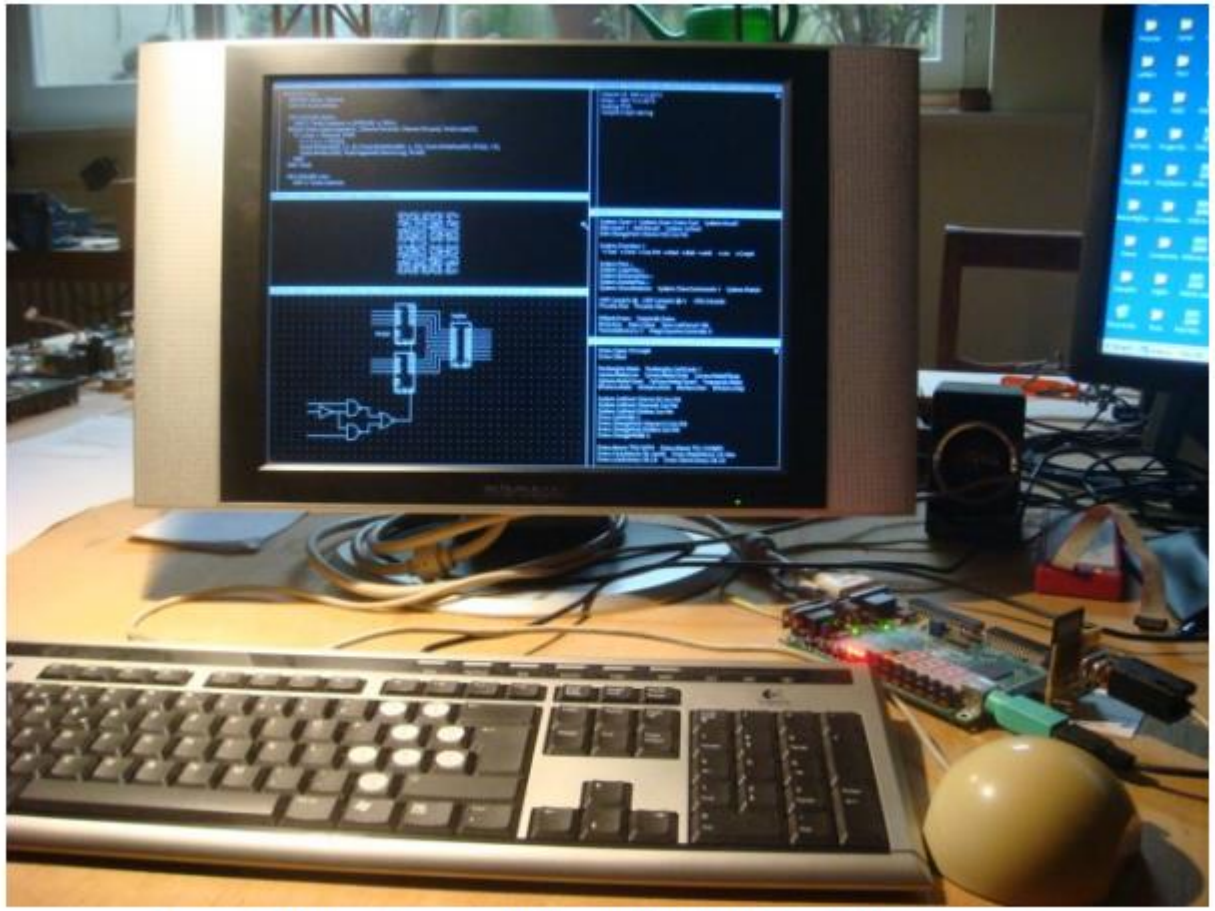


Figure 3.1 Typical Oberon display configuration with tool track on the right

Original Oberon tiling screen

Examples of Industrial Use of Oberon and Co.

Typically Small Business

Swiss Industry Controller (based on A2) Swiss Quality

Colortronic

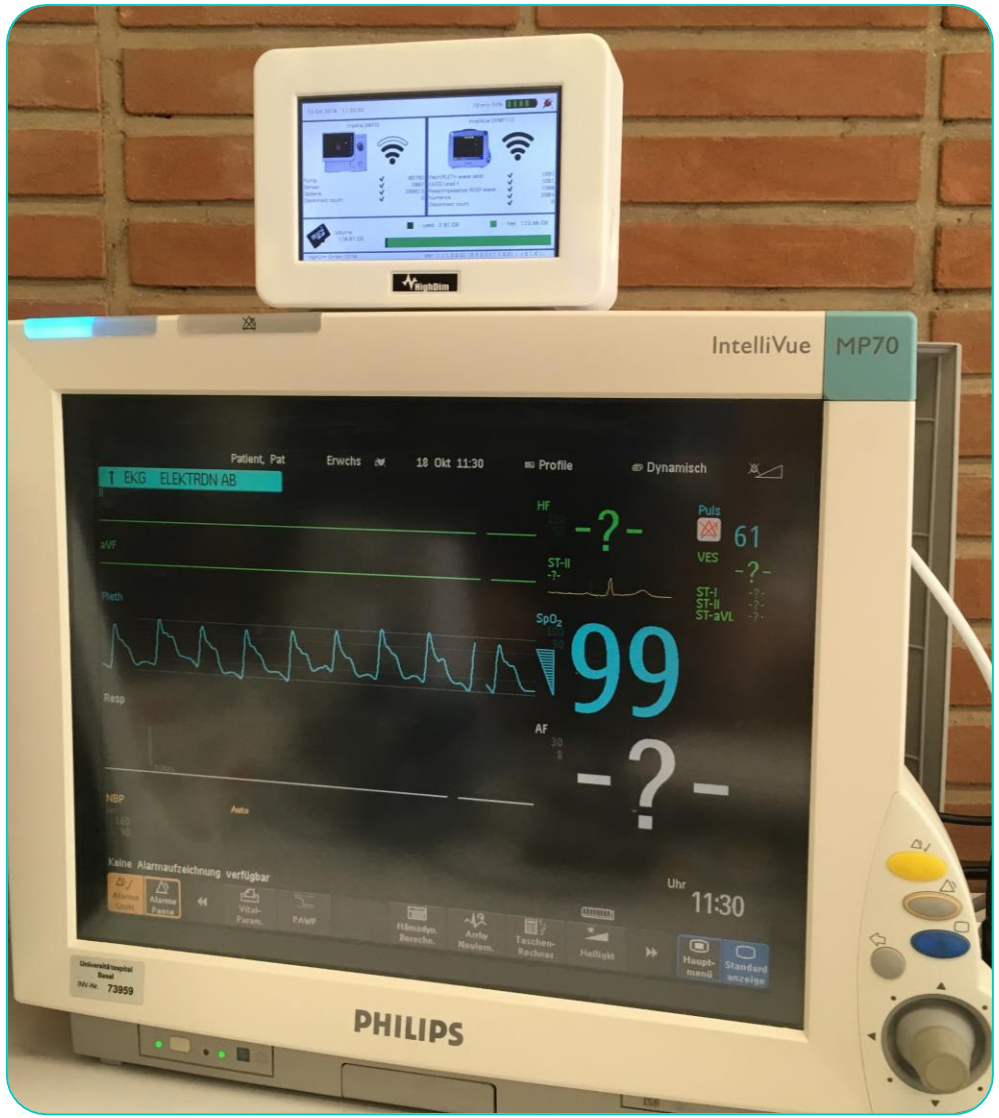
- High precision powder mixing

… in Action

Highdim University Hospital (based on Active Cells) Safety Critical

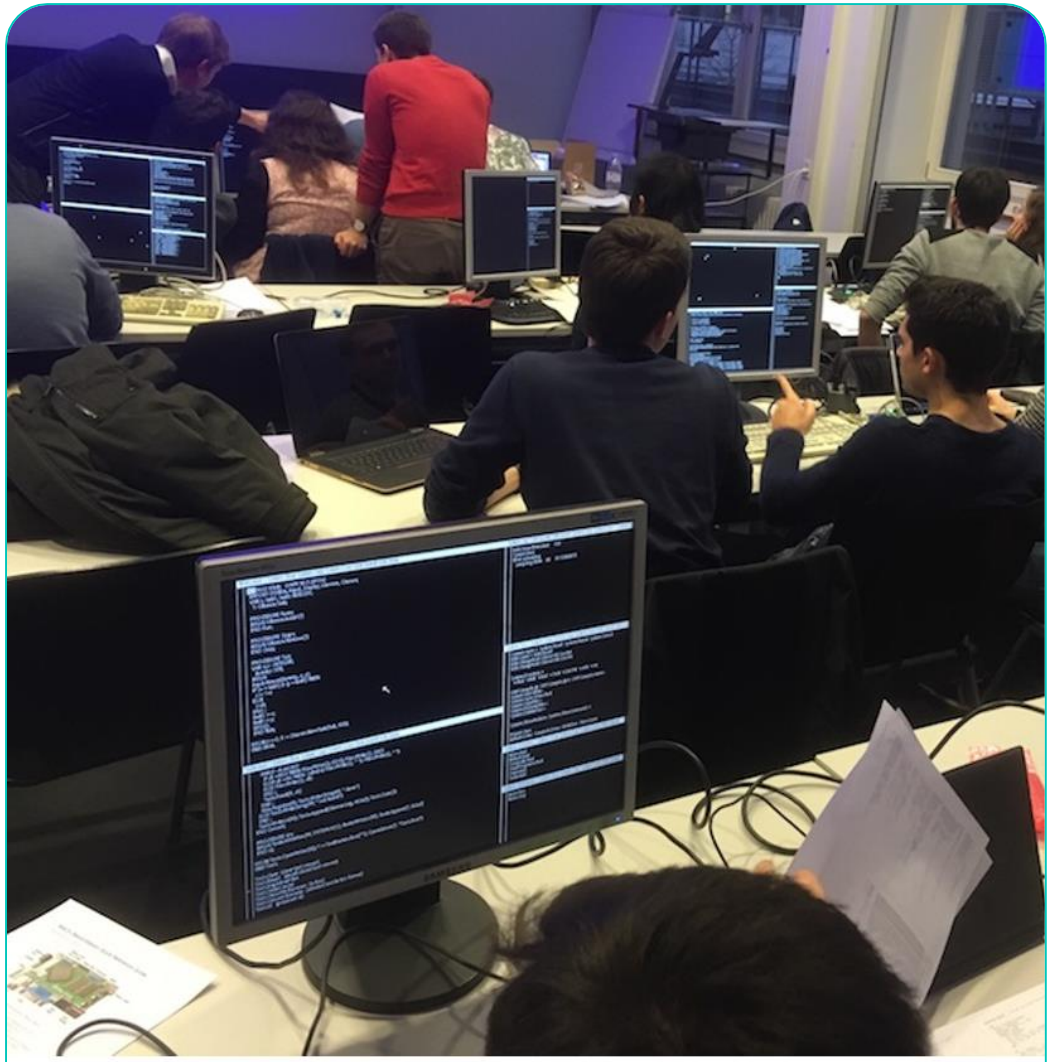
Medical IT

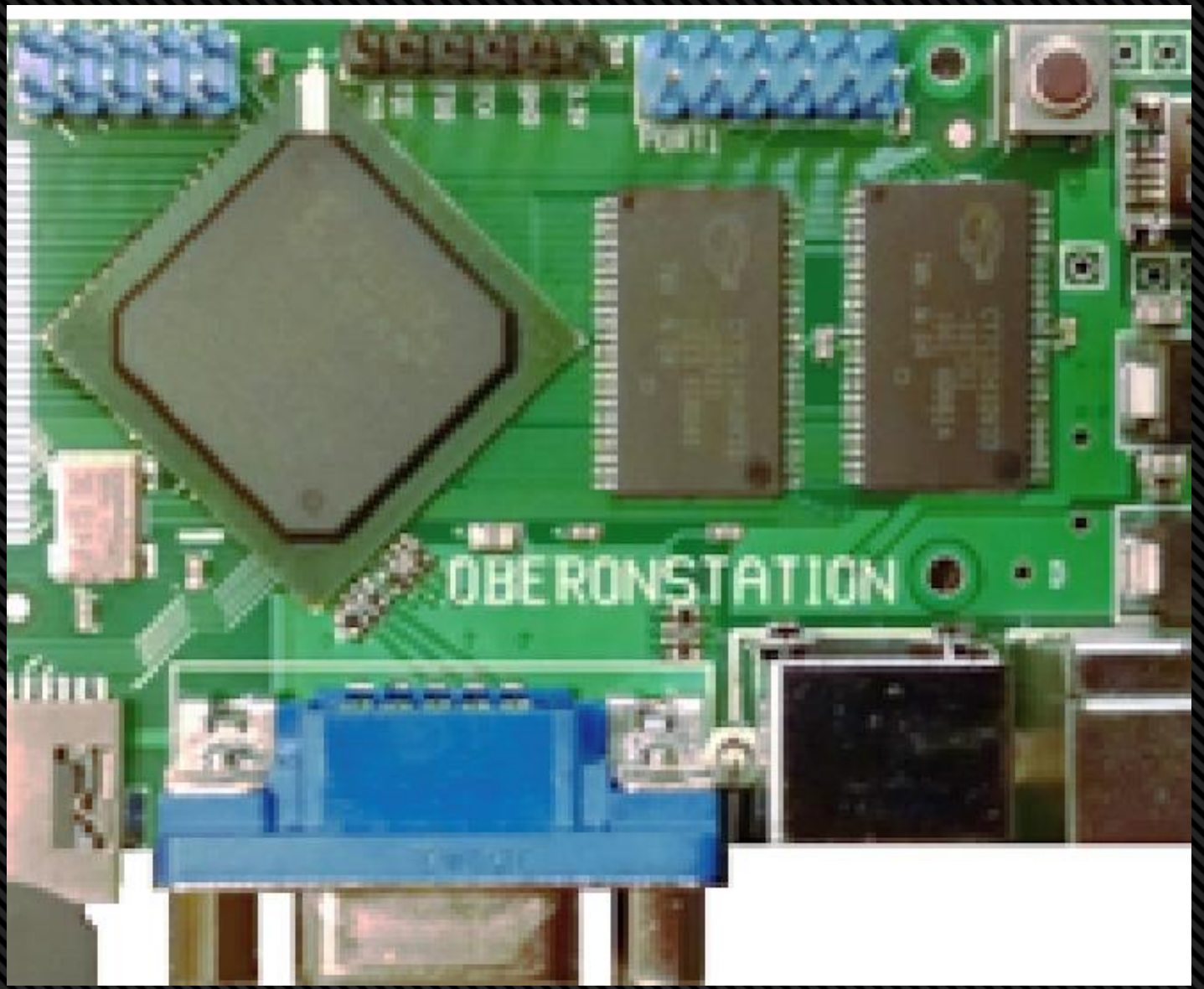
- Simulation of heart activity with implanted pump (Abiomed Impella)
- Deep learning for medical monitor signals
- Computational holography for malaria diagnostics



Many Others (some hiding)

- aurora swiss aerospace
  - Drone control
- Others
  - Running robot
  - Helicopter control
  - Material testing
  - Rehabilitation planning
  - Machine vision for aerospace

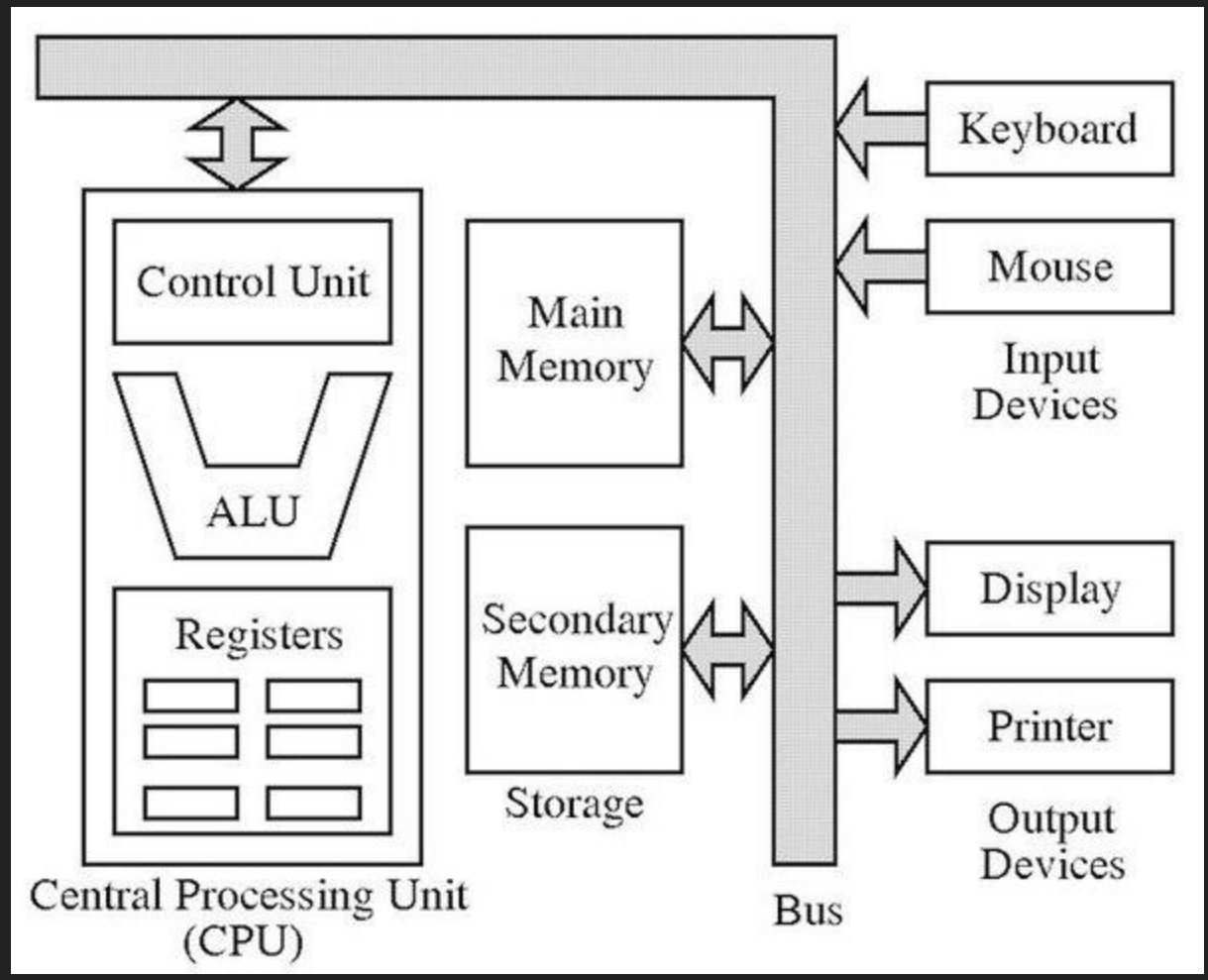



OBERONSTATION

The von Neumann Bottleneck

«One size does not fit all«

- Shared Memory Communication Bottleneck
- Cache Coherence Overhead
- Thread Synchronization Overhead



Codesign @ ETH: Strive for More Coherence

Language/ Compiler

Operating System

Processor

Codesigned

Challenge: Expressive power of language

Application-aware processor, neural net, …

The Vision: Hardware Compilation

High-level program code

Synthesizer, Simulator

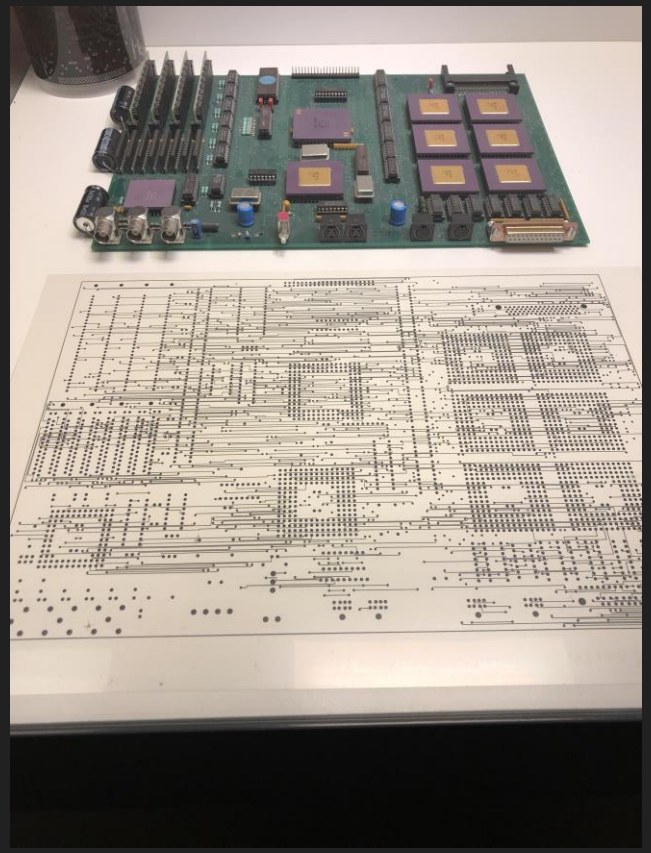
Electronic circuits as bitstream (FPGA, ASIC)

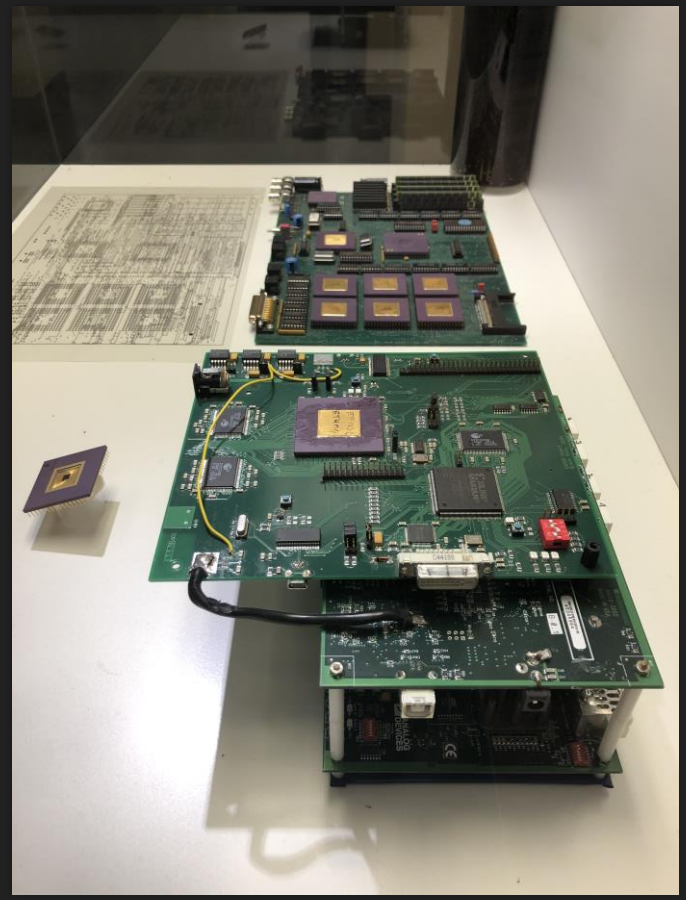
First Attempt: Project Switcherland

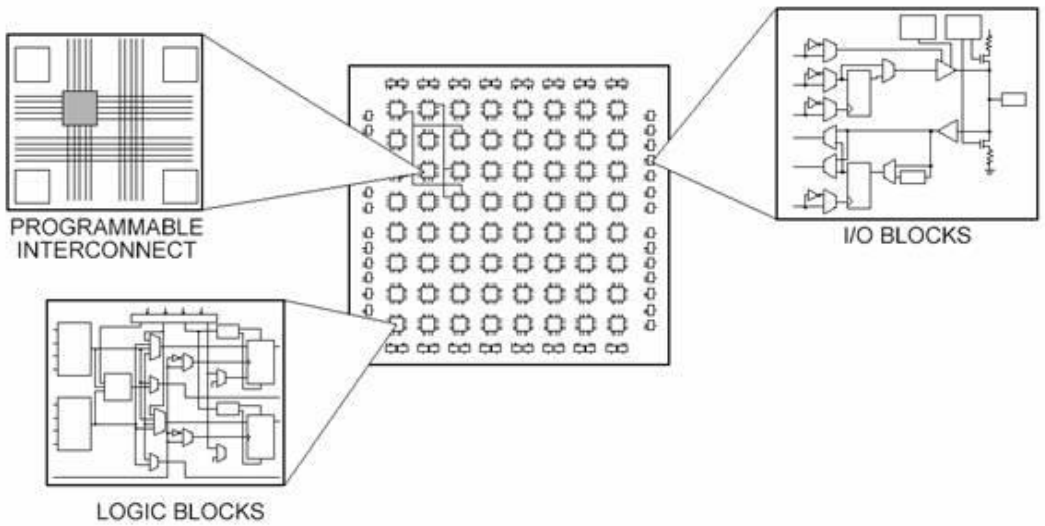
Chameleon/ Lola Multimedia system (Eberle and team)

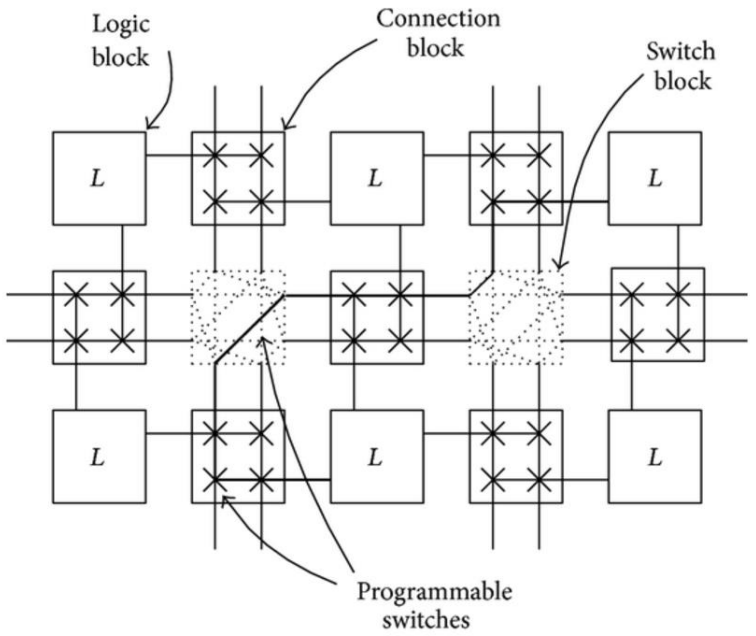
Allowed using ellipses and weigthed average instead of triangles as pixel primitives

Multimedia Chip by ETH









Enabler: Field Programmable Gate Arrays (FPGA)

High-Level Hardware Language Lola (Wirth, 1994)

- Describes static digital circuits instead of dynamic processes
- Variables represent signals or registers
- Values are defined as expressions of other objects and operators representing gate

Sample program

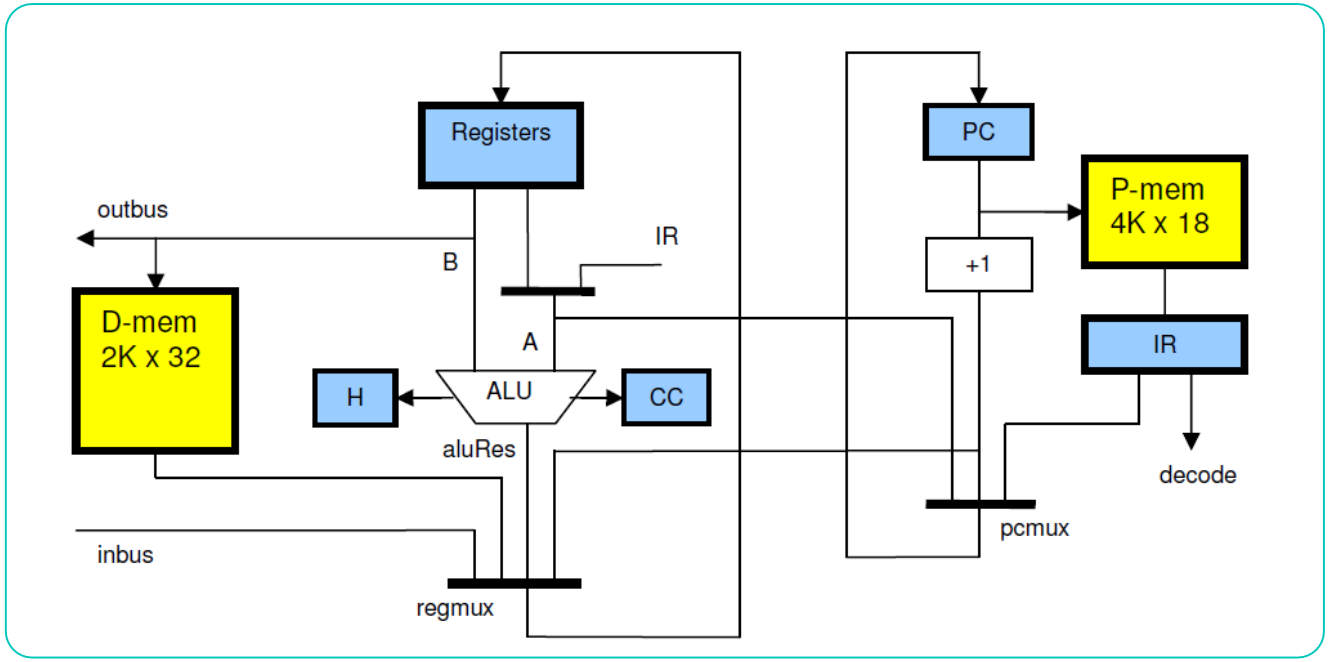
```
TYPE Counter = MODULE (IN clk, rst, enb: BIT; OUT data: WORD);
  REG (clk) R: WORD;
BEGIN
  data := R;
  R := ~rst -> 0 : enb -> R + 1 : R
END Counter
```

The Active Cells Tool Chain

Project Supercomputer in the Pocket

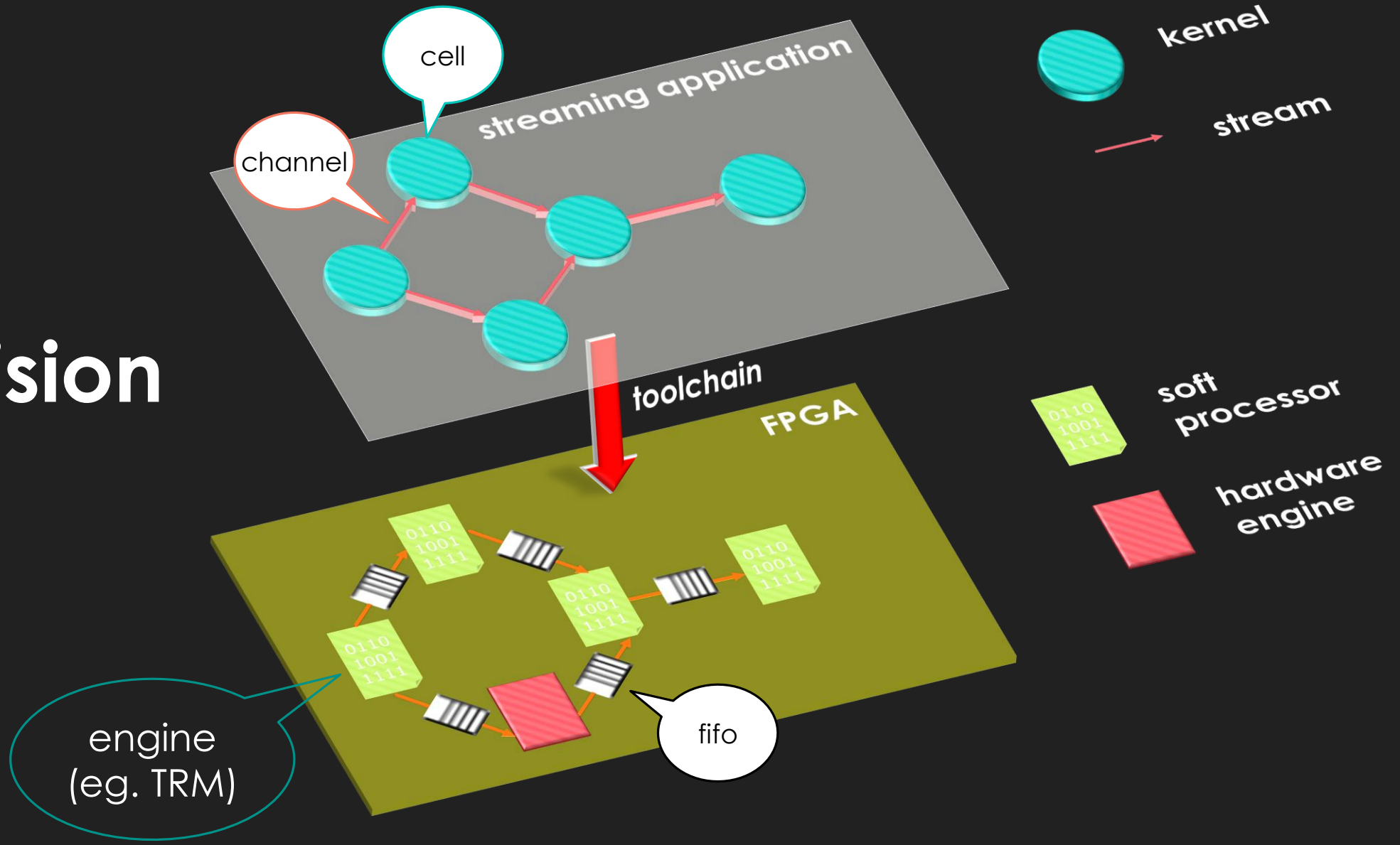
- Funded by Microsoft's Innovation Cluster Initiative, 2009 – 2014
- Topic: High performance custom architectures for embedded systems on the basis of programmable hardware (FPGA)
- Goal: Development of a corresponding tool chain
- Approach: On-chip distributed system based on ultra-simple processors

The Tiny Register Machine TRM (Wirth/ Liu)



Harvard Architecture

The Vision


cell
channel
streaming application
kernel
stream
toolchain
FPGA
soft processor
hardware engine
0110
1001
1111
engine
(eg. TRM)
fifo

Sample Code 1

```
type
    Adder = cell (in1, in2: port in; res: port out);
      var a1, a2: integer;
    begin
      in1 ? a1; in2 ? a2;
      res ! a1 + a2
    end Adder;
```

Single Cell

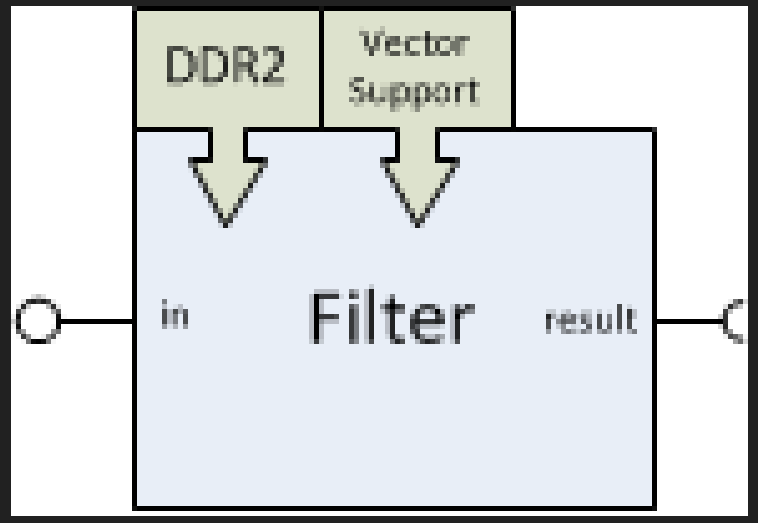
Sample Code 2

Cell constructor

```
type
    Filter = cell (in: port in; res: port out);
      var ...; len: integer;
      procedure & Init(len: integer)
        begin self.len := len
      end Init;
      begin (* ... filter action ... *)
      end Filter;
      var f: Filter;
      begin
        … new(f, 32);
      ...
```

Sample Code 3

Capabilities




```
Type
    Filter = cell
        { VectorTRM, DataMemory(2048), DDR2 }
        (in: port in (64); result: port out);
        var ...
    begin (* ... filter action ... *)
    end Filter;
```

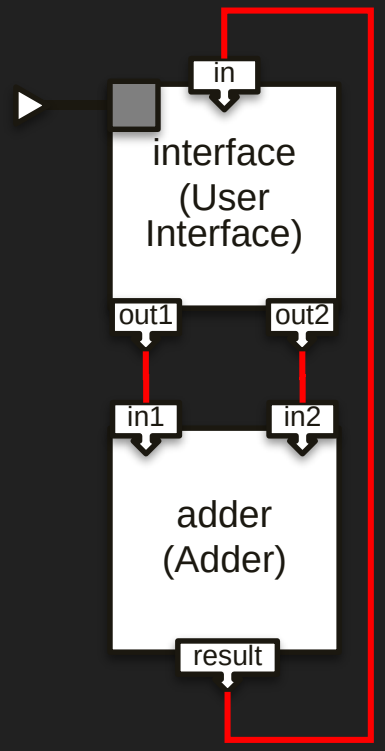
Sample Code 4

```
type
    Convolver2d =
      cell { Engine } (
          in: port in (64); result: port out);
    end Convolver2d;
```

Cell made from hardware

Sample Code 5

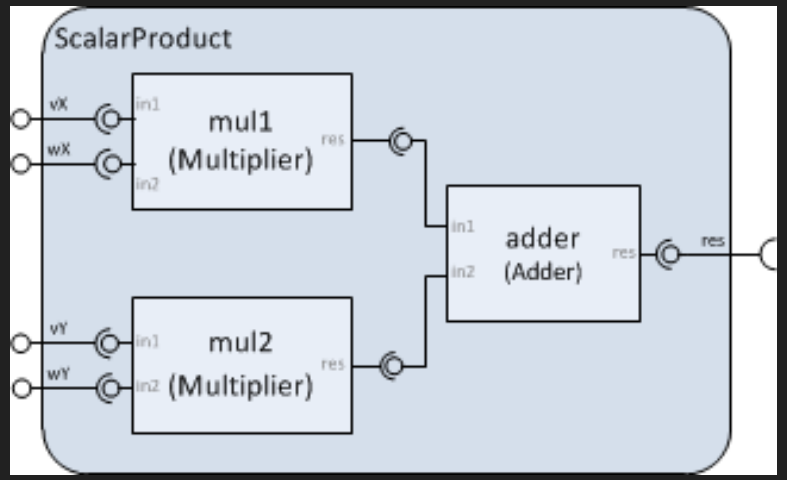
Cell net




```
cellnet Example;
    import RS232;

type
    UI = cell {RS232}
      (out1, out2: port out; in: port in) (*...*)
    end UI;

    Add = cell(in1, in2: port in; out: port out)
      (* ... *)
    end Add;

  var myUI: UI; add: Add;
begin new(myUI); new(add);
    connect(myUI.out1, add.in1);
    connect(myUI.out2, add.in2);
    connect(adder.result, myUI.in);
end Example.
```

Sample Code 6

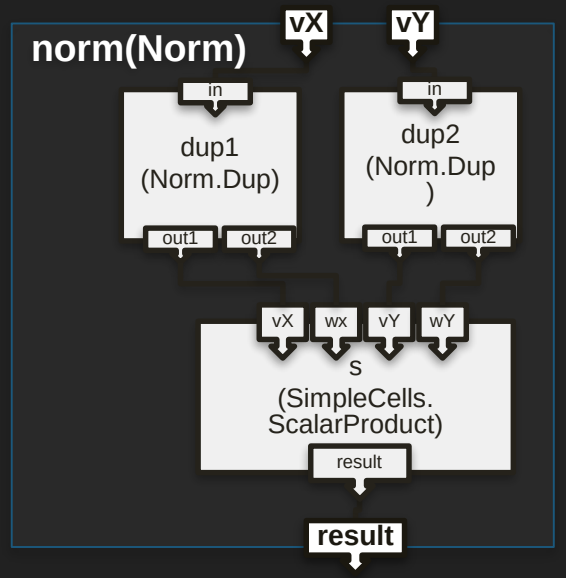
Hierarchic composition



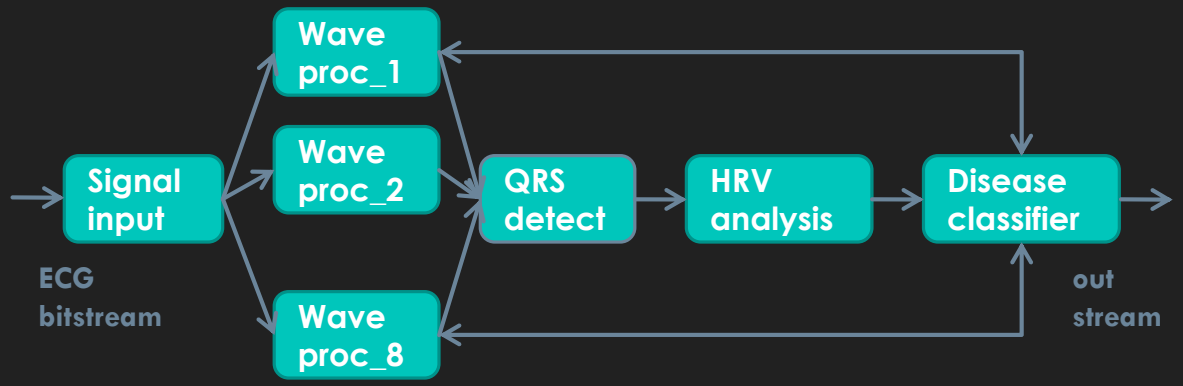

```
module SimpleCells
    import RS232;

type Add = cell (in1, in2: port in; res: port out (* ... *)
    end Add;

    Mult = cell (in1, in2: port in; res: port out) (* ... *)
    end Mult;

    Prod*= cellnet (x, y, u, v: port in; res: port out)
      var a: Add; m1, m2: Mullt;
    begin
      new(a); new(m1); new(m2);
      delegate(x, m1.in1); delegate(u, m1.in2);
      delegate(y, m2.in1); delegate(v, m2.in2);
      connect(m1.res, a.in1);
      connect(m2.res, a.in2);
      delegate(res, a.res)
    end Prod;

end SimpleCells
```

Sample Code 7

Wired Cellnet




```
cellnet Sample;
  import SimpleCells, RS232;
  type
    Norm* = cellnet (vX, vY: port in; result: port out)
      type
        Dup* = cell(in: port in; out1, out2: port out)
          var val: LONGINT;
        begin
          loop in ? val; out1 ! val; out2 ! val end
        end Dup;
    var s: SimpleCells.Prod; d1, d2: Dup;
    begin new(s); new(d1); new (d2);
      connect (d1.out1, s.vX);
      connect(d1.out2, s.wX);
      connect(d2.out1, s.vY);
      connect(d2.out2, s.wY);
      delegate(vX, d1.in); delegate(vY, d2.in);
      delegate(result, s.res);
    end Norm;
```

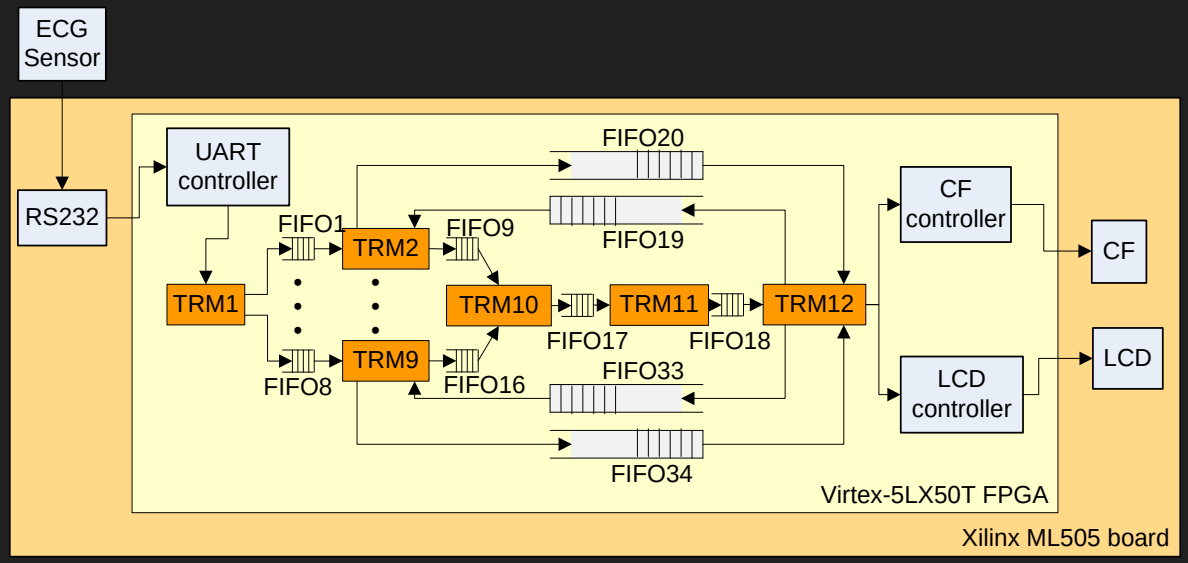


Computational scheme

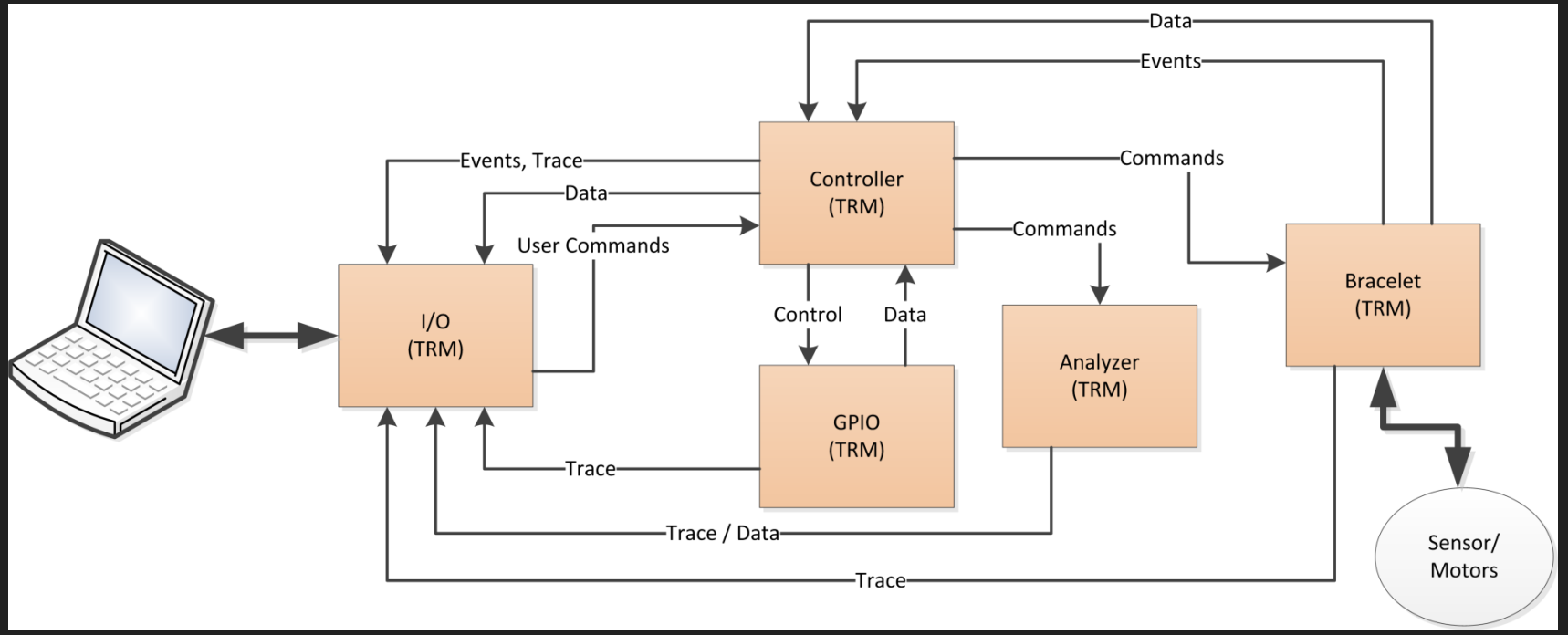
Hybrid compilation

Case Study
from Medical IT:
Realtime ECG Monitor

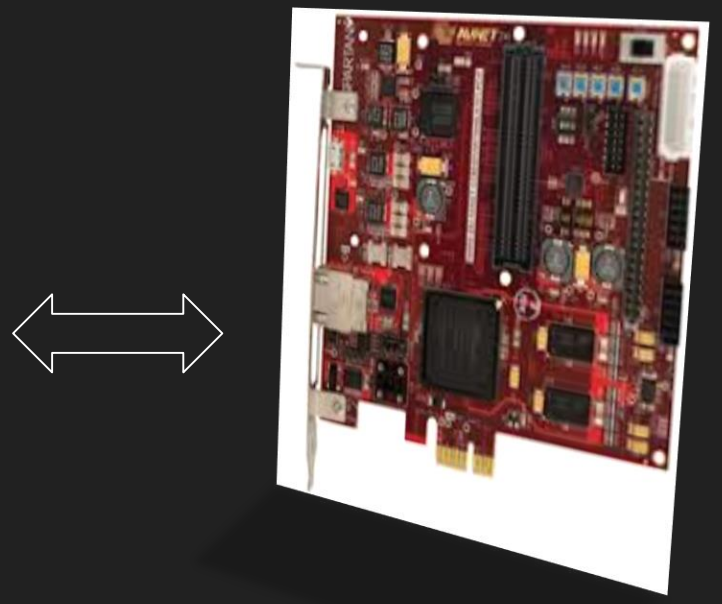
Hardware layout on FPGA



Medical Monitoring with Network On Chip



The Entire System



A2 host OS with GUI

sensor control and medical algorithms< on Spartan 6 FPGA

Sensors and motors on bracelet

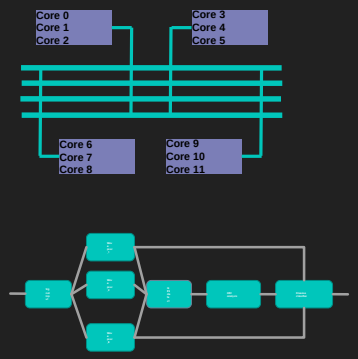
Use of FPGA Resources

8 physical channels @ 500 Hz sampling frequency each

#TRMs	#LUTs	#BRAMs	#DSPs	TRM load
12	13859 (48%)	52 (86%)	12 (25%)	**<5%** @116 MHz

Virtex-6 with 500 TRM capacity!

Power Usage



System Architecture	Static Power (W)	Dynamic Power (W)
Preconfigured multicore ("TRM12")	3.44	0.59
Dynamically configured	0.5	0.58

86% saving!

More details on Active Cells via
felix.friedrich@inf.ethz.ch

The Swiss Oberon Alumni Community

Picture taken on October 22, 2018

Our Most Famous Alumnus

Urs Hölzle, Google No. 8,
Chief Officer of Infrastructure

Conclusions

- No more need to coerce all software to von Neumann hardware architecture
- New technologies such as FPGA allow adjusting hardware architecture to specific application, with the benefits of much better resource and runtime efficiency
- Viable option to use same programming language spirit for both software and hardware description
- Comprehensive system descriptions and hybrid compilers are a feasible solution

Thank you for your interest and attention

- Questions welcome