Aspects of Embedded Systems Design and Methodology

Embedded Systems Design Course, 2020-12-02

Tiberiu Seceleanu

1

Outline

1. Embedded Systems: what ?
2. Platforms
3. Design: flow(s), challenges, solutions, results
4. “New” technologies
5. Recent research and advances
6. Questions ?

2

Embedded systems ?

- Computing systems are all-around
- Still, most of us think of “desktop” hardware and the related applications
  - PC’s
  - Laptops
  - Mainframes
  - Servers
- But…
  - Look into your pockets !

3

Embedded systems

- Embedded computing systems
  - Computing systems embedded within electronic devices
    - That is, HW and SW !
  - Billions of units produced yearly, versus millions of desktop units
  - Tens per household
- Application areas:
  - Communication
  - Computer Peripherals
  - Industrial Control and Automotive
  - Consumer Electronics
  - Test and Measurement
  - Medical
  - Military/Aerospace
  - ETC !!!

- Hard to define. Nearly any computing system other than a desktop computer
  - Cell-phones
  - Automatic Teller Machine
  - The Digital Interfaced Gasoline Station
  - Airborne Flight Control System
  - Automotive Engine Health Monitoring System
  - Home Security Systems
  - Modern Air-conditioners
  - Washing Machines
  - Medical Equipment
  - DVD Players
  - Printers
  - Medical Equipment

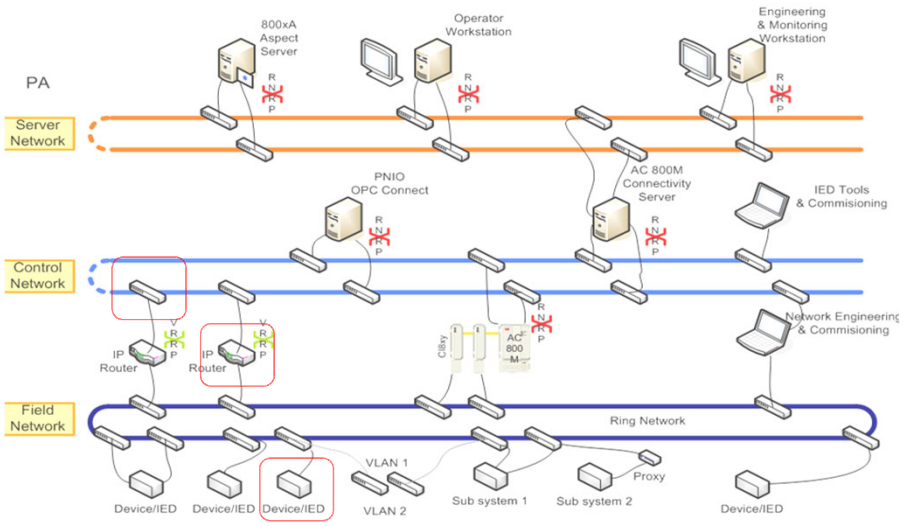
4

In general, embedded systems

- Are single / few - functioned
  - Execute a single (set of) program(s), repeatedly, or whenever requested / necessary
- Are highly - constrained
  - Low cost (**large volumes**),
  - Low power (**many operate on batteries**),
  - Small (**many are man-portable**),
  - Fast (**may require high performance**), **etc**.
- Usually are reactive, sometime also with real-time requirements
  - Continually reacts to changes in the system's environment
  - Must compute certain results with deterministic latency

5

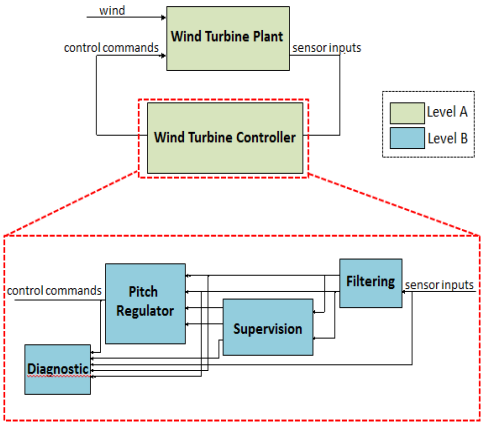
Process Automation Domain



6

Case Study

Wind Turbine Controller

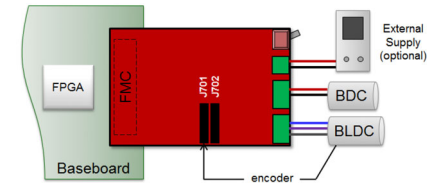


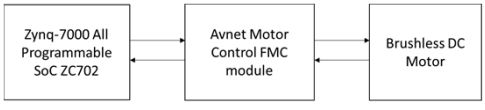
Picture from http://www.nrel.gov/wind/

7

Case Study

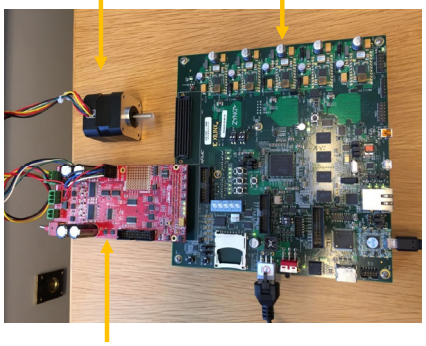
BLDC Motor





Xilinx ZC702 Zynq board

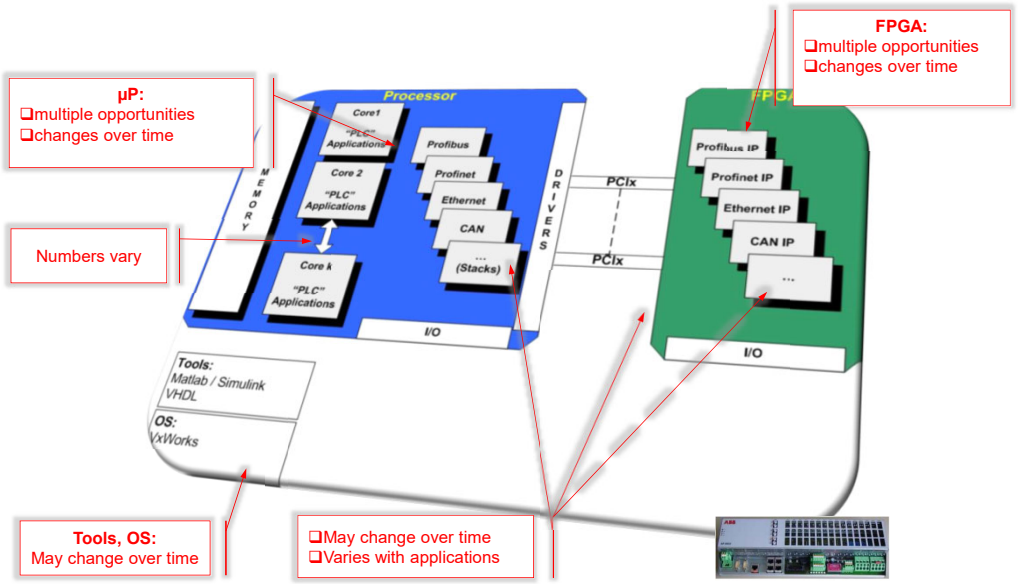
Anaheim BLDC motor



Avnet FMC motor contol board

8

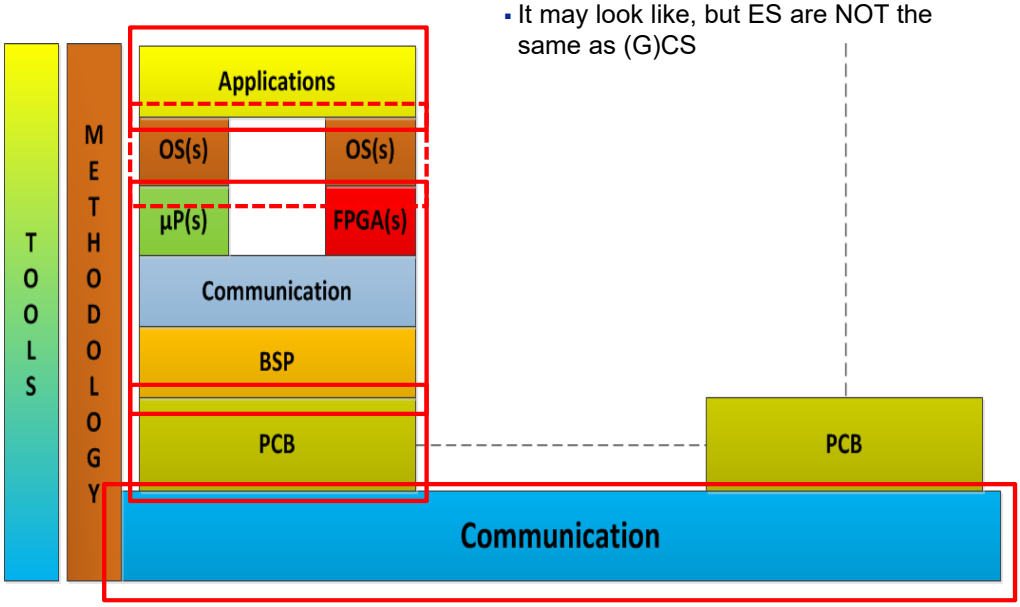
Industrial Processing Platform



9

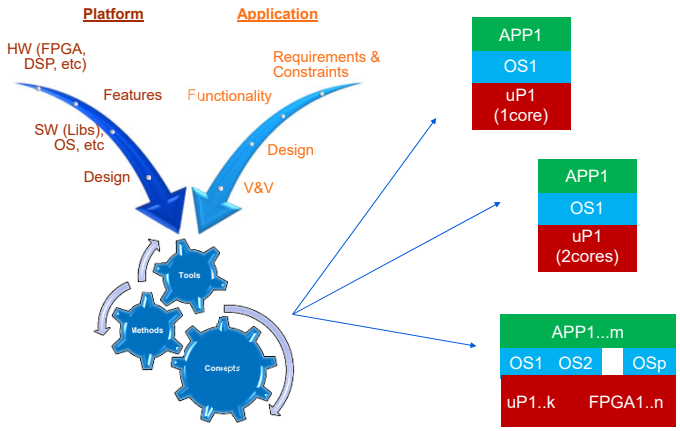
Platform ?

- It may look like, but ES are NOT the same as (G)CS



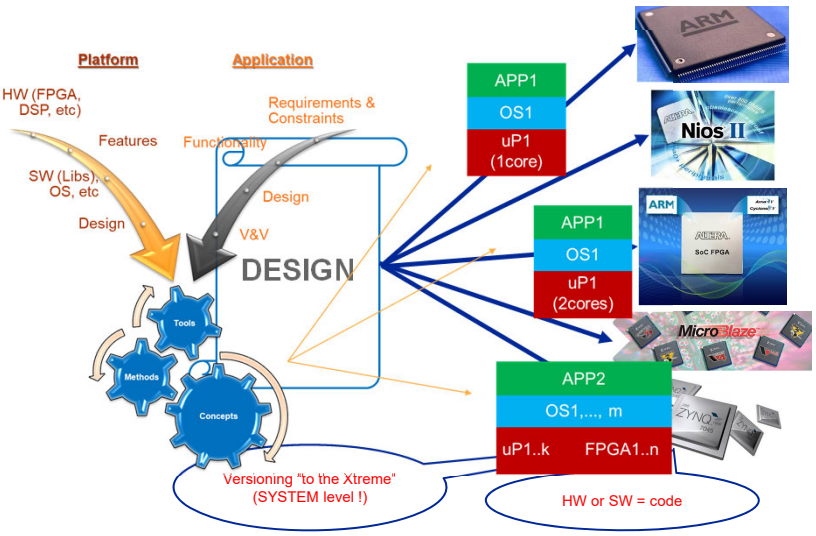
10

Vision



11

Vision



12

Things to be considered…
(one out of very many !)

```
#include "xparameters.h"
#include "stdio.h"
#include "xbasic_types.h"

#include "uartlite_header.h"
#include "mb_interface.h"

#include "time.h"

#include "sys/time.h"
#include "sys/types.h"

int main (void) {

   clock_t c1,c2;
   double dur;

   Xuint32 arr[400],i;

   for(i=0;i<400;i++)
      arr[i]=i;

   c1=clock();

   for(i=0;i<400;i++)
      arr[i]=2*arr[i];

   c2=clock();

   dur=(double) ((c2-c1)/CLK_TCK);

   xil_printf("\r\nDuration is %f", dur);
   return 0;
}
```

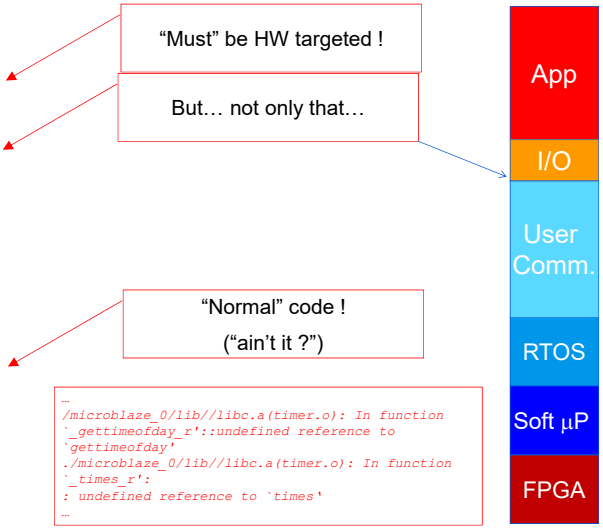
"Must" be HW targeted !

But… not only that…

"Normal" code !

("ain't it ?")

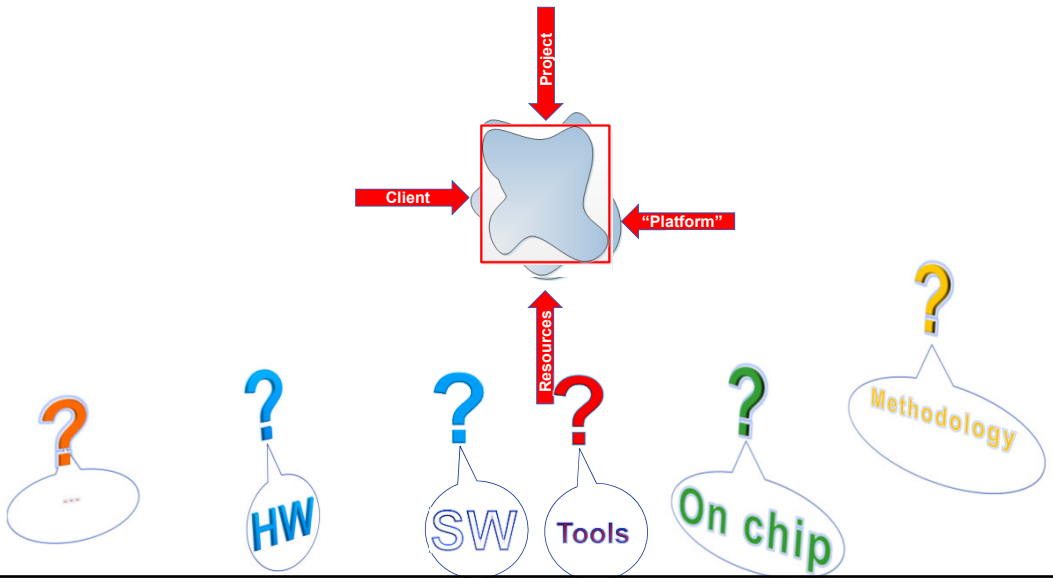
```
…
/microblaze_0/lib//libc.a(timer.o): In function
`_gettimeofday_r':undefined reference to
`gettimeofday'
./microblaze_0/lib//libc.a(timer.o): In function
`_times_r':
: undefined reference to `times'
…
```



13

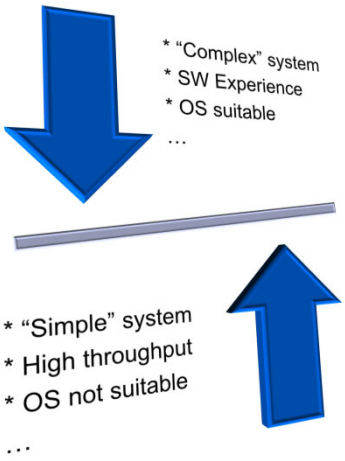
What we will design
What we can / want to design

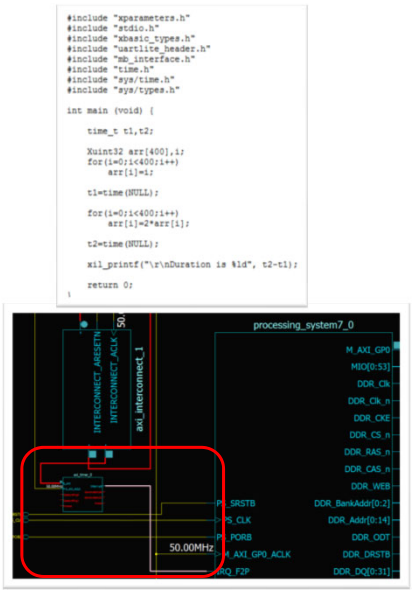
The key to embedded systems design is gaining control of the interplay between computation and […] constraints to meet a given set of requirements on a given implementation platform.
Thomas A. Henzinger, Joseph Sifakis. **The Discipline of Embedded Systems Design,** *Computer, October 2007, pp. 32-40.*



14

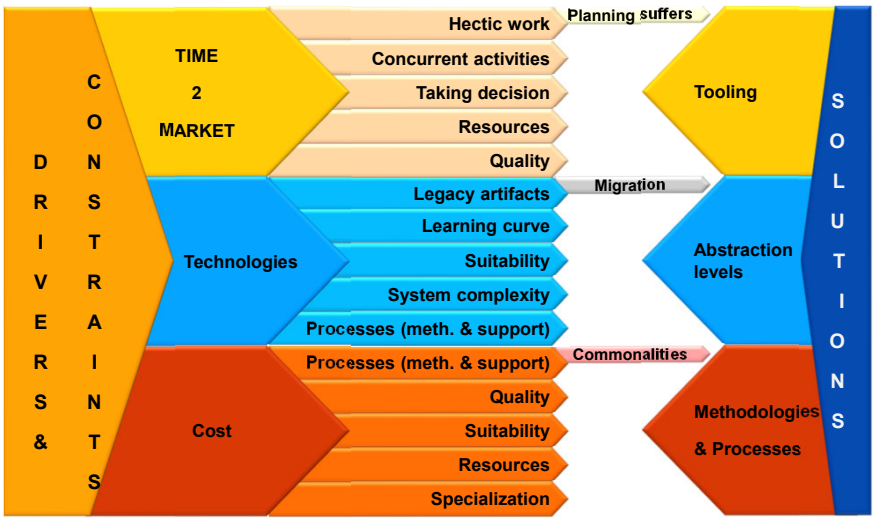
Selection?





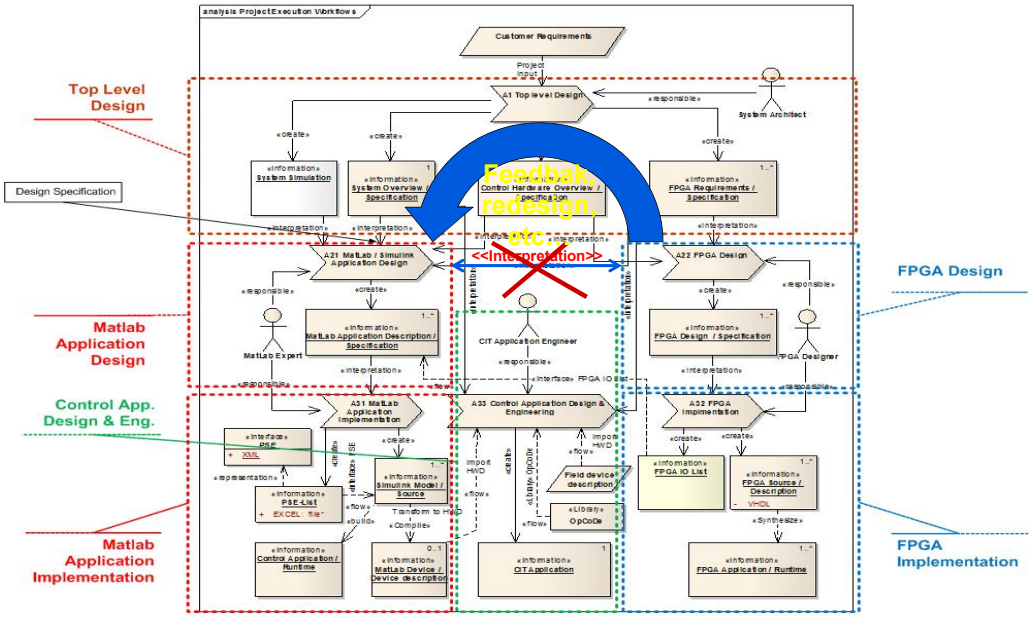
15

High level, un-specific pressure factors (non-exhaustive)



16

Design process



17

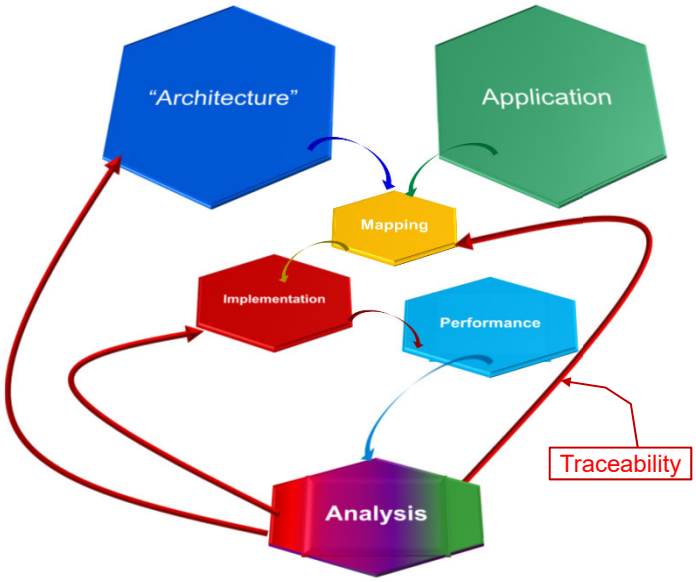
Y-Chart approach (and design refinement)

Functionality
Timing
Relations
…

Processor
Bus(es)
Accelerators
Memory
…

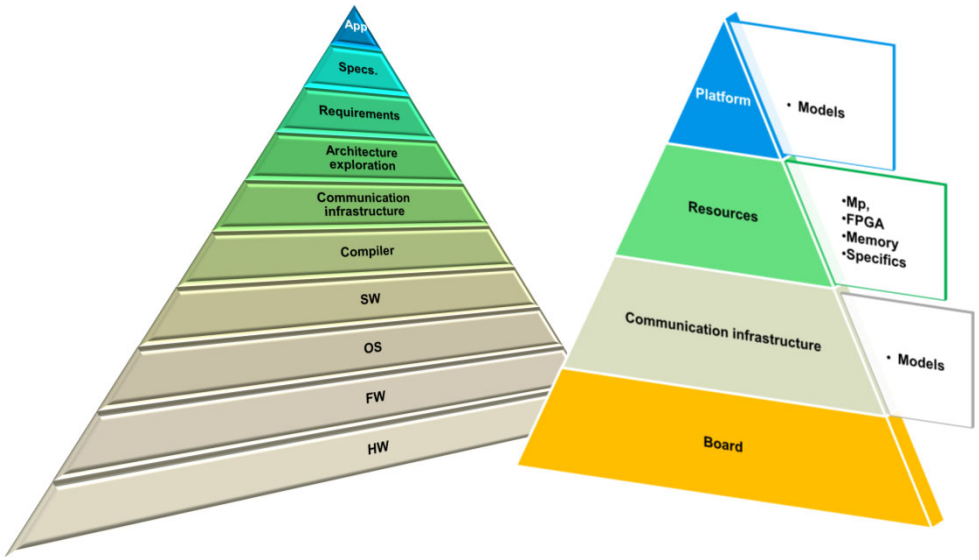
HW
SW
On-chip
On-board
Distributed
…

Throughput
Power
Scheduling
Operational
…



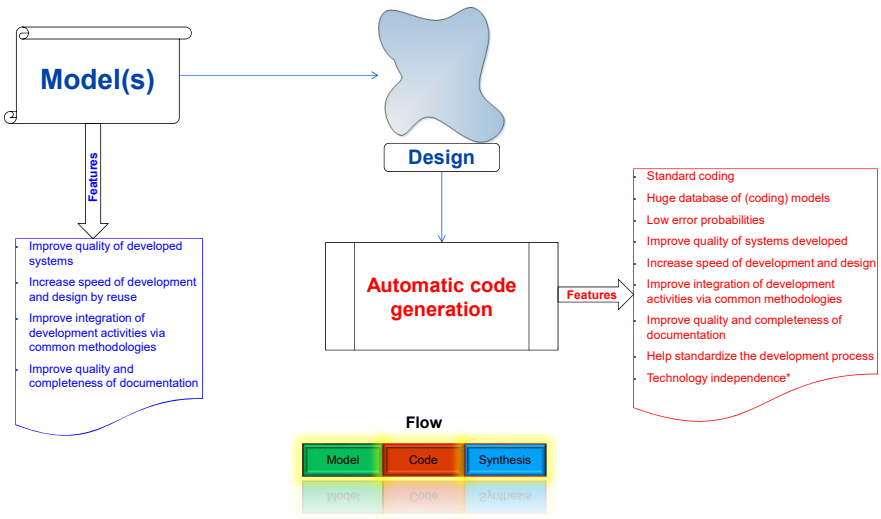
18

Application ←→Platform mapping



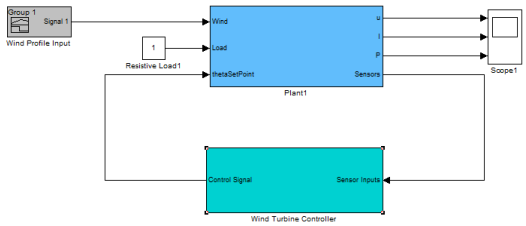
19

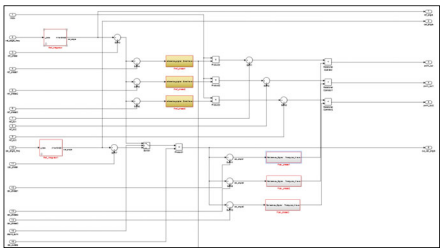
One way to go - ACG

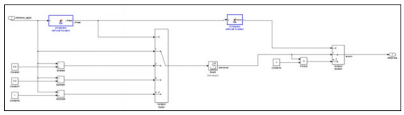


20

A small example – a wind turbine controller
What is the flow ?







A small example

Contents

Summary

Clock Summary

Resource Utilization Report

High-level Resource Report

Target-specific Report

Optimization Report

Distributed Pipelining

Streaming and Sharing

Target Code Generation

Traceability Report

```
50  SIGNAL selector                 : std_logic_vector(7 DOWNTO 0); -- ufix8
51  SIGNAL selector_signed          : signed(7 DOWNTO 0); -- int8
55  SIGNAL Subtract_out1            : signed(15 DOWNTO 0); -- int16
```

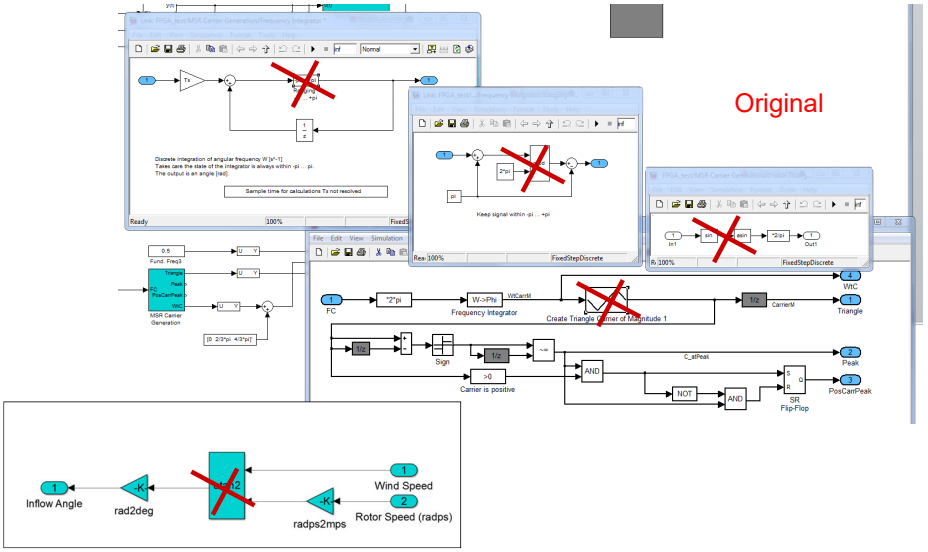
No

```
64  BEGIN
65    -- <S2>/Embedded MATLAB Function
66    u_Embedded_MATLAB_Function : Embedded_MATLAB_Function
67      PORT MAP( u => Reference_Signal,  -- int16
68                yPhase => selector  -- int8
69                ); 71    selector_signed <= signed(selector);
72
73    Reference_Signal_signed <= signed(Reference_Signal);
74
75    -- <S2>/Constant
76    Constant_out1 <= to_unsigned(16384, 16);
77
78    -- <S2>/Subtract
79    Subtract_sub_cast <= signed(resize(Constant_out1(15 DOWNTO 14), 16));
80    Subtract_out1 <= Subtract_sub_cast - Reference_Signal_signed;
81
82    -- <S2>/Constant1
83    Constant1_out1 <= to_unsigned(32768, 16);85     -- <S2>/Subtract1
86    Subtract1_sub_cast <= signed(resize(Constant1_out1(15 DOWNTO 14), 16));
87    Subtract1_out1 <= Reference_Signal_signed - Subtract1_sub_cast;
88
89    -- <S2>/Multiport Switch
90
91    afterMux <= Reference_Signal_signed WHEN selector_signed = 1 ELSE
92      Subtract_out1 WHEN selector_signed = 2 ELSE
93      Subtract1_out1;
94
95    -- <S2>/Even Triangle LUT
96    Even_Triangle_LUT_sub_temp <= afterMux - 1;
97
98    Even_Triangle_LUT_k <= to_unsigned(0, 4) WHEN afterMux <= 1 ELSE
99      to_unsigned(15, 4) WHEN afterMux >= 16 ELSE
100     unsigned(Even_Triangle_LUT_sub_temp(3 DOWNTO 0));
101   solution <= nc(to_integer(Even_Triangle_LUT_k));
102
103   Triangular_Wave <= std_logic_vector(solution);
104
105 END rtl;
```

Traceability links

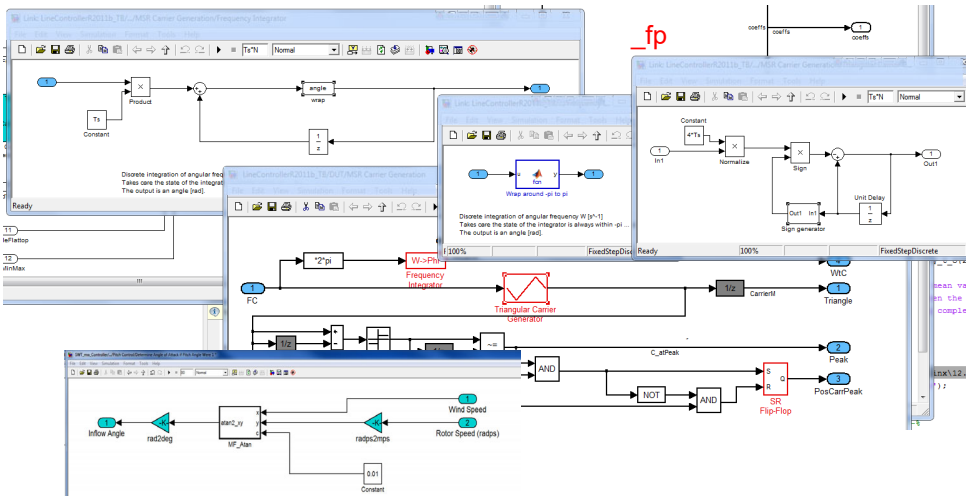
THAT'S SIMPLE ? ENOUGH ?

Compliance - original



23

Compliance - changes



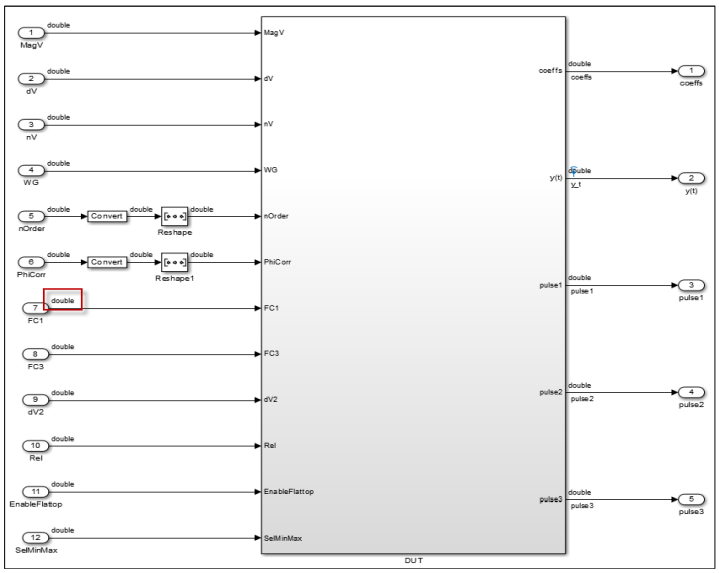
24

Flow

25

Consider a HW design here…

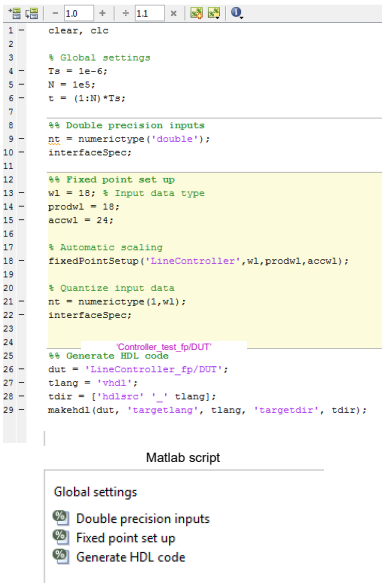
What is wrong?



26

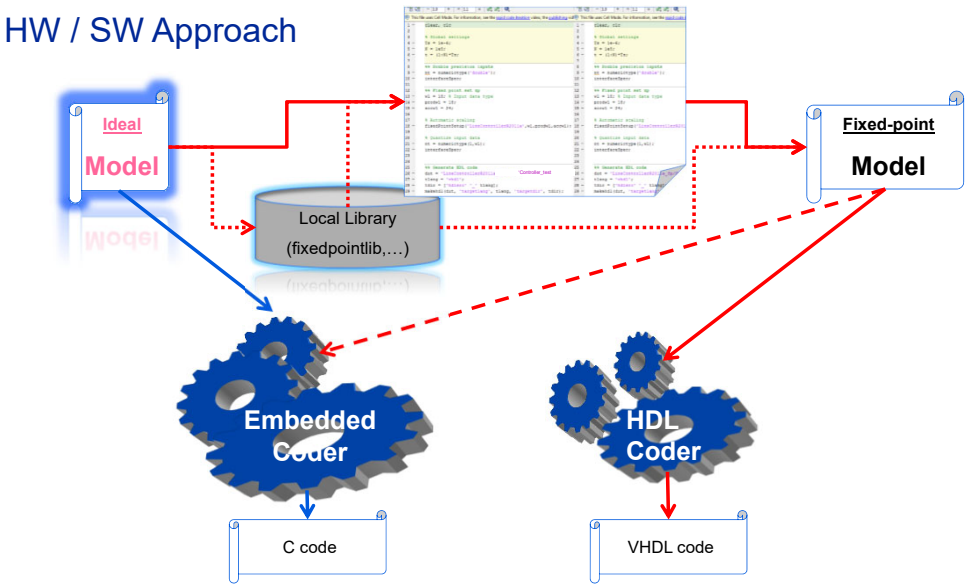
Demo Components

- **Controller_TestBench.m**
  - Defines the input state (types and values)
  - Proposes a word length and a fraction point position
  - fixedPointSetup.m
    - Simulates the model
    - Provides ”optimal” value for the fraction point
    - Aplies this to all elements requiring it (product, sum, etc)
  - Generating HDL code. Can be automated, too.
  - Input: ”file.mdl”
  - Output: ”file_fp.mdl”



27

HW / SW Approach

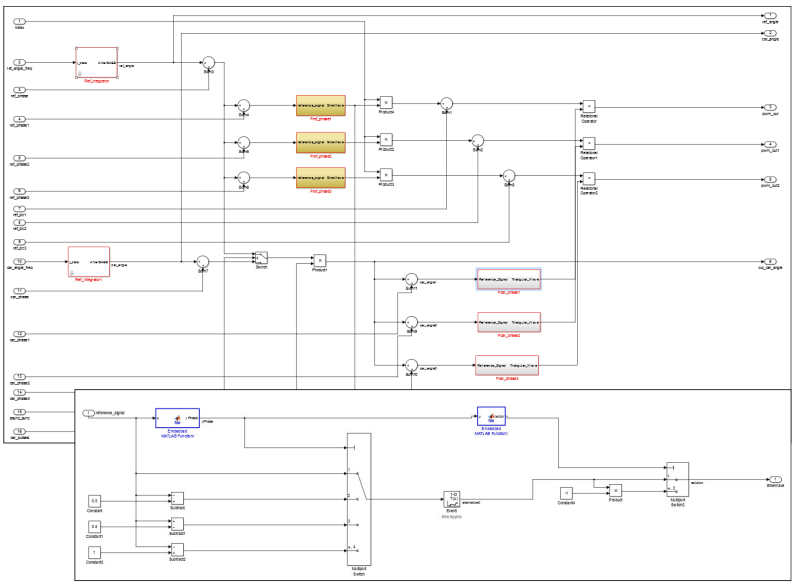


28

Flow

29

A small example



30

Results

Device: xc6slx150t-3fgg484				
Feature	**Existing design (VHDL)**	**HDL-C: Sol.1**		**Comments**
Number of Slice Flip Flops	320 (0%)	422 (1%)		
Number of Slice LUTs	478 (0%)	1158 (1%)		
Number of RAMB8/16BWERs	2(1%)			
Number of DSP48A1s (multipliers)	2 (1%)	21 (11%)		
Frequency (MHz)	64,3	17,8		40/80MHz required !

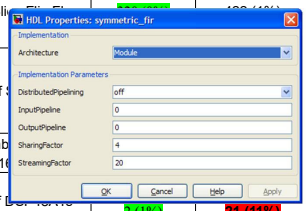
31

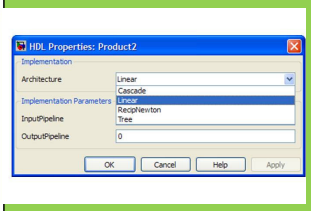
Flow

32

Results

Device: xc6slx150t-3fgg484				
Feature	**Existing design (VHDL)**	**HDL-C: Sol.1**		**Comments**
Number of Sli				
Number of				
Numb RAMB8/1				
Number of (multipliers)	2 (1%)	21 (11%)		
Frequency (MHz)	64.3	17.8		40/80MHz required !





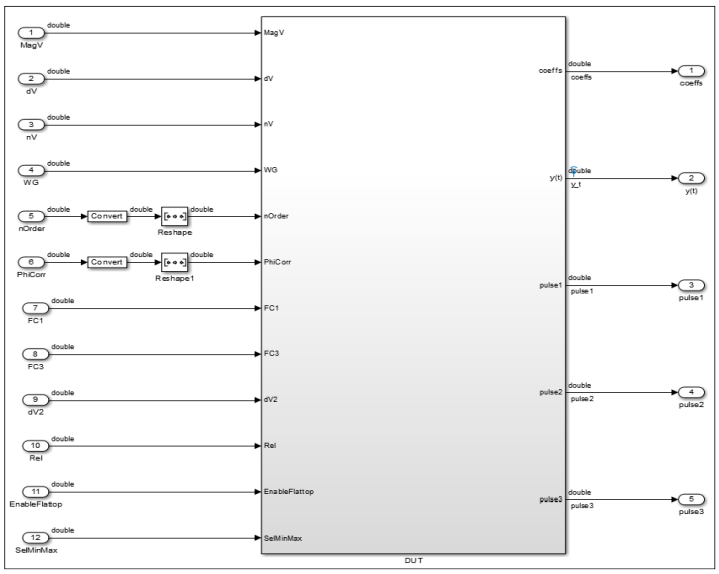
33

Results

Device: xc6slx150t-3fgg484					
Feature	**Existing design (VHDL)**	**HDL-C: Sol.1**	**HDL-C: Sol. 3**		**Comments**
Number of Slice Flip Flops	320 (0%)	422 (1%)	635 (1%)		
Number of Slice LUTs	478 (0%)	1158 (1%)	1024 (1%)		
Number of RAMB8/16BWERs			2(1%)		
Number of DSP48A1s (multipliers)	2 (1%)	21 (11%)	13(7%)		
Frequency (MHz)	64.3	17.8	34,4		40/80MHz required !

34

Coupling / Legacy



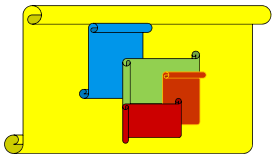
35

35

Wrapper and code

```
entity fpga_top is
        port(
                -- global signals ------------------------------------------------------
                i_reset             : in std_logic;     -- reset
                i_clk               : in std_logic;     -- clock
                i_clk_en            : in std_logic;     -- clock enable
--              o_ce_out            : out std_logic;    --
                o_Blk               : out   std_logic; --
                -- input interface -----------------------------------------------------
                i_input_dpram_waddr  : in std_logic_vector(4 downto 0);  --
                i_input_dpram_data : in std_logic_vector(31 downto 0);   --
                i_input_dpram_we   : in std_logic;                       --
                -- output interface ----------------------------------------------------
                i_output_dpram_raddr : in std_logic_vector(4 downto 0); --
                o_output_dpram_q     : out std_logic_vector(31 downto 0) --
                );
end entity fpga_top;
```

```
Controller_Instance: RC
  port map(
     clk                => i_clk,
     reset              => i_reset,
     clk_enable         => i_clk_en,
       index            => input_reg(0)(15 downto 0),
--       ce_out           => output_reg(0)(0),
       ref_angle_freq   => input_reg(1),
.....................................................................
  );
```



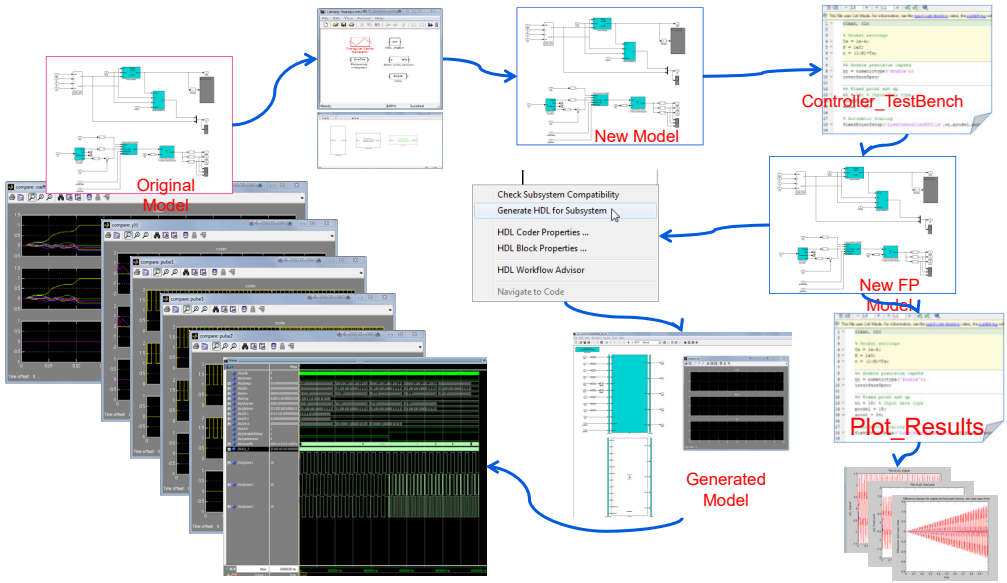
36

36

Flow

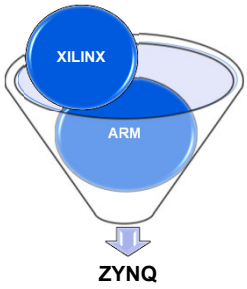
37

Recap Flow



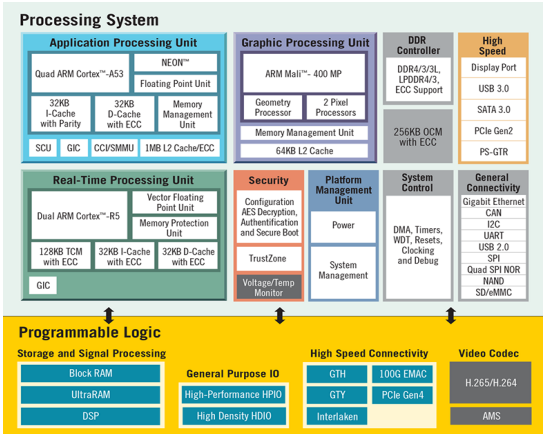
38

Technology: Multiprocessing



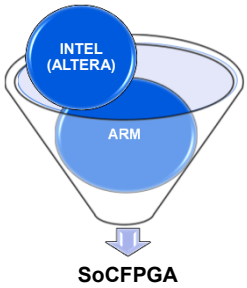
ZYNQ

XILINX® ARM®



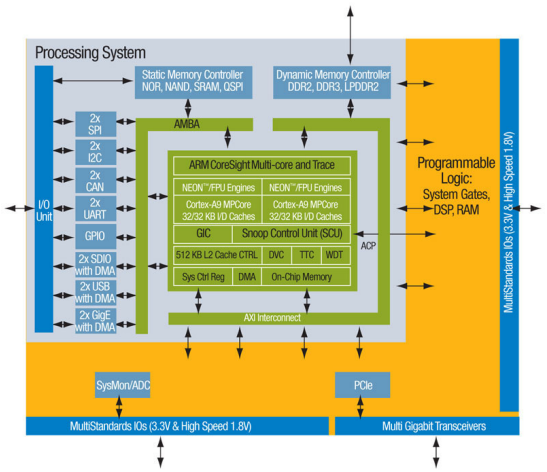
39

Technology: Multiprocessing



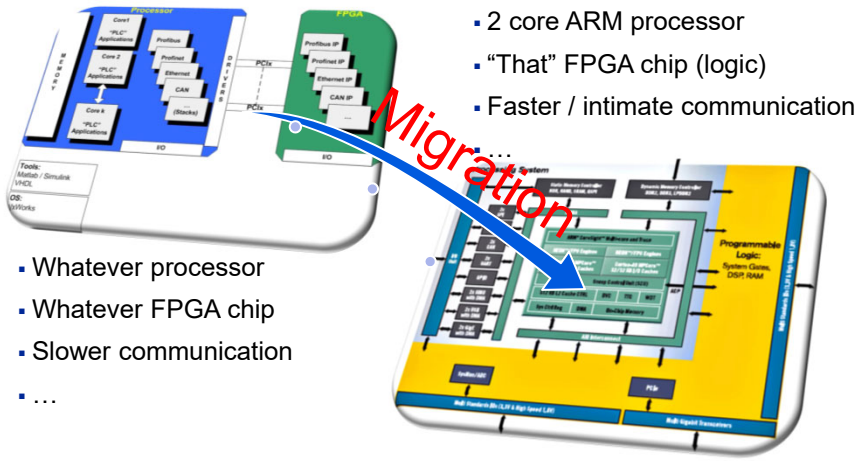
SoCFPGA

ALTERA® ARM®



40

On-Chip
Platform, Multiprocessing, Migration

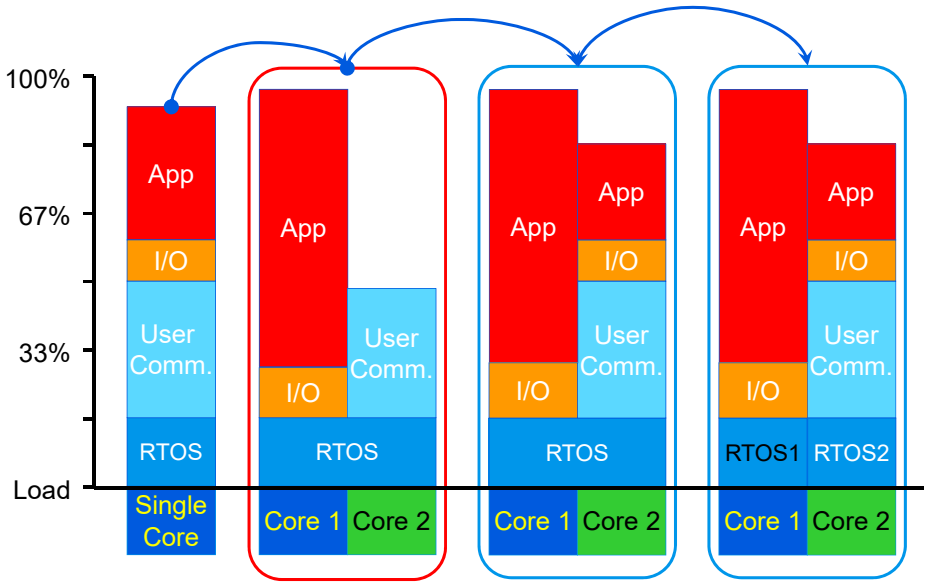


- 2 core ARM processor
- "That" FPGA chip (logic)
- Faster / intimate communication
- …

- Whatever processor
- Whatever FPGA chip
- Slower communication
- …

41

Dual Core Usage – A Staged Approach

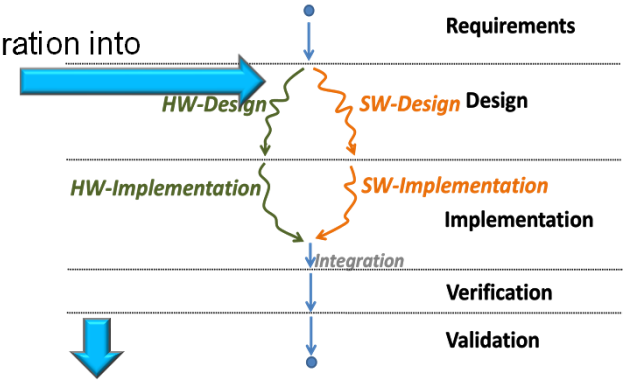


42

HW/SW Partitioning

Gaetana Sapienza – "HW/SW Partitioning methodology for embedded applications using multiple criteria decision analysis", PhD thesis, MDH nr. 210, Nov. 2016.

Early start of design separation into
hardware design flow
and software design flow



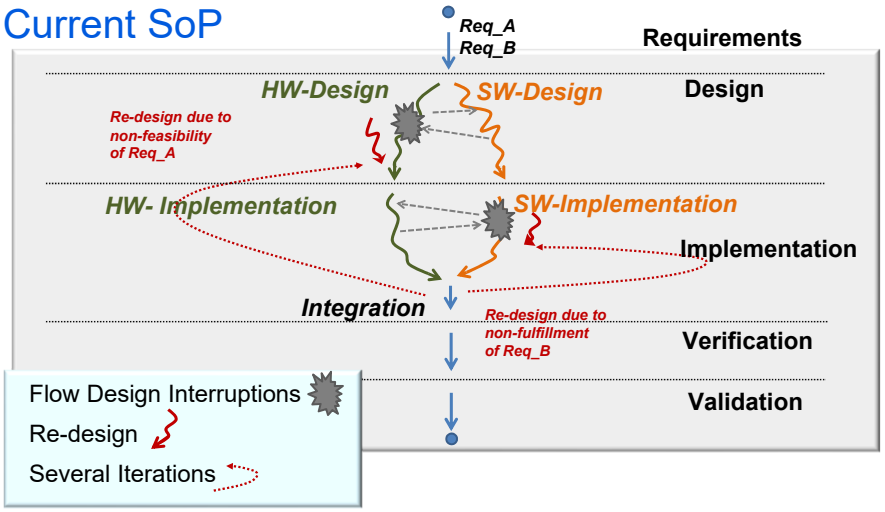
Underestimation of the partitioning decision process

(Unmature and hasteful decisions, shadowed by model-based tool trend, unaccurate estimation of project and application constraints)

43

Simplified Typical Development Process

Current SoP

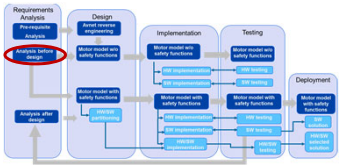


44

Requirements Elicitation

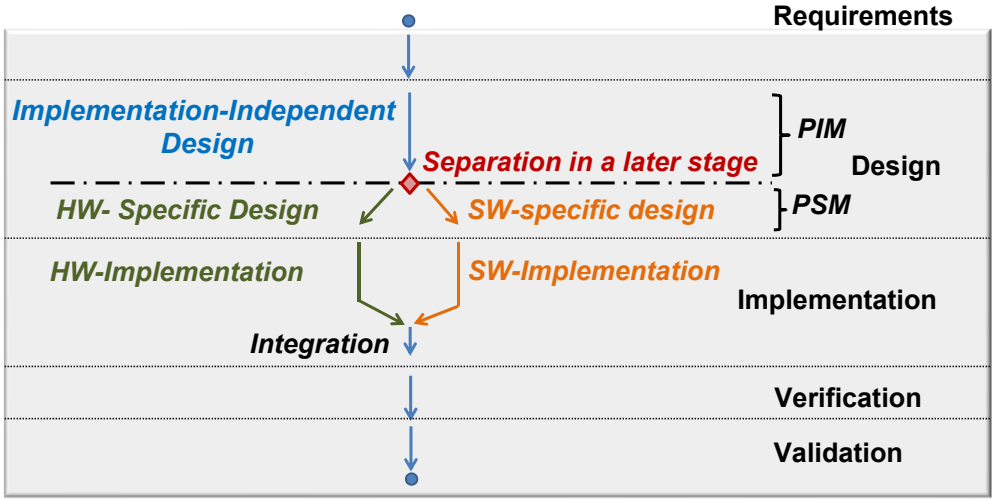
List of key requirements and project development constraints

- The maximum allowed execution time shall not exceed 30 microsec
- The power consumption shall not exceed 1W
- The design phase shall be completed within 3 months
- The entire project shall be completed within 6 months
- The application shall provide a Safety Torque Off (STO) function
- The application shall provide a Safety Limited Speed (SLS) function
- The overall system development cost shall not exceed the cost of XXX
- The application shall be developed by using a component-based design approach
- The application shall be developed by using a model-based design approach
- The application shall be developed by using Matlab and Simulink for design and implementation
- The application shall be developed on an heterogeneous platform (Xilinx Zynq ZC702)



45

A Possible Approach ?

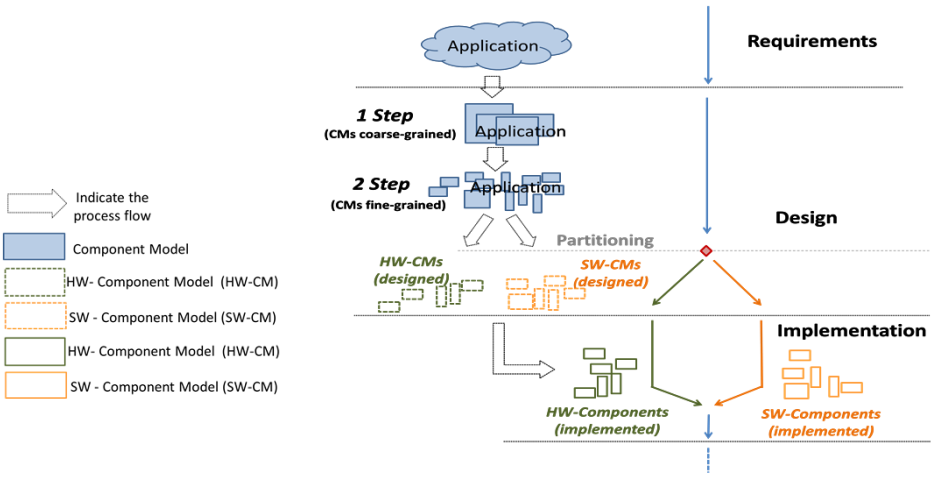


PIM (Platform-Independent Model)
PSM (Platform-Specific Model)

46

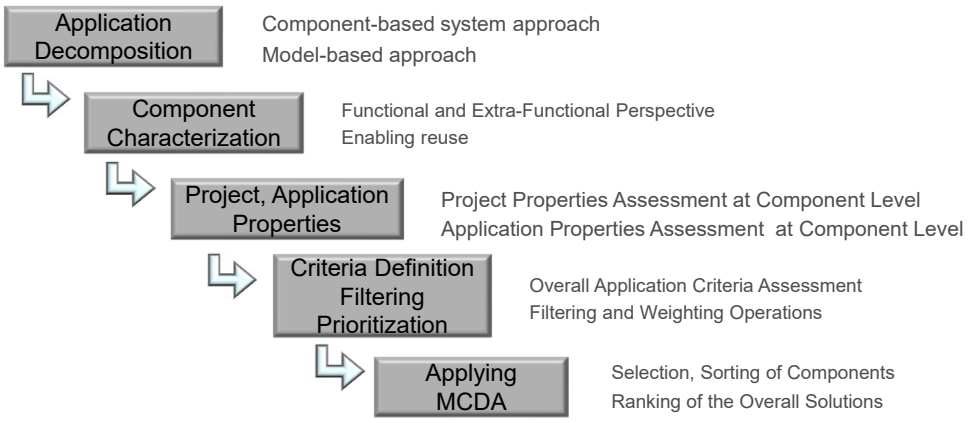
Application Decomposition

1st - STEP of the partitioning decision process



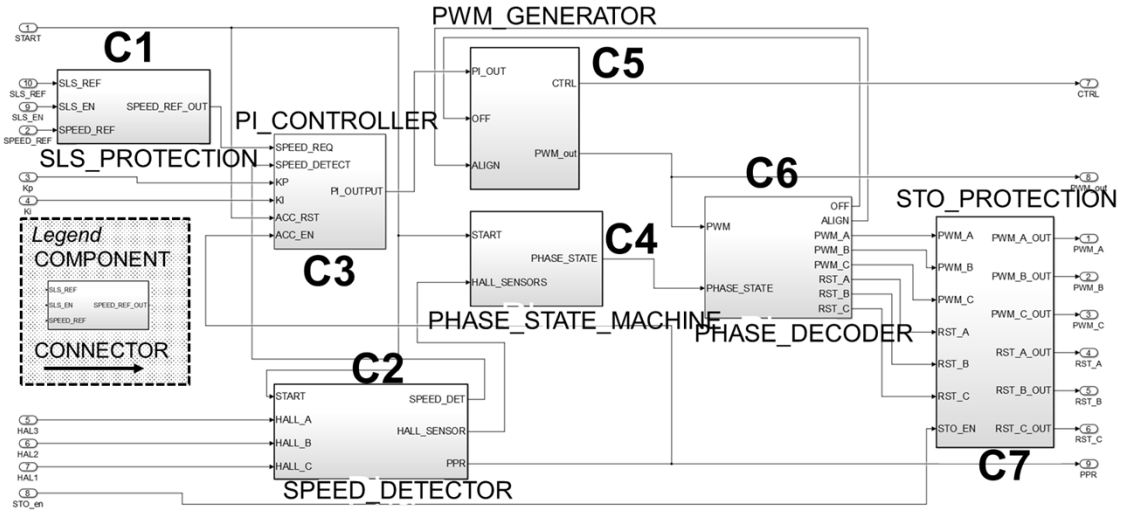
47

Overall Partitioning Decision Process Overview
5 – Steps Process



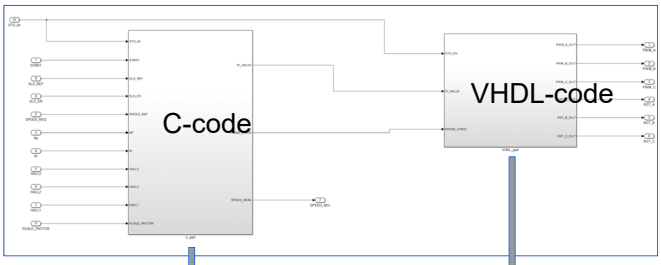
48

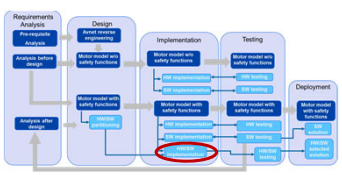
Design refinement for final deployment – BLDC Motor



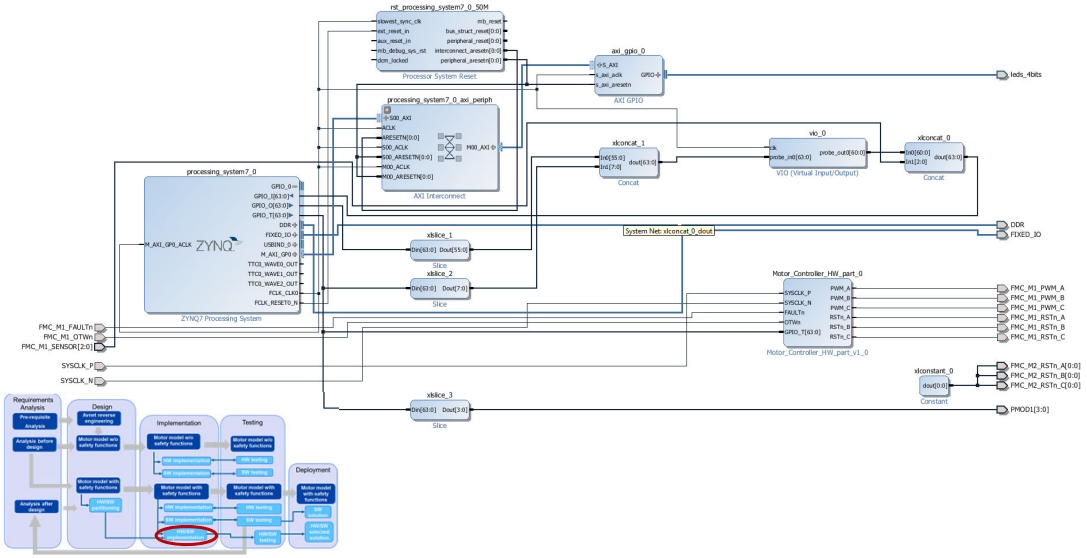
HW/SW Implementation

The partitioning of the application was carried out based on the input from MultiPar





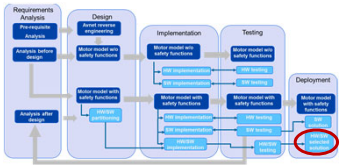
HW/SW Implementation



51

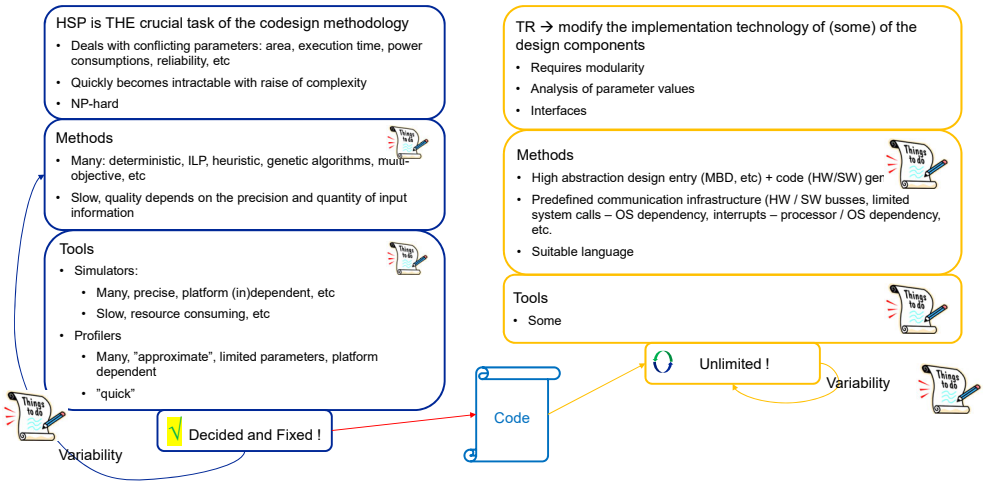
HW/SW Deployment

- The selected HW/SW deployment configuration implemented on Zynq platform
- The motor properly spinning
- Safety functions also implemented
- Requirements and constraints satisfied



52

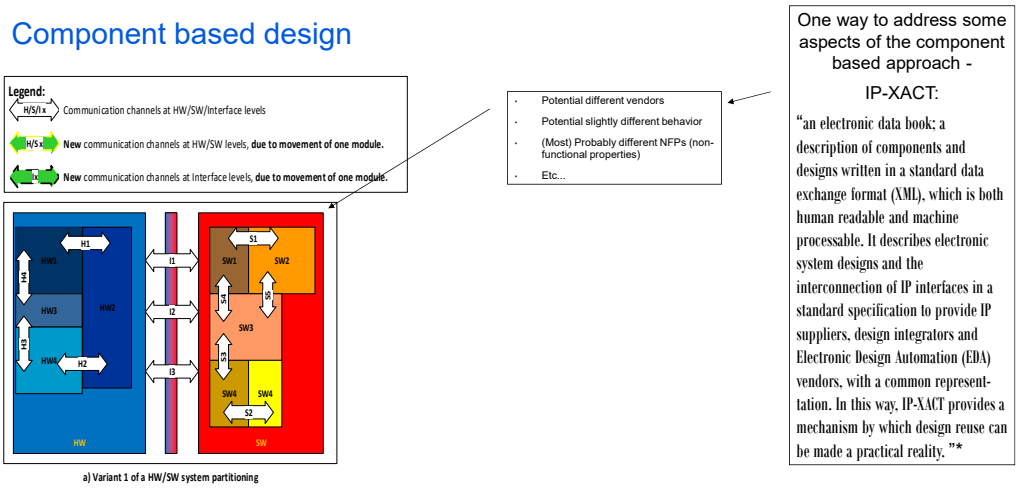
HW / SW Partitioning (HSP) + Technology Re-targeting (TR)



53

Technology independence aspects

Component based design



a) Variant 1 of a HW/SW system partitioning

- Potential different vendors
- Potential slightly different behavior
- (Most) Probably different NFPs (non-functional properties)
- Etc...

One way to address some aspects of the component based approach -

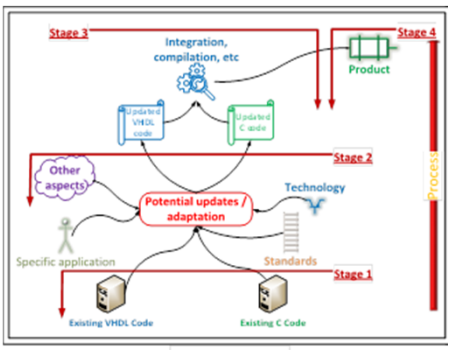
IP-XACT:

"an electronic data book; a description of components and designs written in a standard data exchange format (XML), which is both human readable and machine processable. It describes electronic system designs and the interconnection of IP interfaces in a standard specification to provide IP suppliers, design integrators and Electronic Design Automation (EDA) vendors, with a common representation. In this way, IP-XACT provides a mechanism by which design reuse can be made a practical reality. "*

* *Marc van Hintum, NXP Semiconductors, Paul Williams, Mentor Graphics Consulting.* The Value of High Quality IP-XACT XML. Design & Reuse

54

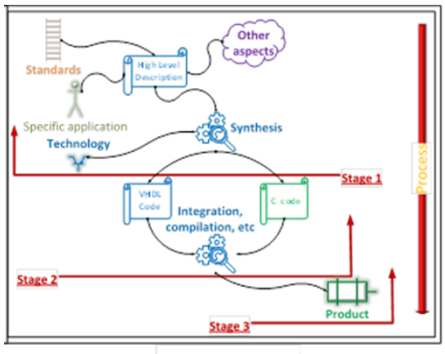
Flows. Legacy & New Designs



a) The first solution path

55

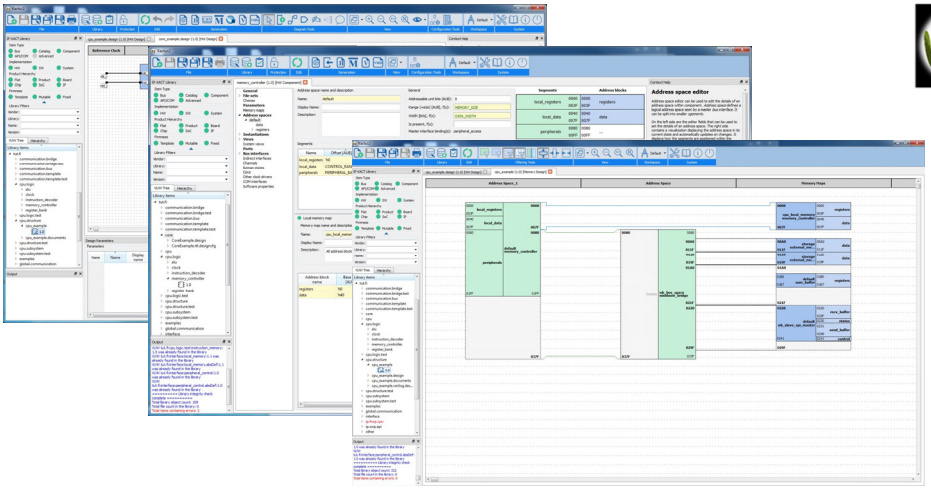
Flows. Legacy & New Designs



b) The second solution path

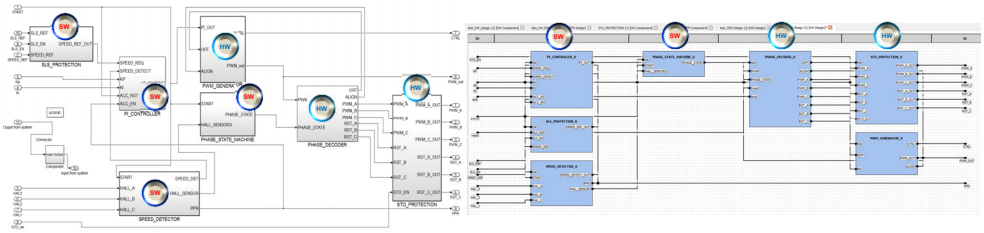
56

Kactus2: an IP-XACT tool



57

Simulink vs. Kactus2

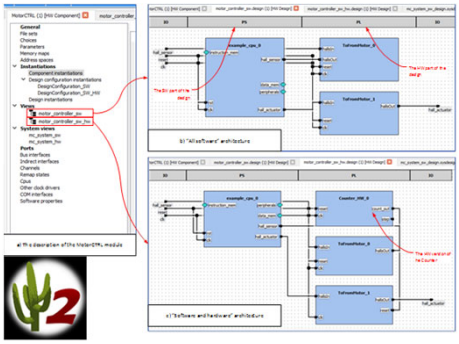


- Language
- Specification & Design
- Selection of platform(s)
- ACG (“HLS” ?)
- Integrated environments
- Legacy ?
- Versioning ?
- ...

- Include CPU ?
- “Place” SW on CPU ?
- Connect SW to HW ?
- How to build the separate projects ?
- Keep track of various potential versions of the SW / HW ?
- “Link” to Xilinx / Altera / etc. tools ?
- “HLS” potential ?
- …

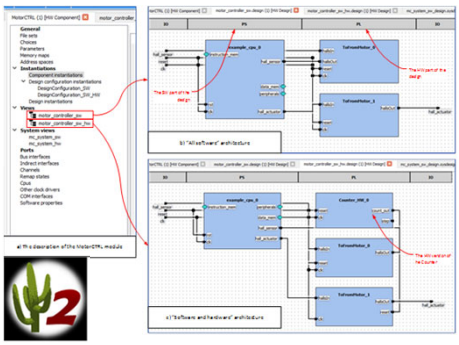
58

BLDC Motor: Kactus2



59

Kactus2 vs. VSI



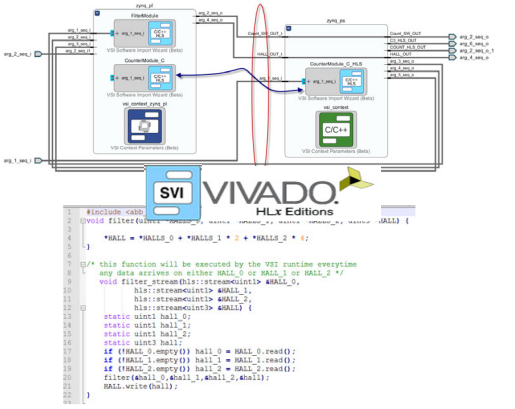
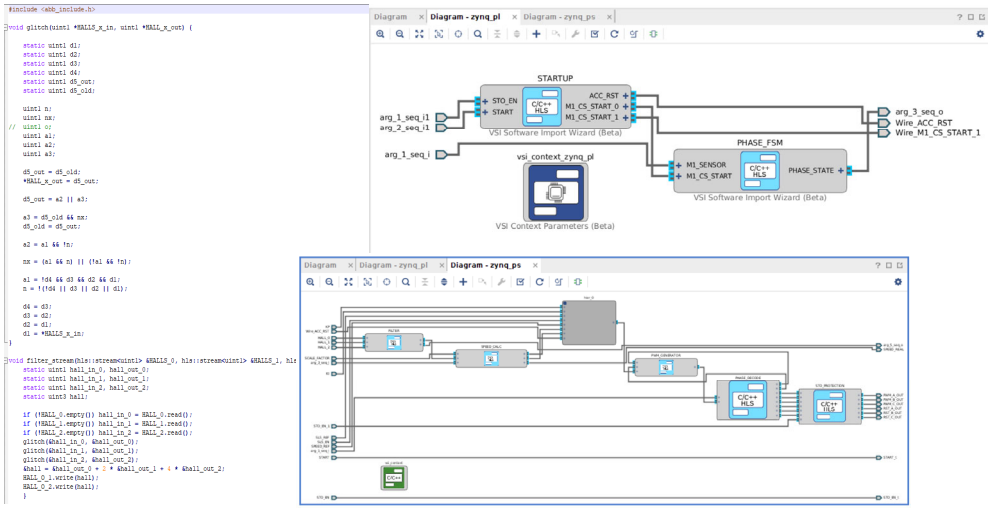


Figure 9. *FilterModule* description in HLS C.

SCONS

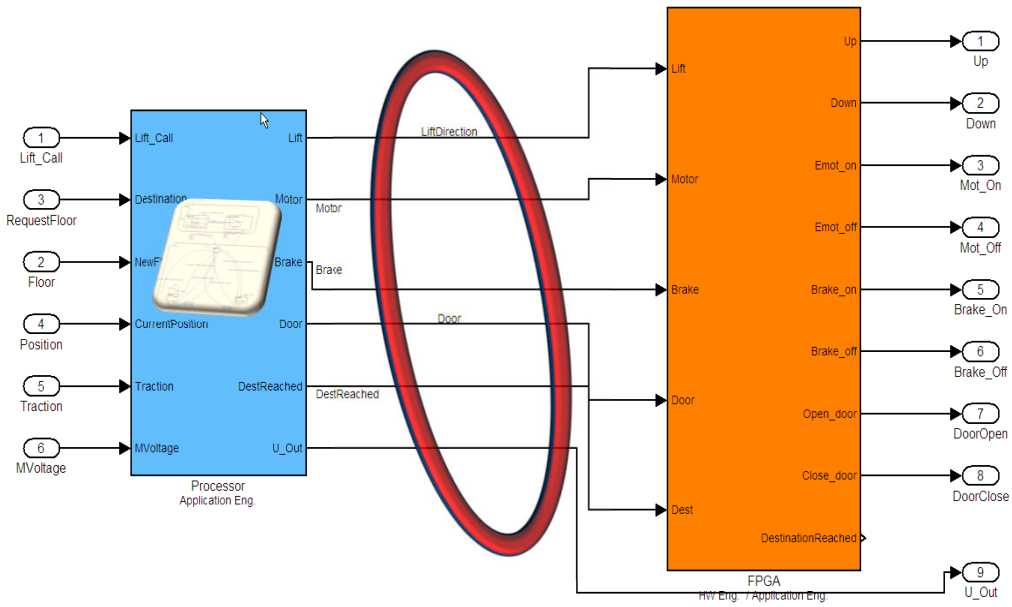
60

BLDC Motor: VSI



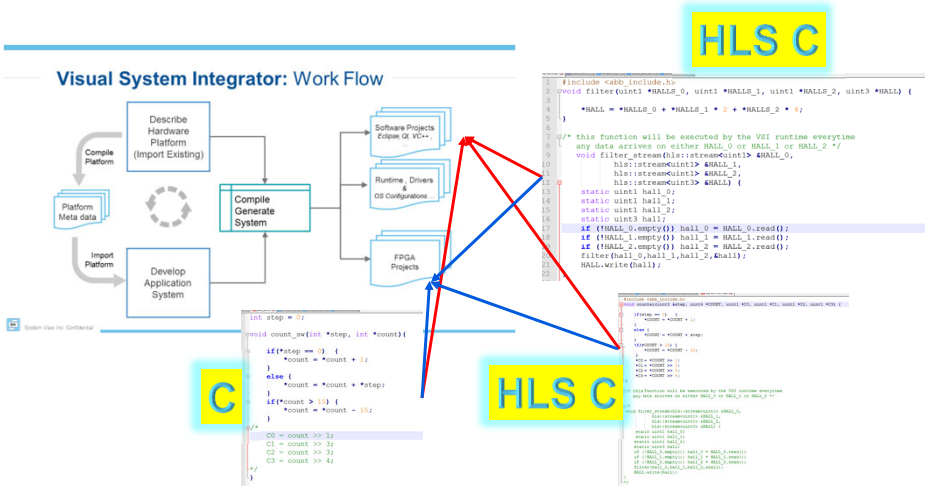
61

Desired Features / Capabilities



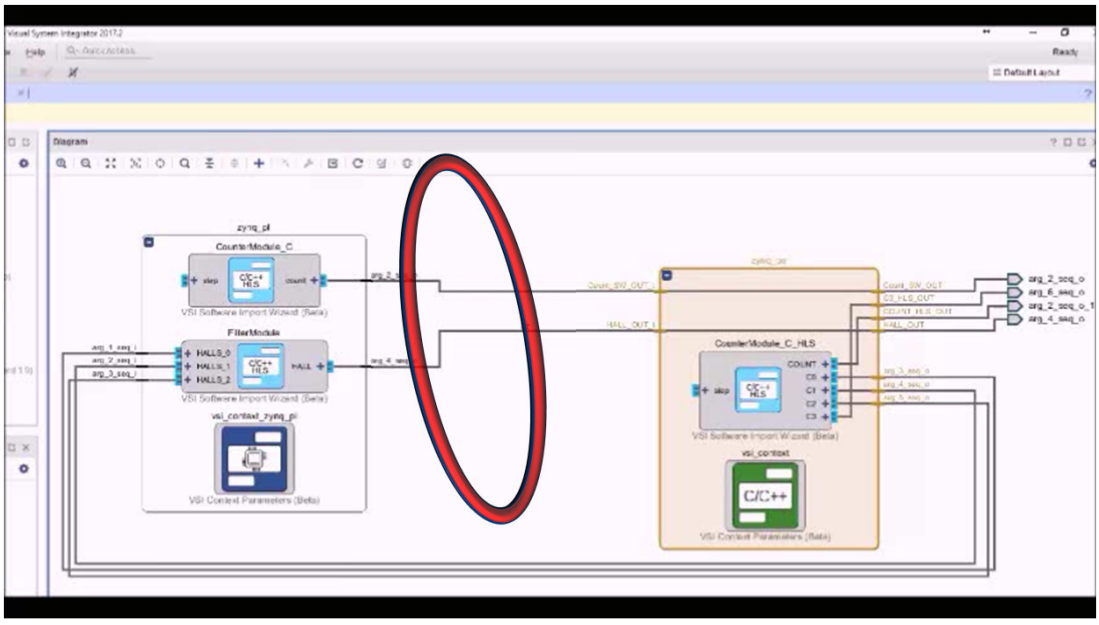
62

Dealing with interfaces - Visual System Integrator



63

Dealing with interfaces - Visual System Integrator



64

Open for

Questions

Thank you !

Recent developments*

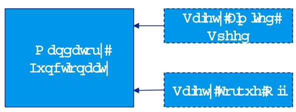
J oredo#uhdfkdeldw|#srvhv#qhz#fkdoohqjhv#lq#wkh#
p rghuq#hp ehgghg#v|vwhp v#+HV,#ghyhorsp hqw=

- G liihuhqw#vwdqgdugv
- Ydu|lqj#rshudwlqj#frqglwlrqv
- G liihuhqw#fxvwrp hu#qhhgv

Wkhvh#fkdoohqjhv#lqwurgxfh#wkh#**variability** suroeohp =

- Vriwz duh#Hqjlqhhulqj#Vroxwlrq#0Vriwz duh#Surgxfw#Dlqhv#+VSO,
- Exw#VSO#lv#d#vriwz duh0rulhqwhg#whfkqltxh

Ydubdelolw|#lq#wkh#frqvlghuhg#surmhfw



Features=

- Vdihw|#Dlp lwhg#Vshhg#+VOV,
- Vdihw|#Wrutxh#R ii#+VWR,

Lp sohp hqwdwlrq#ri#wkh#VOV#dqg#VWR#uhsuhvhqw#**variants**#+#
z kloh#VOV#dqg#VWR#lwhd#uhsuhvhqw#**variation points** +YS,

*A. Urvantsev, M. E. Johansson, N. Müllner,T. Seceleanu. *Experiencing Technology Independence.* IEESD, COMPSAC 2019

Problem formulation

Sursrvhg#ghvljq#iorz=

- Whfkqrorj|#lqghshqghqfh
- Yduldelolw|
- KVS
- Ghshqghqf|#Uhvroylqj
- Lqwhuidflqj#Sureohp

Uhvhdufk#txhvwlrqv=

- **RQ1**=#Krz#wr#hqdeoh#d#whfkqrorj|#lqghshqghqw#ghvljq#iorz#lq#wkh#hduo|#vwdjh#ri#wkh#ghvljq#surfhvv#dqg#shuirup#wkh#sduwlwlrqlqj#ghflvlrq#surfhvv#lq#d#odwhu#vwdjhB
- **RQ2**=#Krz#wr#hqdeoh#d#v|vwhpdwlf#dqg#hiihfwlyh#surfhvv#vxssruwlqj#ghvljqhuv#ehiruh#dssolfdwlrq#sduwlwlrqlqj#dqg#surylgh#whfkqrorj|#uh0wdujhwlqjB

67

Limitations

Zh#lqwurgxfh#d#whfkqrorj|#lqghshqghqw#ghvljq#iorz#iru=

- Uhvroylqj#ghshqghqflhv
- Elqglqj#yduldqwv#wr#h{lvwlqj#LSv
- Uhsodflqj#yduldqwv#iru#h{lvwlqj#LSv
- Kdqgolqj#wkh#KVS#surfhvv

Phwkrgrorj|#uhodwhg#olplwdwlrqv=

- Uhvxowv#duh#gliilfxow#wr#jhqhudol}h
- Wkh#vwxg|#fryhuv#d#olplwhg#qxpehu#ri#wrrov#dqg#whfkqltxhv#dssolhg#wr#wkh#jlyhq#surmhfw

Lpsohphqwdwlrq#uhodwhg#olplwdwlrqv=

- Xvhg#wrrov#ghshqgv#rq#[lolq{*v#surgxfwv/#zklfk#lqwurgxfhv#d#yhqgru0ghshqghqf|/#exw…
- [lolq{#fryhuv#84(#ri#wkh#pdunhw

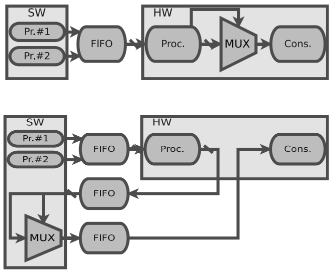
68

KVS#dqg#Lqwhuidflqj

Kdugzduh0Vriwzduh#Sduwlwlrqlqj#+KVS,=

- QS0kdug#sureohp
- Vroyhg#e|#khxulvwlf/#jhqhwlf#dqg#foxvwhulqj#dojrulwkpv
- Uhtxluhv#lqwhjudwlrq#surfhvv
- Lqwurgxfhv#**interfacing** sureohpv

Lqwhuidflqj#sureohp =



Q sruw#zrxog#uhtxluh#5Q fkdqjhv

Ylydgr dqg#YVL

Ylydgr Ghvljq#Vxlwh#+KO{=

- Vriwzduh0ghilqhg#LS#fruh#jhqhudwlrq#e|#**KOV0F**
- Eorfn0edvhg#LS0lqwhjudwlrq
- Prgho0edvhg#ghvljq#lqwhjudwlrq

Ylvxdo#V|vwhp#Lqwhjudwru#+YVL,=

- Iurqw0hqg#iru#Ylydgr
- Vroyhv#lqwhuidflqj#sureohp#gxulqj#fkdqjlqj#ri#wkh#h{hfxwlrq#frqwh{w
- Vsdwldo#yhlz#rq#d#jlyhq#v|vwhp#lq#wzr#sodqv=#V|vwhp#dqg#Sodwirup

KOV

KOV0F=

- Dxwrqrp rxv#wudqvodwlrq#ri#d#F 0edvhg#ghvfulswlrq# lqwr#gljlwdo#kdugz duh
- Khosv#wr#dfklhyh#wkh#ghvlung#sodwirup 0 lqghshqghqfh
- J hqhudwhg#kdugz duh#ghvfulswlrq#lq#p rvw#fdvhv#lv# ixqfwlrqdo|#fruuhfw#exw¤
- ¤ #dxwrqrp rxv#wudqvodwlrq#diihfw#wkh#shuirup dqfh

KOV0F#h{dp soh=

```
void multiplier(uint21 *SPEED_IN,
                uint21 *SCALE_FACTOR,
                uint21 *MULT_OUT){
    *MULT_OUT = *SPEED_IN * *SCALE_FACTOR;
}

void stream(hls::stream<uint21> &SPEED_IN,
            hls::stream<uint21> &SCALE_FACTOR,
            hls::stream<uint21> &MULT_OUT){
      static uint21 speed_in;
      static uint21 scale_factor;
      static uint21 mult_out;

      if (!SPEED_IN.empty())
            speed_in = SPEED_IN.read();

      if (!SCALE_FACTOR.empty())
                scale_factor =
      SCALE_FACTOR.read();

      multiplier(&speed_in, &scale_factor,
      &mult_out);
      MULT_OUT.write(mult_out);
}
```

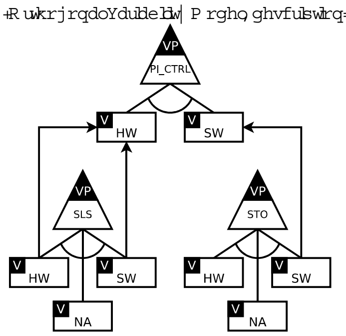
71

•F odvvlfdo¶#R YP

Ghshqghqf|#h{dp soh=

- Li#VWR lv#h{hfxwhg#lq#VZ wkhq SLbFRQWUROOHU vkrxog#eh#h{hfxwhg#lq#VZ
- Li#VOV lv#lqfoxghg lqwr#wkh#surmhfw#wkhq SLbFRQWUROOHU vkrxog#eh#h{hfxwhg#lq#KZ

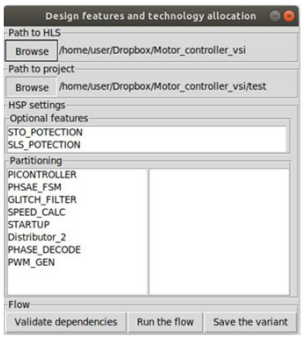
R YP +R uwkrjrqdoYduldelolw| P rgho,ghvfulswlrq=



72

Ghyhorshg#wrro

JXI=



Frqilj xudwlrq#ledn#{dp sdn=

```
STO_PROTECTION
    context
    dependencies
        PI_CONTROLLER
    rules
    If STO_PROTECTION.context == 'SW'
        PI_CONTROLLER.context == 'SW'


SLS_PROTECTION
    context
    dependencies
        PI_CONTROLLER
    rules
    If SLS_PROTECTION.status == True
        PI_CONTROLLER.context == 'HW'
```

73

Open for

Questions

Thank you !

74