MATLAB EXPO 2019

Du modèle au materiel :
Solutions temps-réel pour prototypage rapide

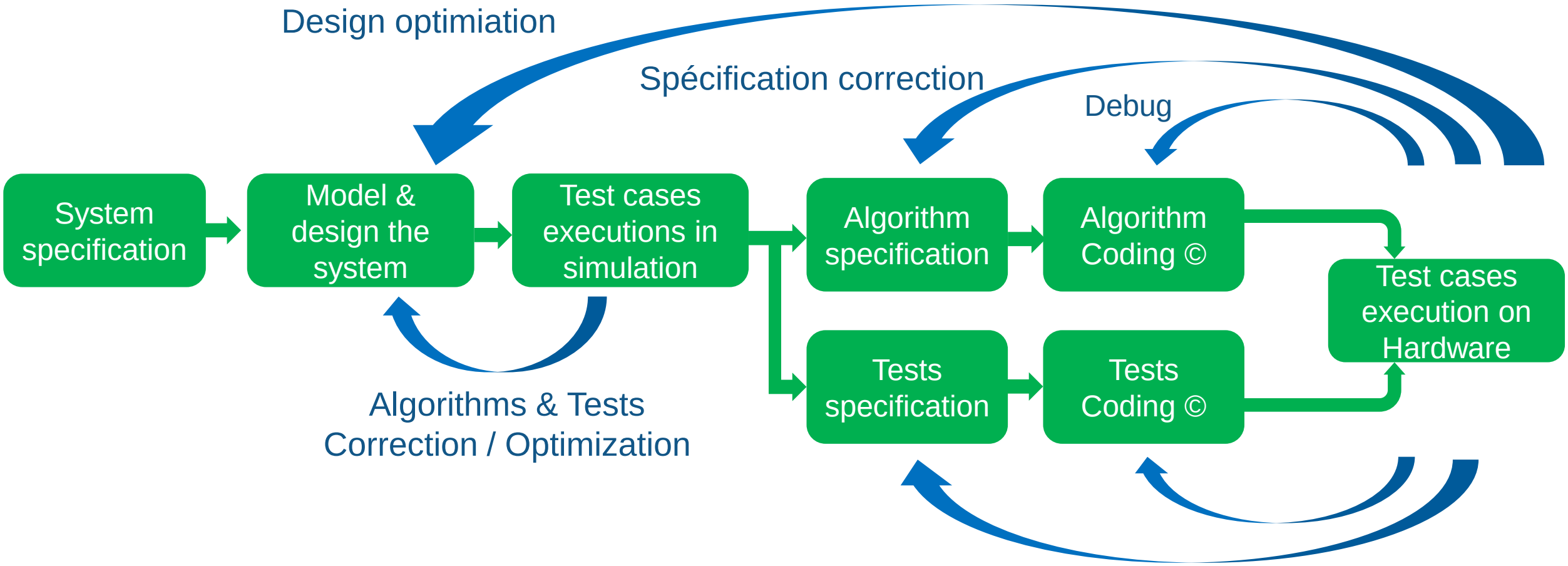
Olivier Berard

Agenda

1. **Why**, **What**, **How** Hardware deployment?
2. **Use cases** details
3. **Case study**
4. **Conclusions** & what's next…

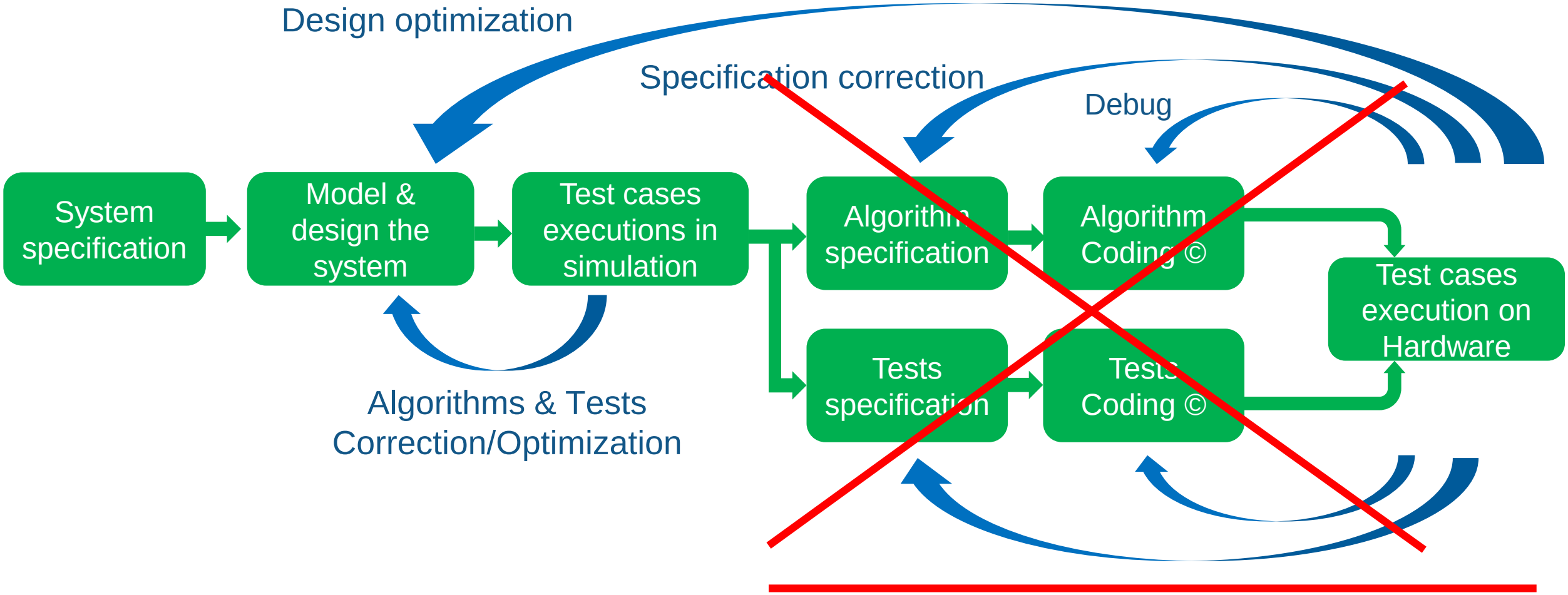
Why deploying model to Hardware?

Manual Hardware deployment



Why deploying model to Hardware?

Automatic Hardware deployment

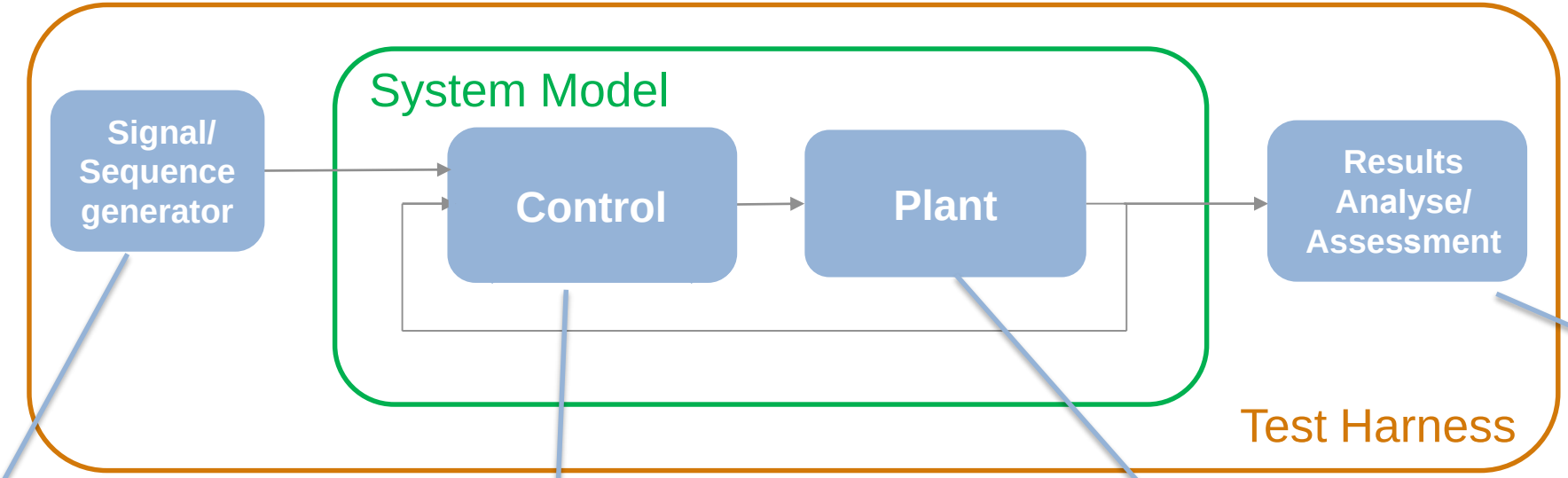


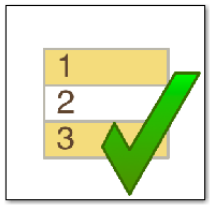
Save 40% to 80% of **Time to Market**

Why deploying model to Hardware?

- **Control Prototyping:** Physical plant test & characterization
- **Production code:** robust, maintenable & certified code for industrial product
- **Verification:** verifiy hardware controller

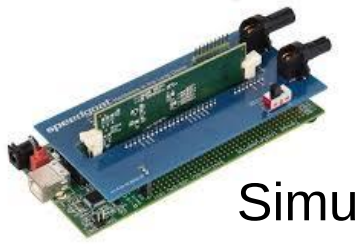
What can be modeled & deployed?



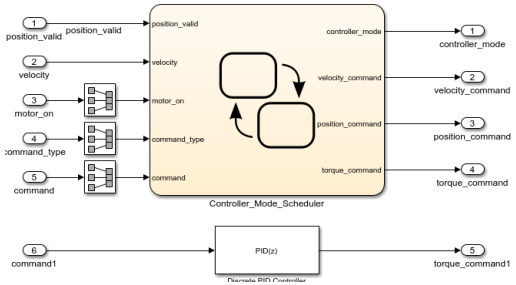


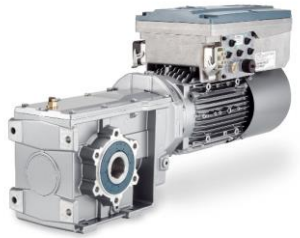
Simulink
Simulink Test
Simulink Requirements

Step	Transition	Next Step
PressNeitherButton RedButtonIN = false; GreenButtonIN = false;	1. after(1,sec)	PressBothButtons
PressBothButtons RedButtonIN = true; GreenButtonIN = true;	1. after(1,sec)	PressRedButton
PressRedButton RedButtonIN = true; GreenButtonIN = false;	1. after(1,sec)	PressGreenButton
PressGreenButton RedButtonIN = false; GreenButtonIN = true;	1. after(1,sec)	EndTest
EndTest		

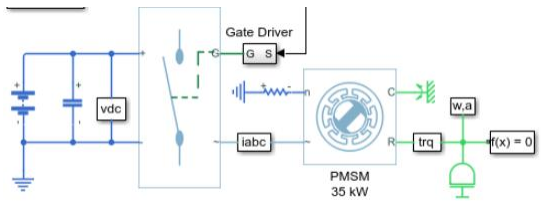


Simulink/Stateflow

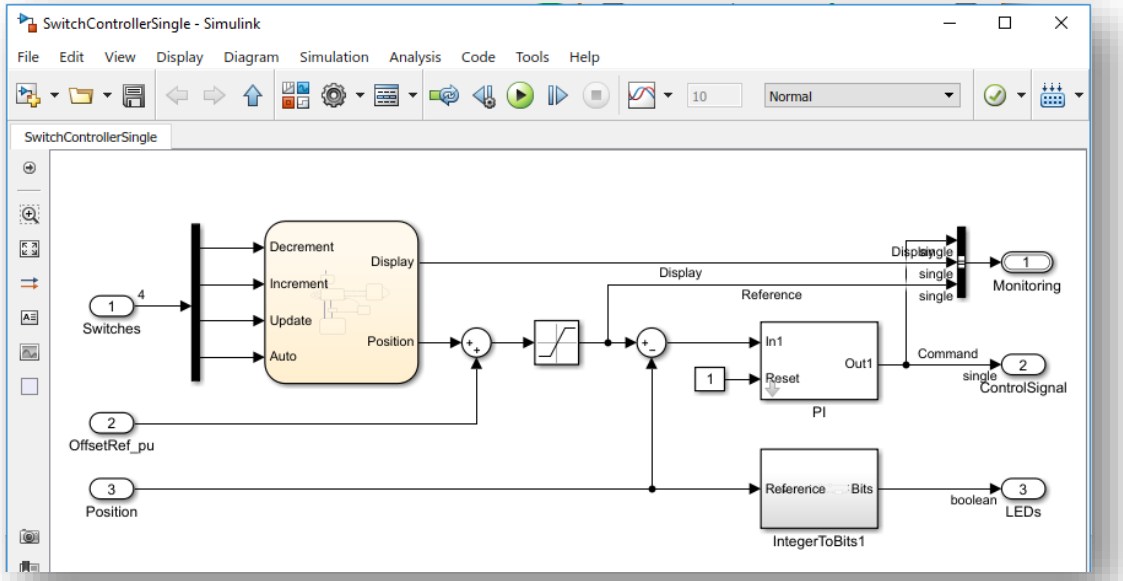




Simulink/Simscape

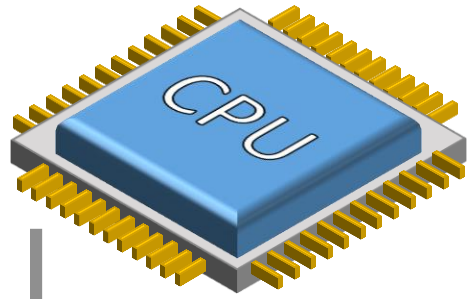


How to deploy? – MathWorks code generation tools



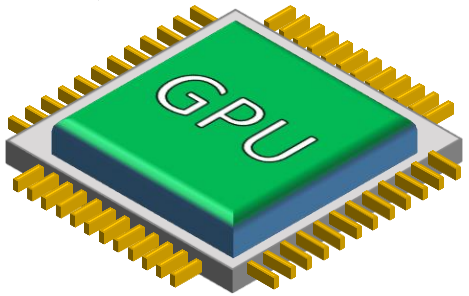
MATLAB / Simulink Coder

Embedded Coder

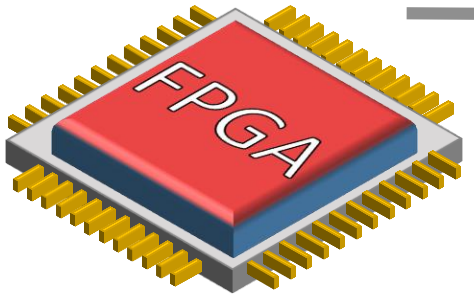


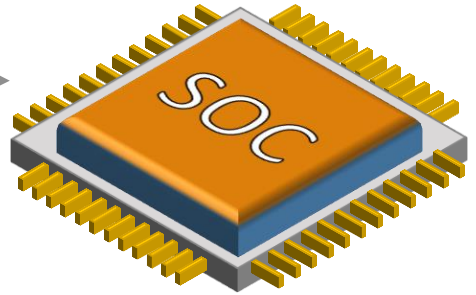
PLC coder

GPU coder

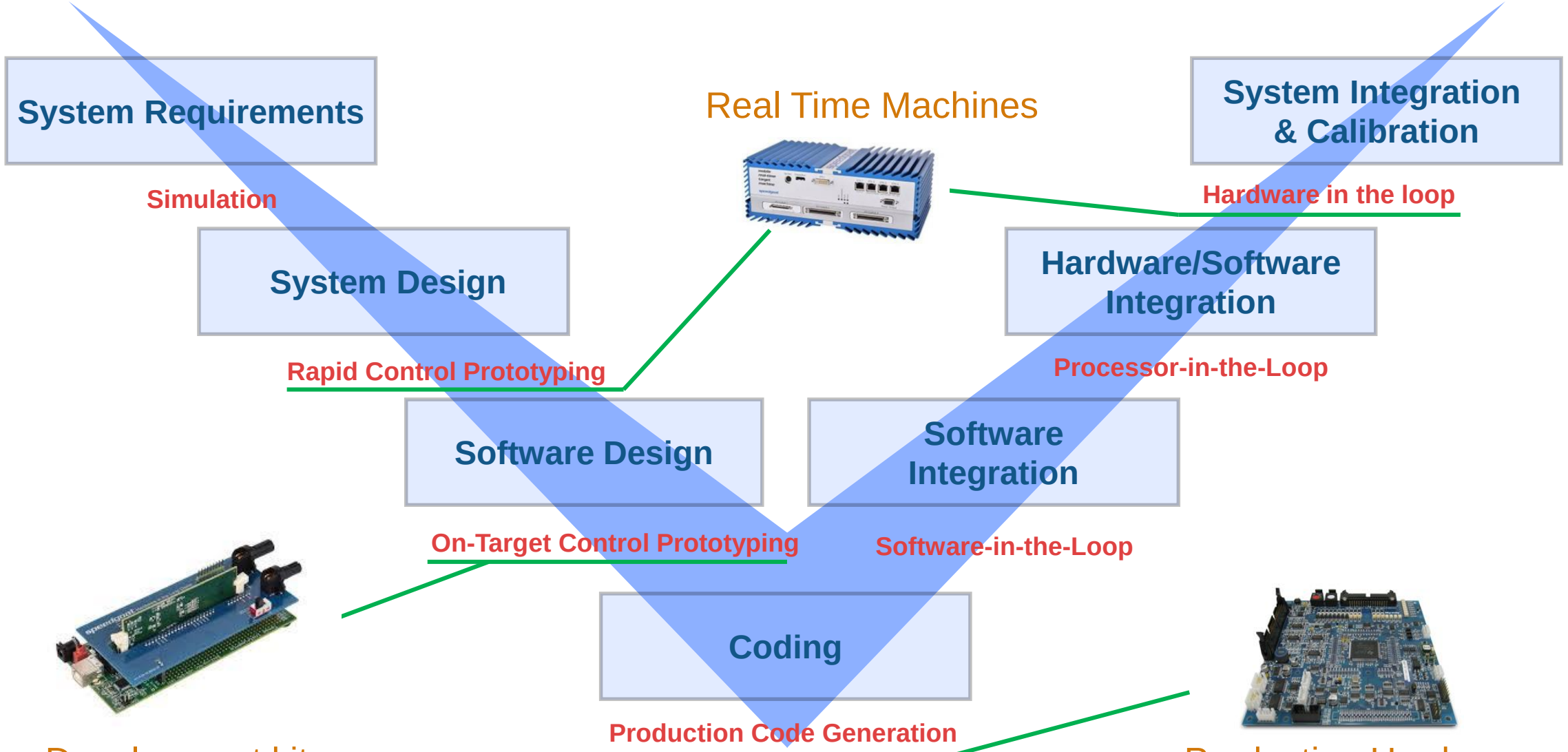


HDL Coder

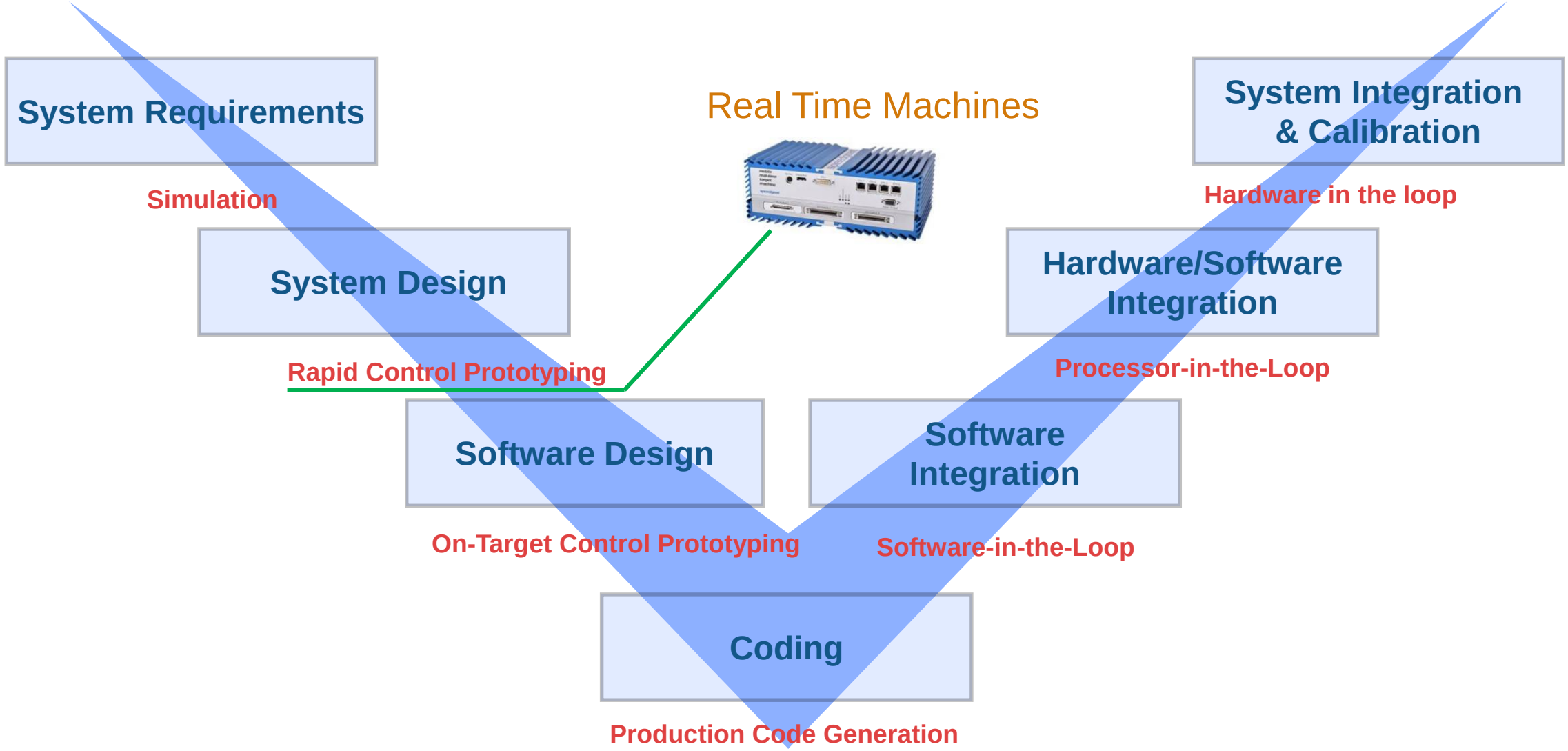




How to deploy? – Hardware targets



Rapid Control Prototyping



Rapid Control Prototyping (RCP)

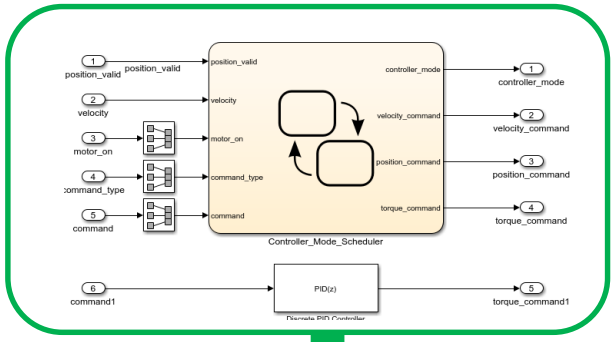
Context

- Target undefined
- Algorithms exploration
- Requirements consolidation

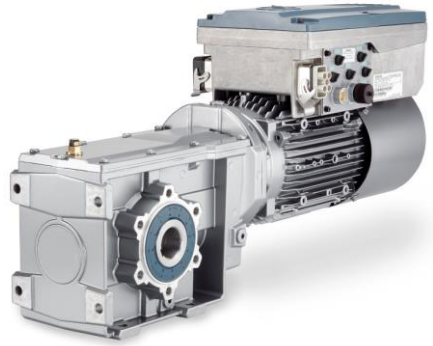
Values

- Keep algorithm generic
- Refine interfaces
- Monitor & Identify plant

Algorithms under evaluation



Real plant prototype



Real-time target

A fully integrated Solution

MathWorks®

Simulink Real-Time (SLRT)

- Control/physical modeling
- Test Automation
- Advance data analysis
- Code Gen (C/VHDL)

SLRT Kernel

speedgoat
real-time simulation and testing

Real-time Target Machines

- Modular hardware solutions
- Hardware driver library
- I/O-/protocol support
- Low latency FPGA-based solutions

Coordinated Customer Support

Who uses control prototyping and **Why** ?

Villanova University: Students compete at international level with a state-of-the-art autonomous robotic boat

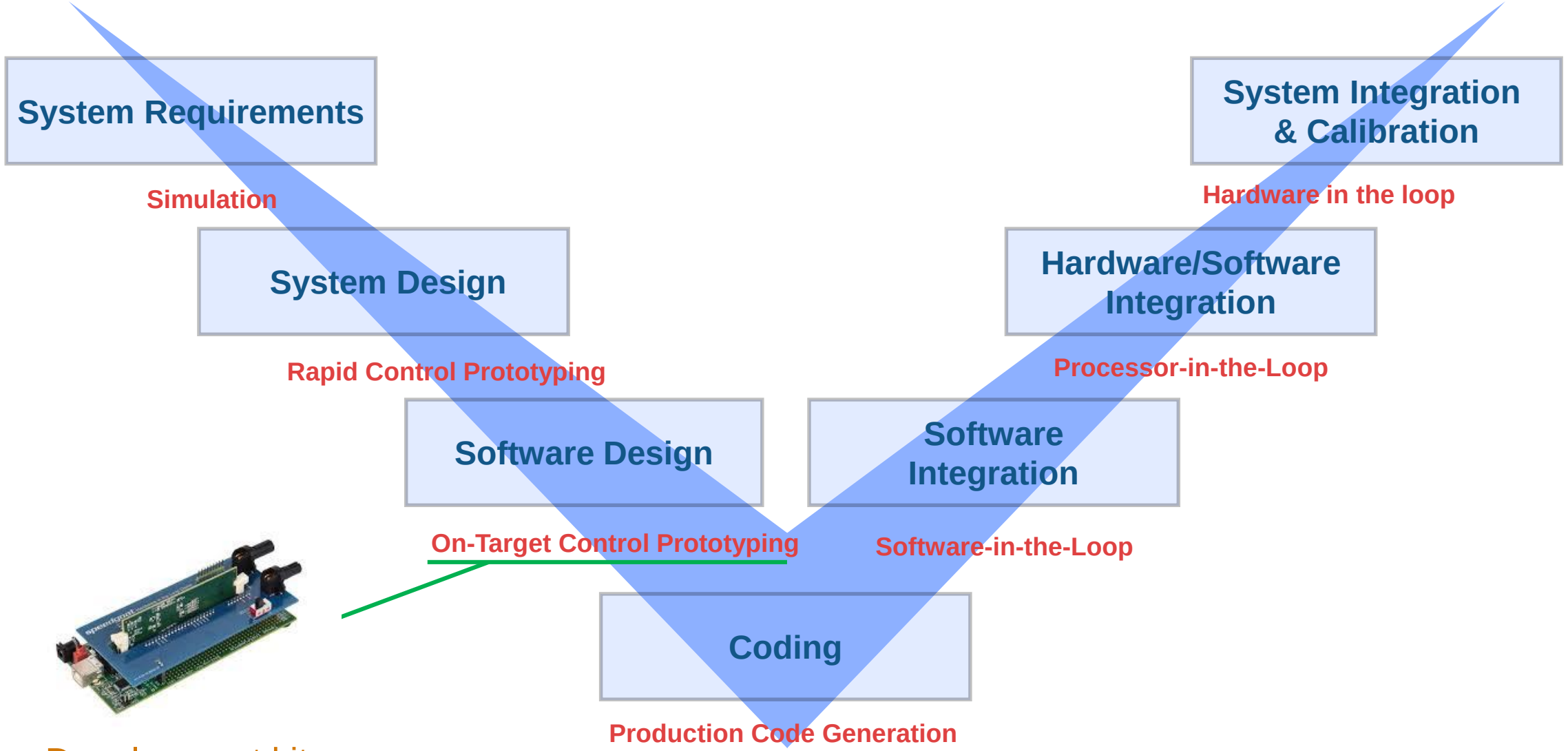
"Data logging & monitoring capabilities have proved invaluable for debugging"

Mobileye: Driving technology towards a fully autonomous vehicle

"With the Speedgoat system, changing parameters and tuning the system is very easy and straightforward. It saves us a lot of time. There's no need to re-compile and burn each new version of the control algorithm."

— Eyal Bagon, Senior Director Production Software and Development Coordination at Mobileye

On-Target Control Prototyping



Development kits

On-Target Control Prototyping

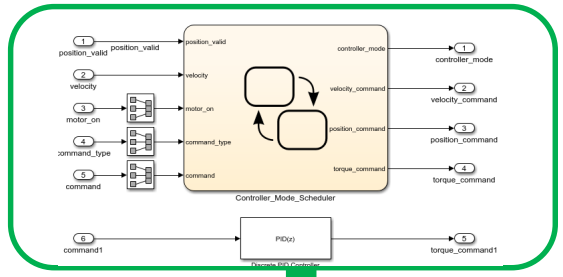
Context

- Algorithms selected
- Interfaces identified
- First target choice

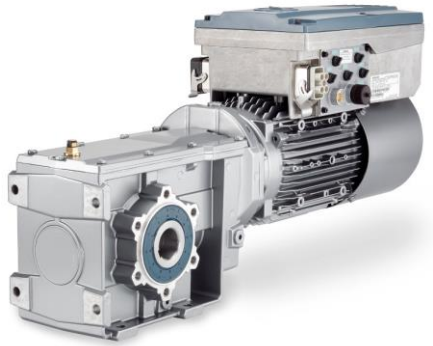
Values

- Target choice validation
- Manage target specificities
- Minimum viable product

Algorithms for production

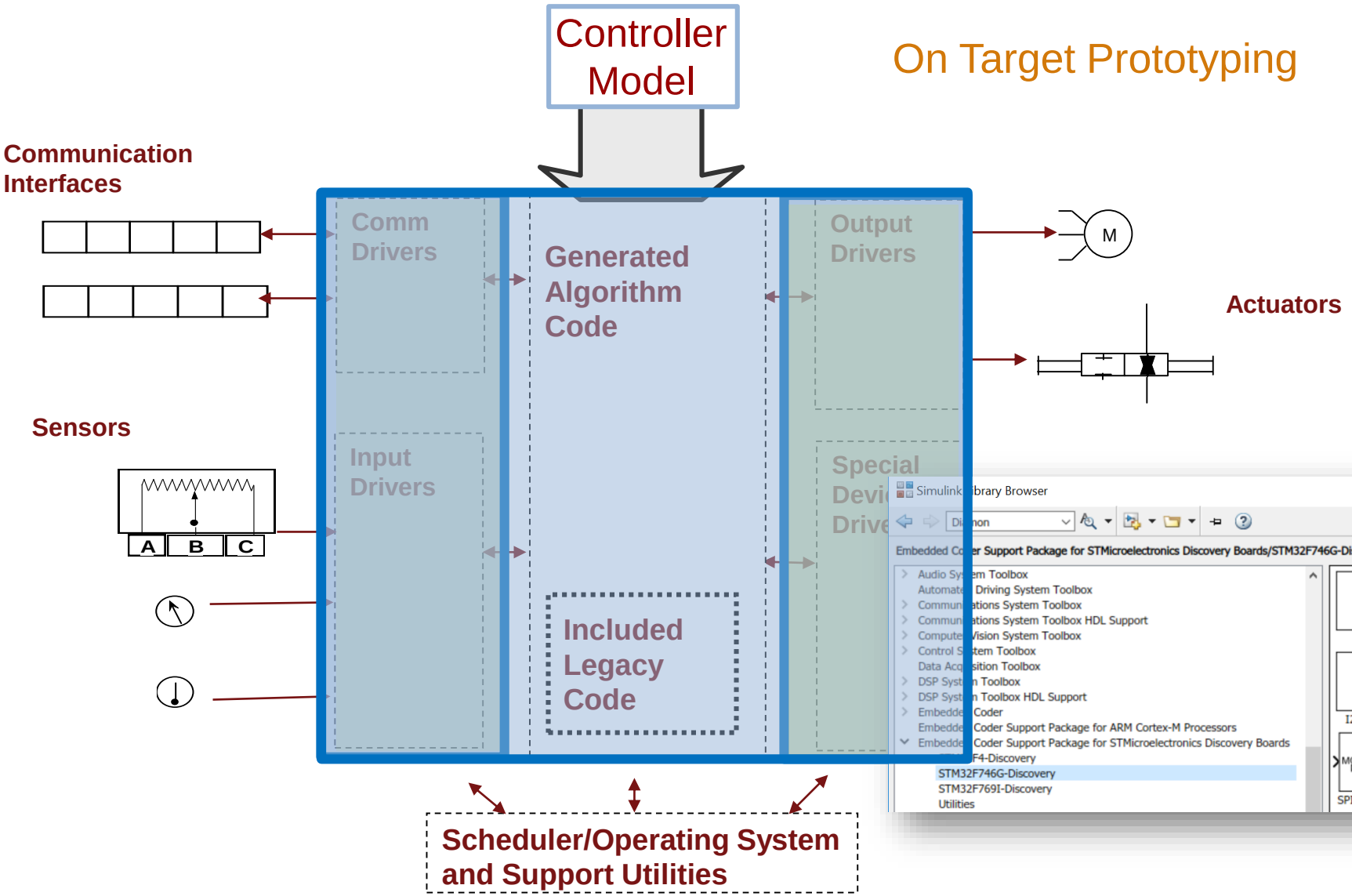


Real plant prototype



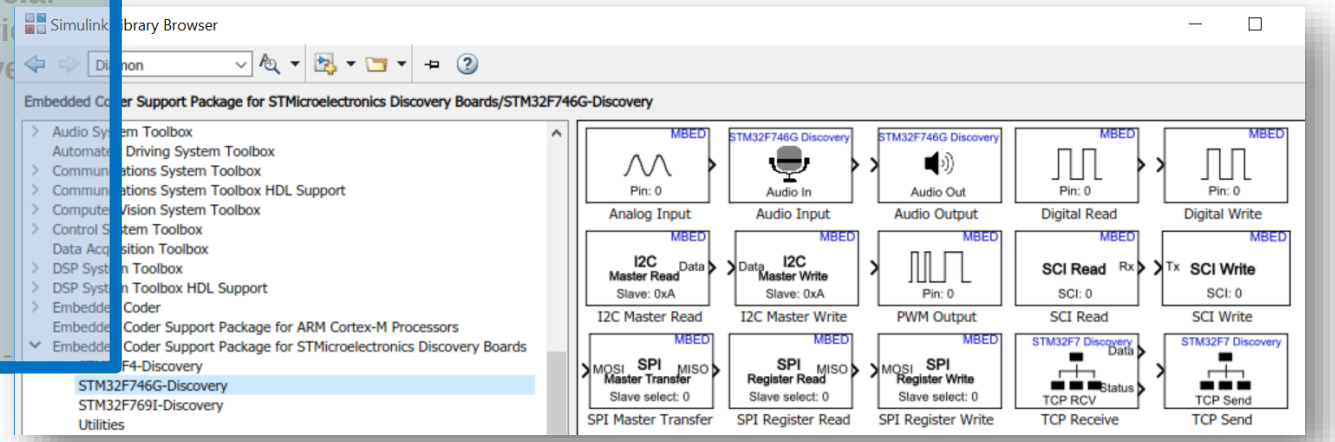
Development kit

On-Target Control Prototyping

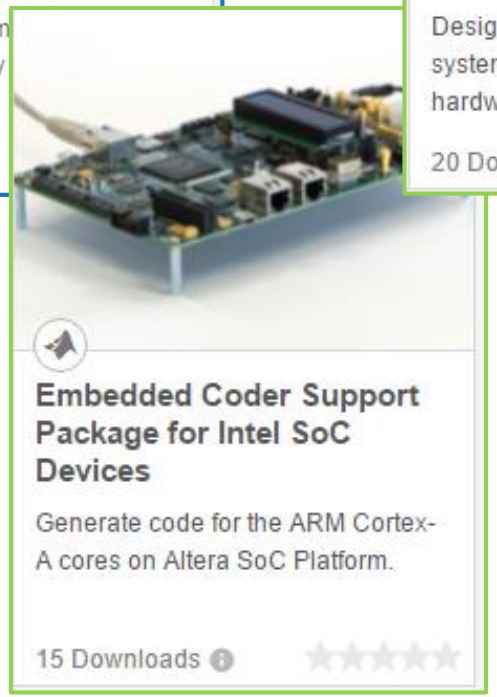


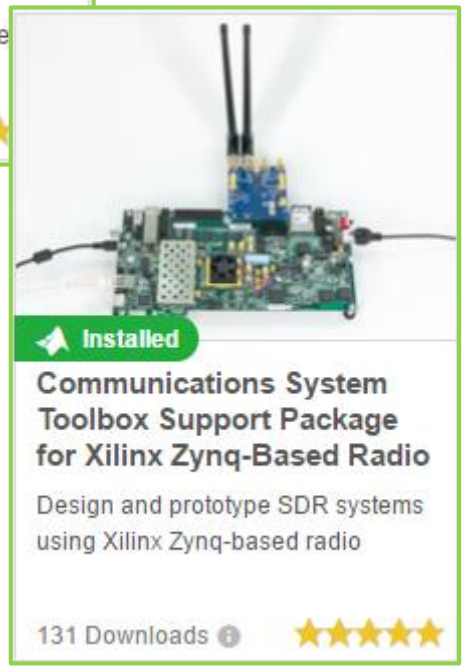
On Target Prototyping

Hardware support packages



Using Development Kits for Rapid Prototyping





MCU,

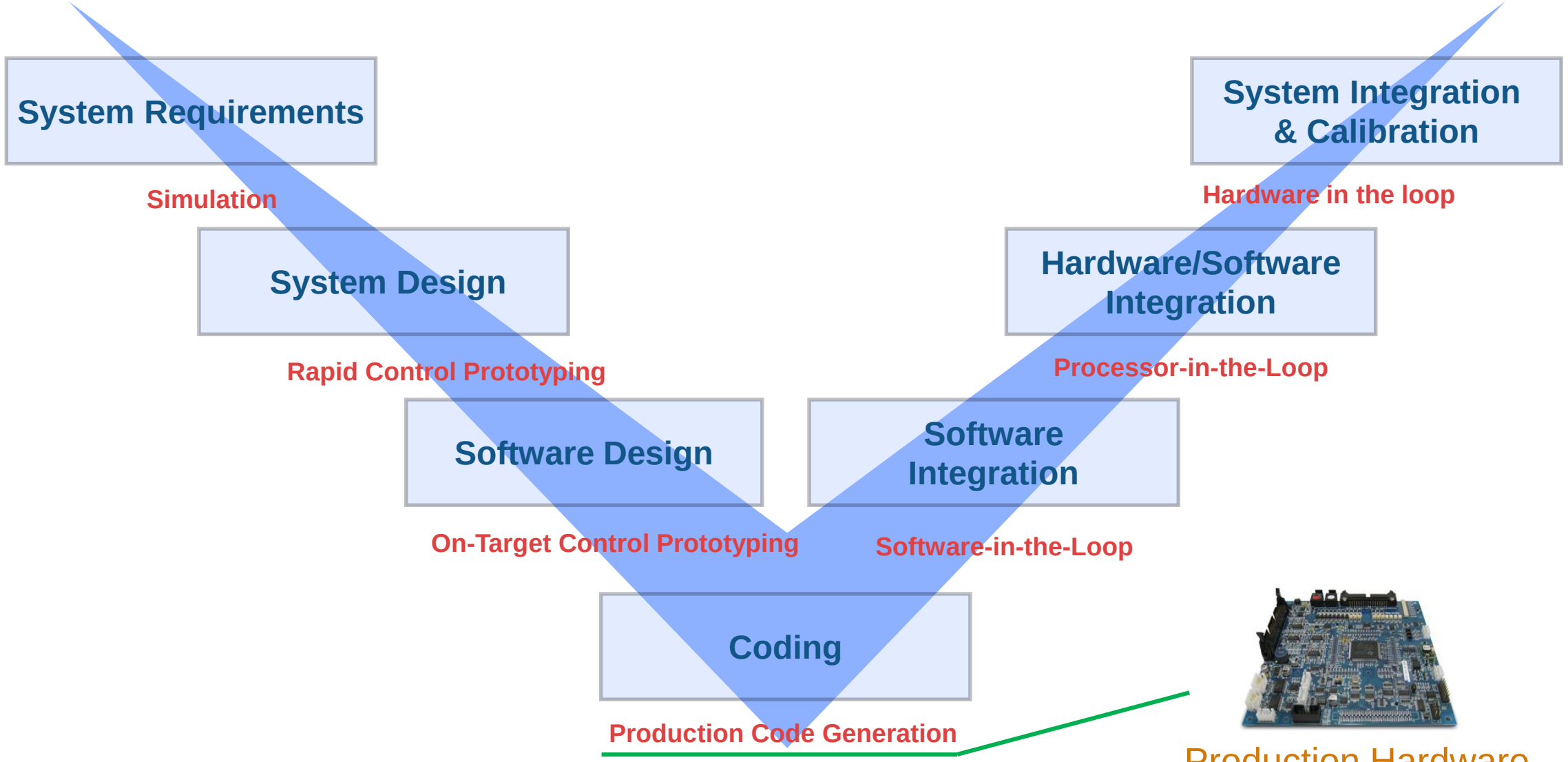
ARM,

FPGA,

SOC,

GPU

Production Code



Production Code Generation

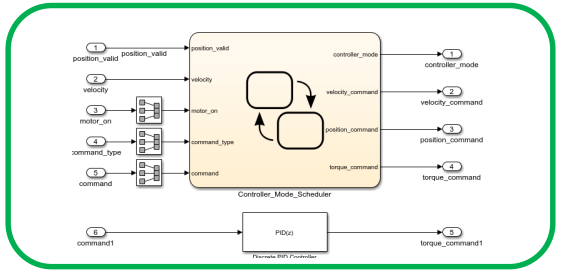
Context

- Algorithms verified in simulation
- Target choice validated
- C/C++ code project started

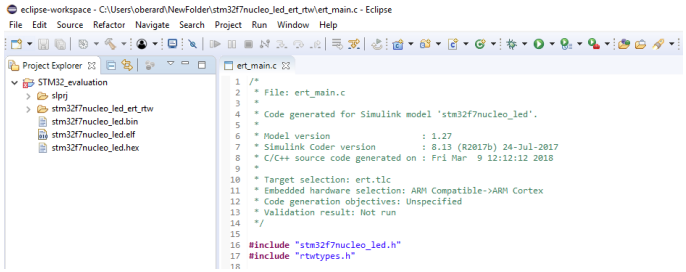
Values

- Ease transfer from Algorithm to Software team
- Validated algorithms integration
- Code certification Workflow

Algorithms for production

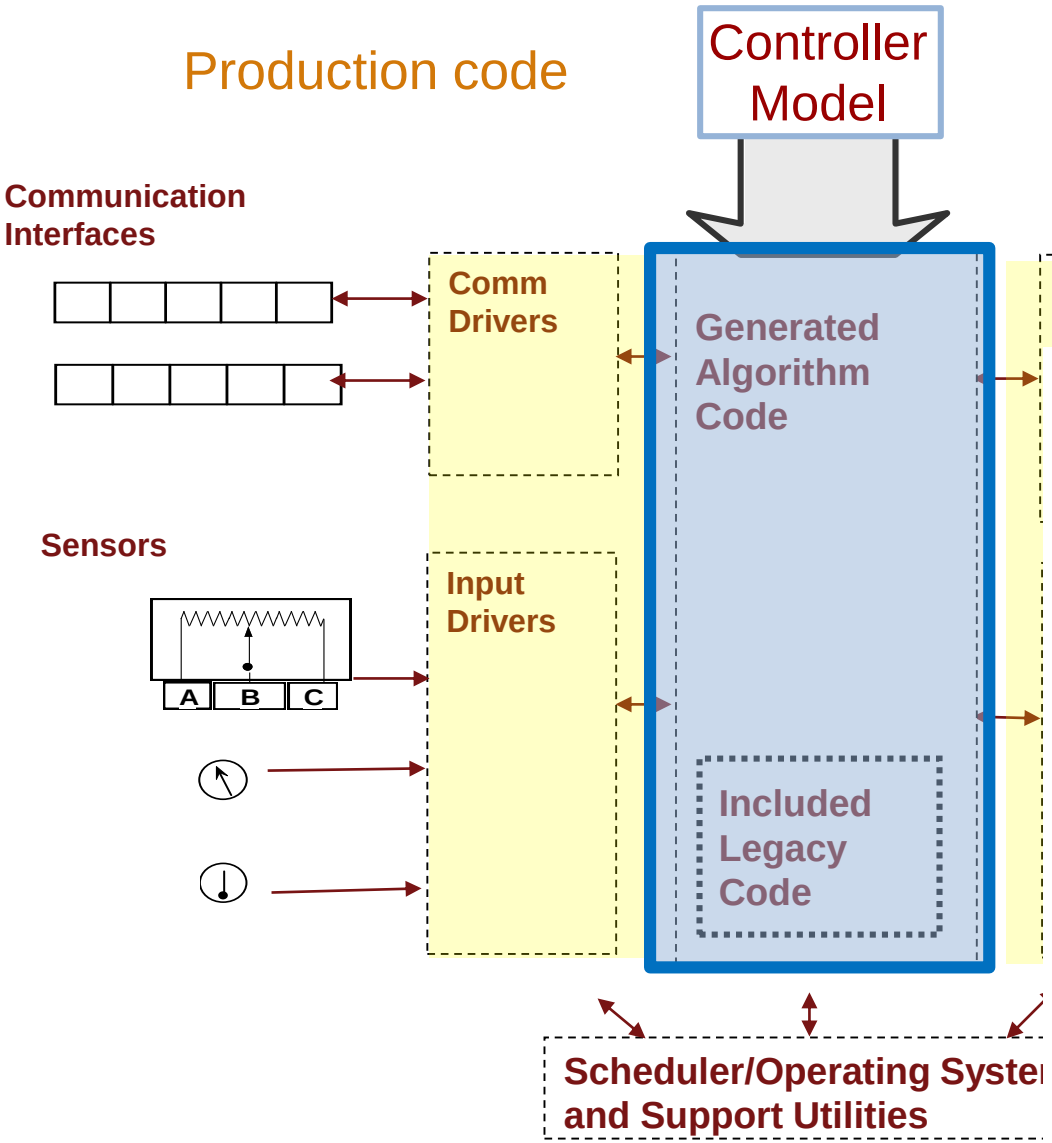


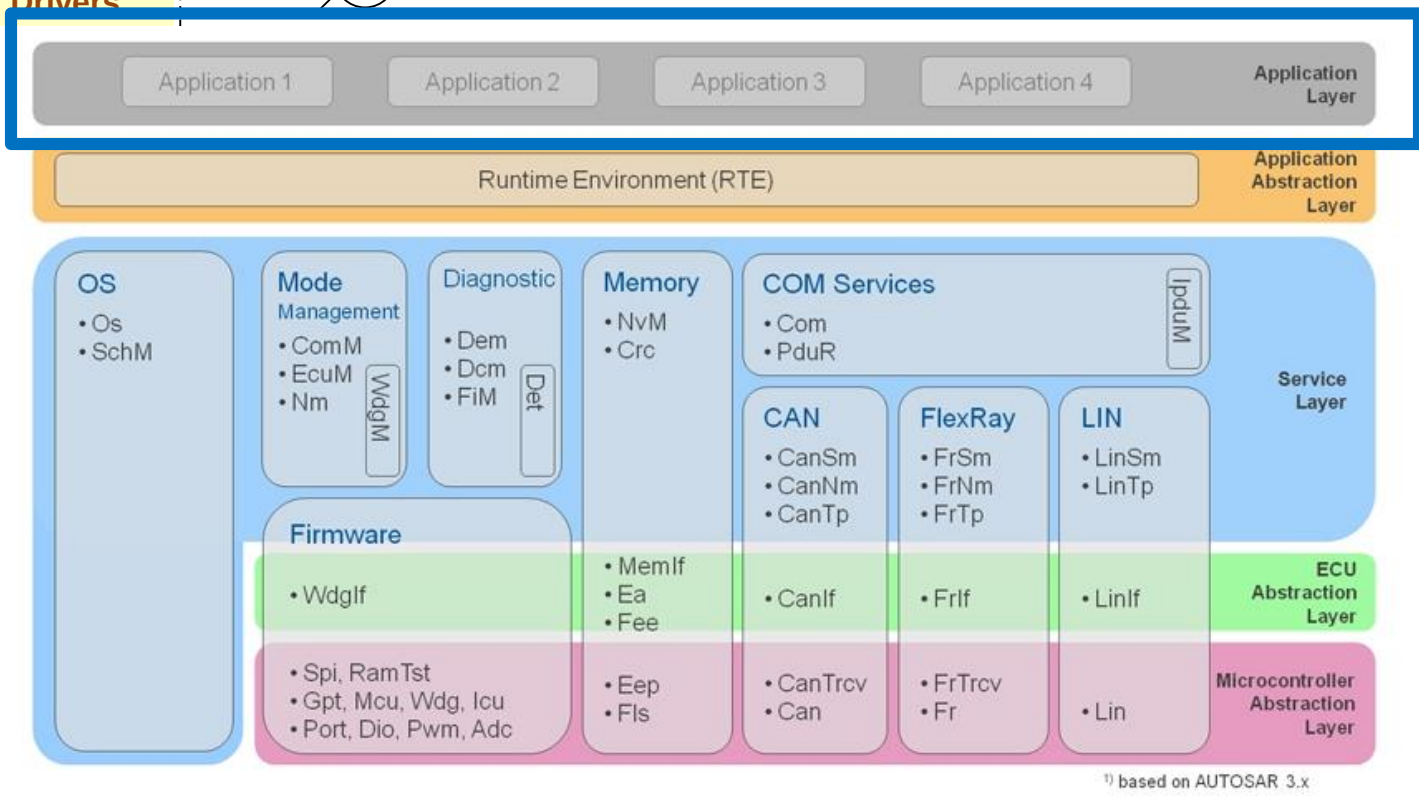
Software project



Production hardware

Production Code Generation





Who uses generated production code and **Why**?

Airbus Helicopters Accelerates Development of DO-178B Certified Software with Model-Based Design

"We use our system design model in Simulink for ARP4754 to establish stable, objective requirements. We save time by using the model as the basis for our software design model for DO-178—from which we generate flight code—and reusing validation tests for software verification."

— *Ronald Blanrue, Airbus Helicopters*

The Airbus Helicopters EC130 helicopter.

Challenge

Speed the development, validation, and verification of DO-178B certified helicopter flight software

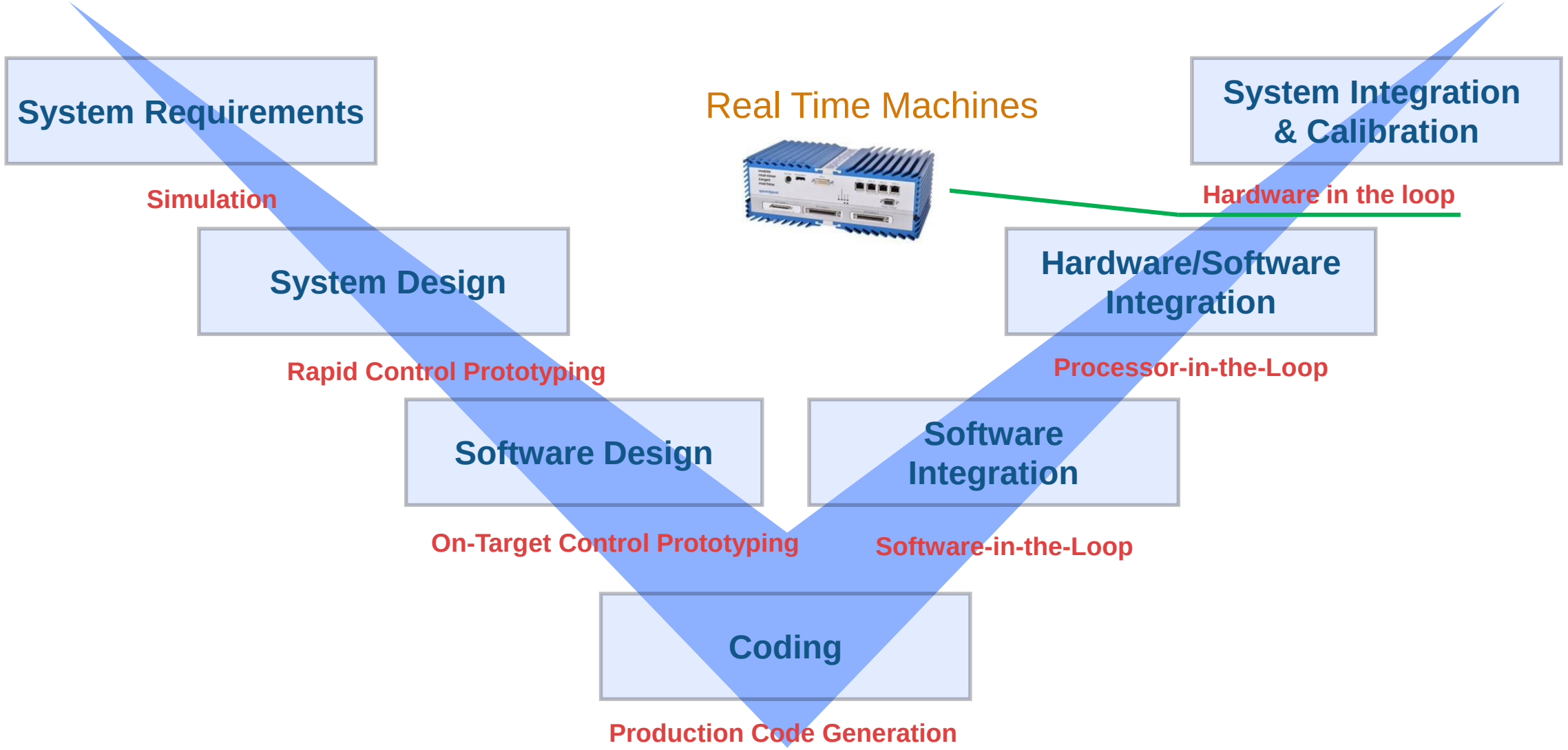
Solution

Use Model-Based Design to model the system design and software design, and to generate flight code

Results

- Software testing time cut by two-thirds
- Requirements stabilized earlier
- Certified flight software automatically generated

Hardware in The Loop



Hardware In the Loop (HIL)

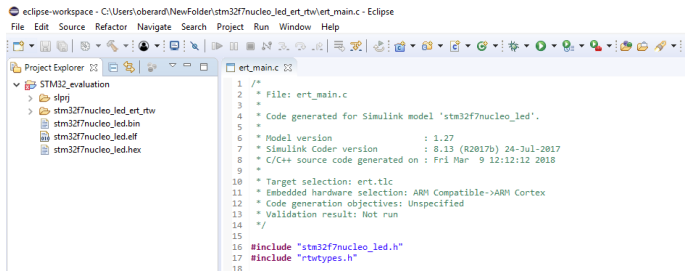
Context

- Hardware Control board ready
- Implemented production code

Values

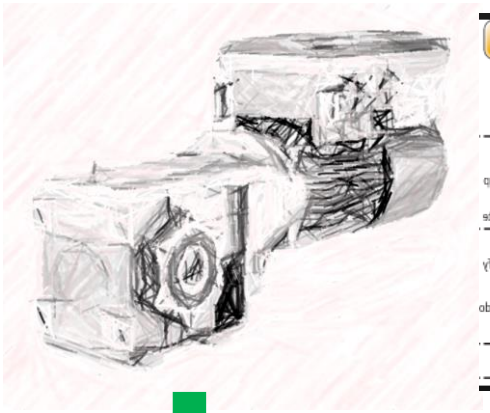
- HW + SW verification
- Incremental verification
- Reuse Test Scenarios

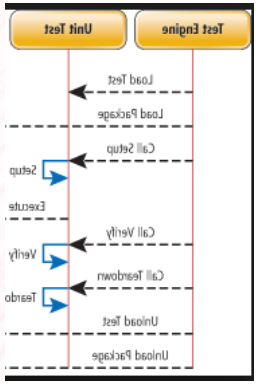
Software project for production



Production hardware

System & Test Model





Real-time target

Who uses Hardware In the Loop and **Why** ?

Schindler Elevator Corporation: Validating Schindler's next generation elevator controller family with hardware-in-the-loop simulation

"We didn't want to have issues with the linking of the model and hardware, so we decided to work with Speedgoat.

The i
so it

"The

— Ma

— Titu

Aalto University: Students' mission to get Finland's first satellite into orbit

"The cost of setting up and using a HIL solution is very low compared to performing all ADCS tests with a physical test setup emulating the orbit environment."

"The ability to perform HIL testing allows even the most complex satellite mission operations to be verified on real hardware."

— *Tuomas Tikka, Aalto University*

Case Study

Applications

What do these applications have in common?

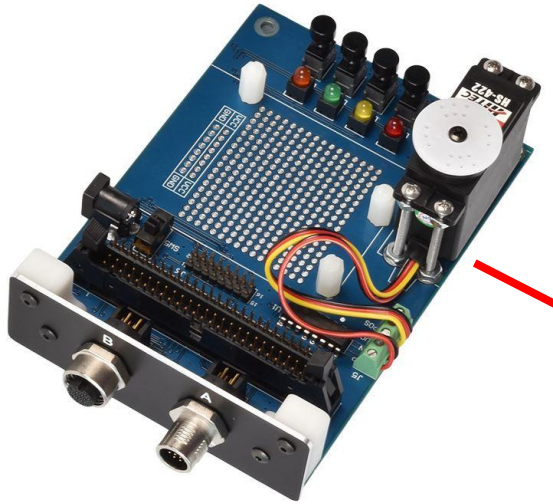
Many things…

Including **Motor control**

Hardware setup

Speedgoat Real Time Machine

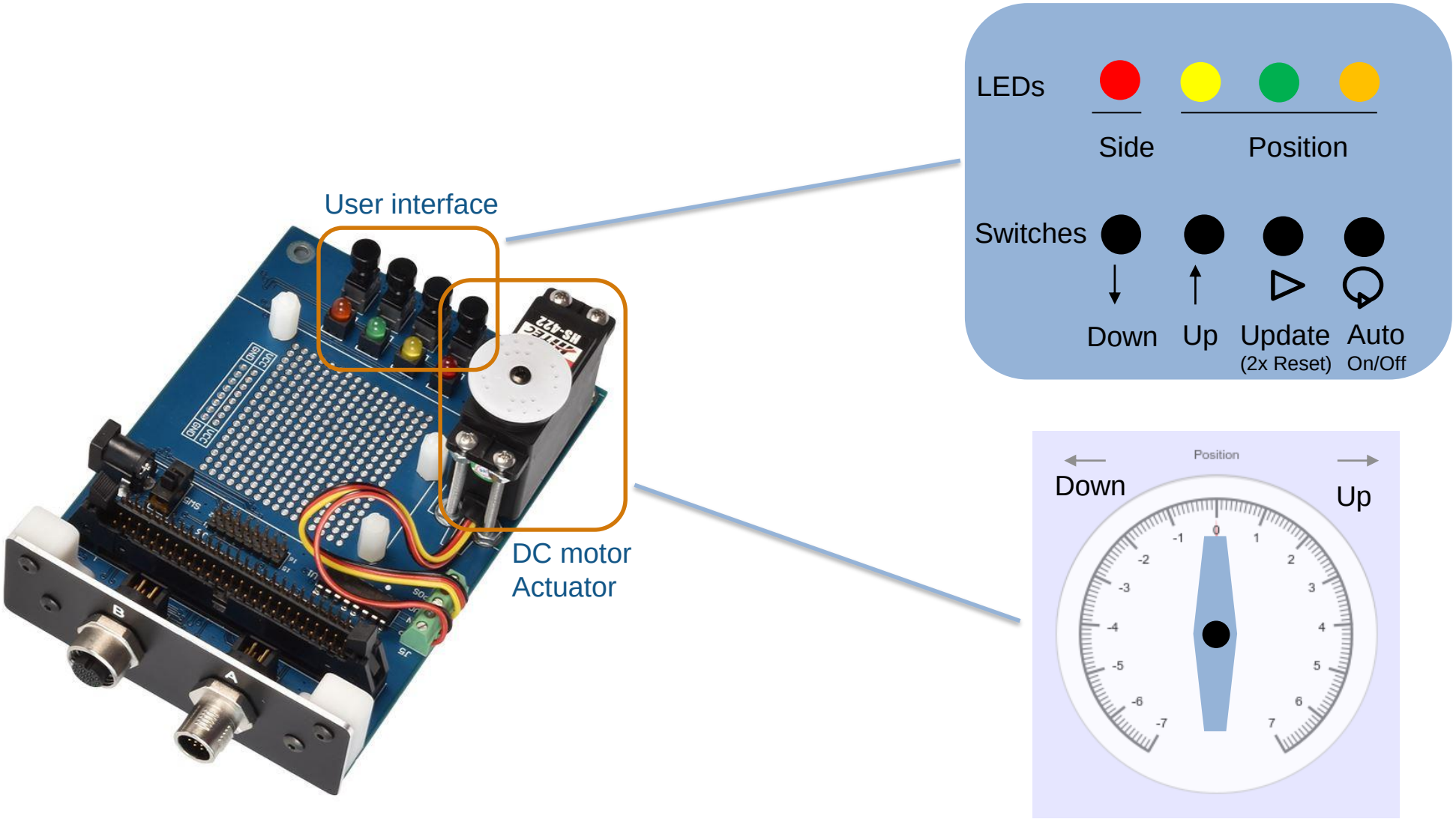
Speedgoat DUT demo kit



Speedgoat C2000 RCP demo kit

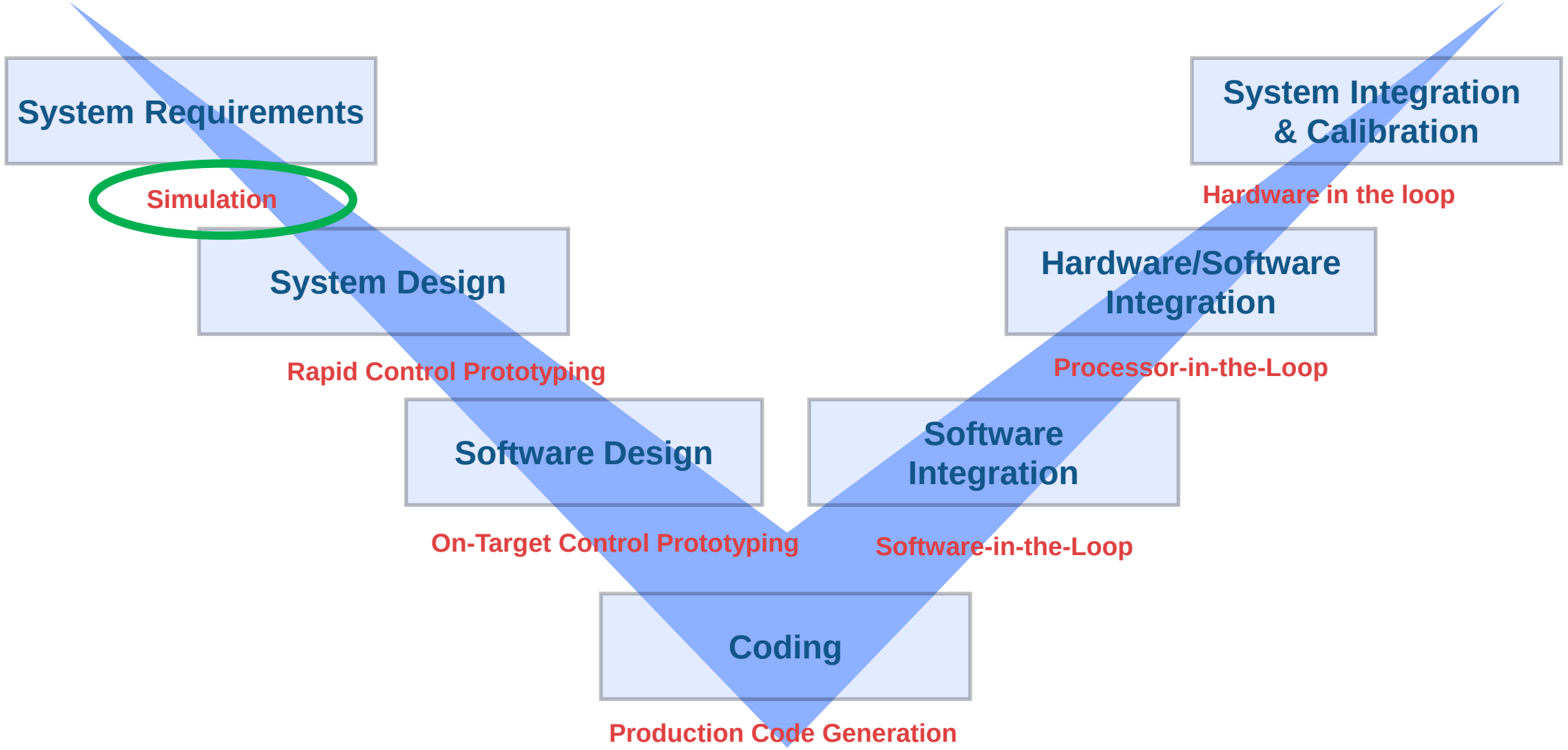
Available for you!

Control objectives

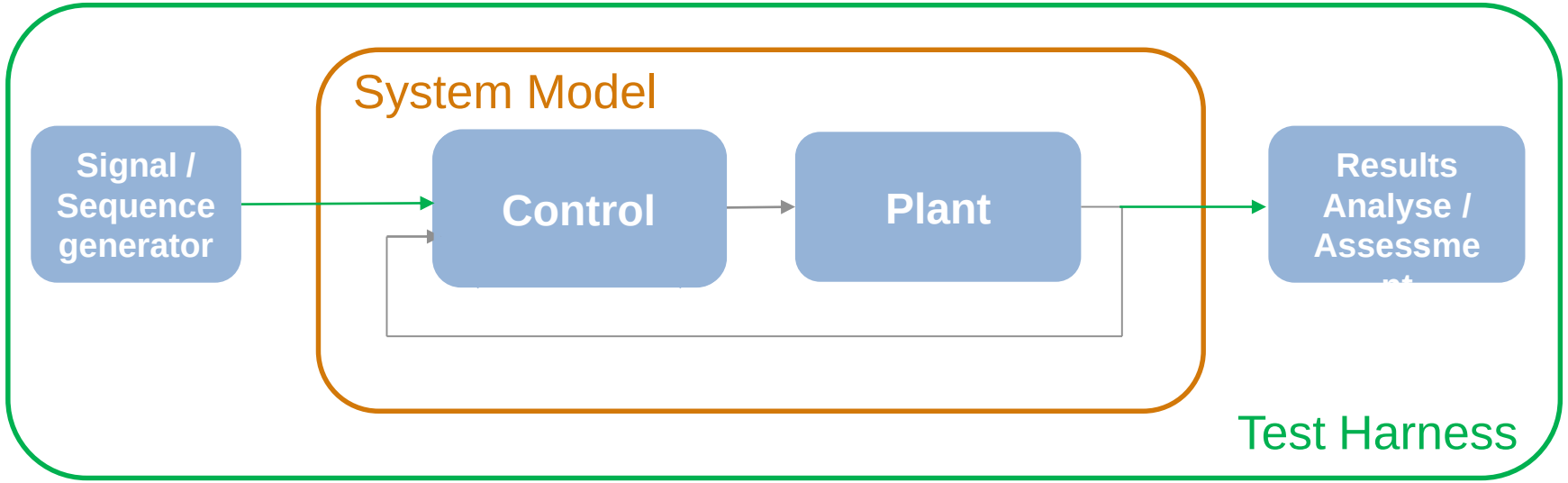


Demonstration

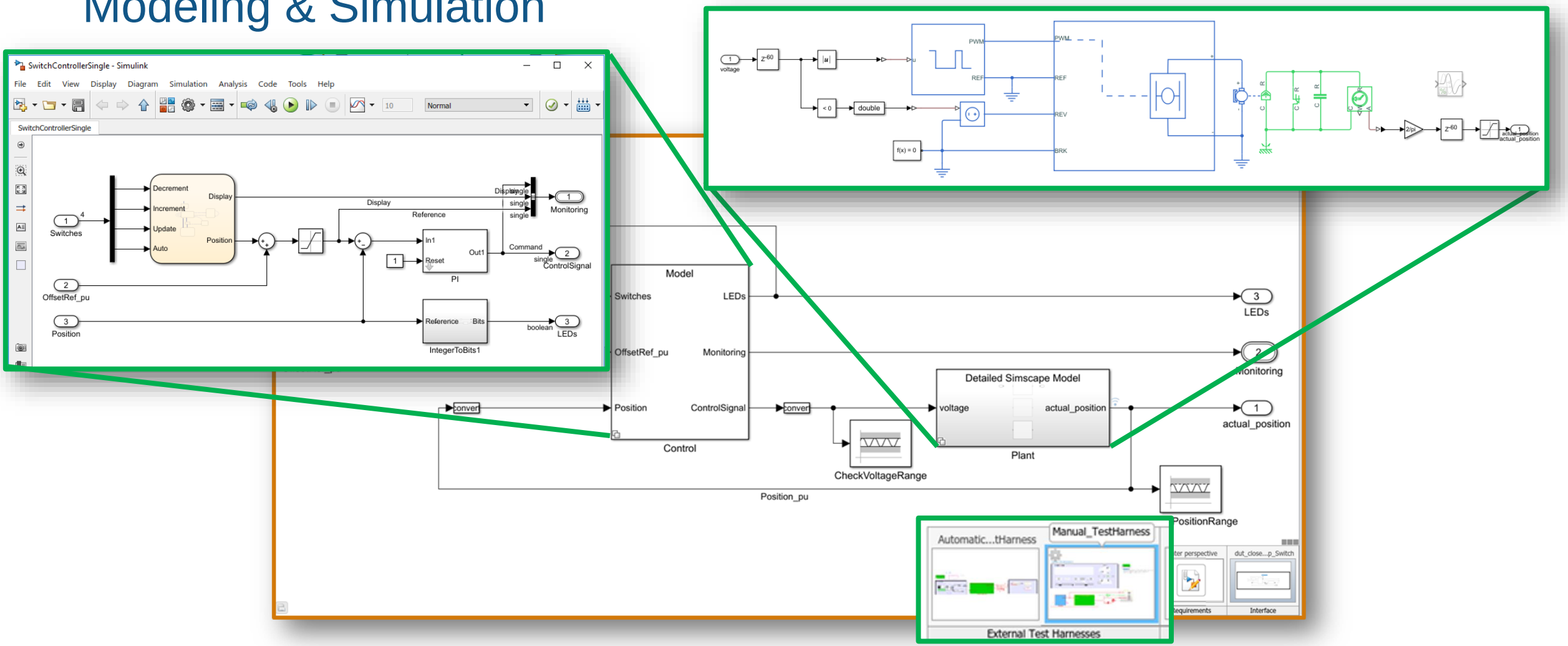
Modeling & Simulation



Modeling & Simulation

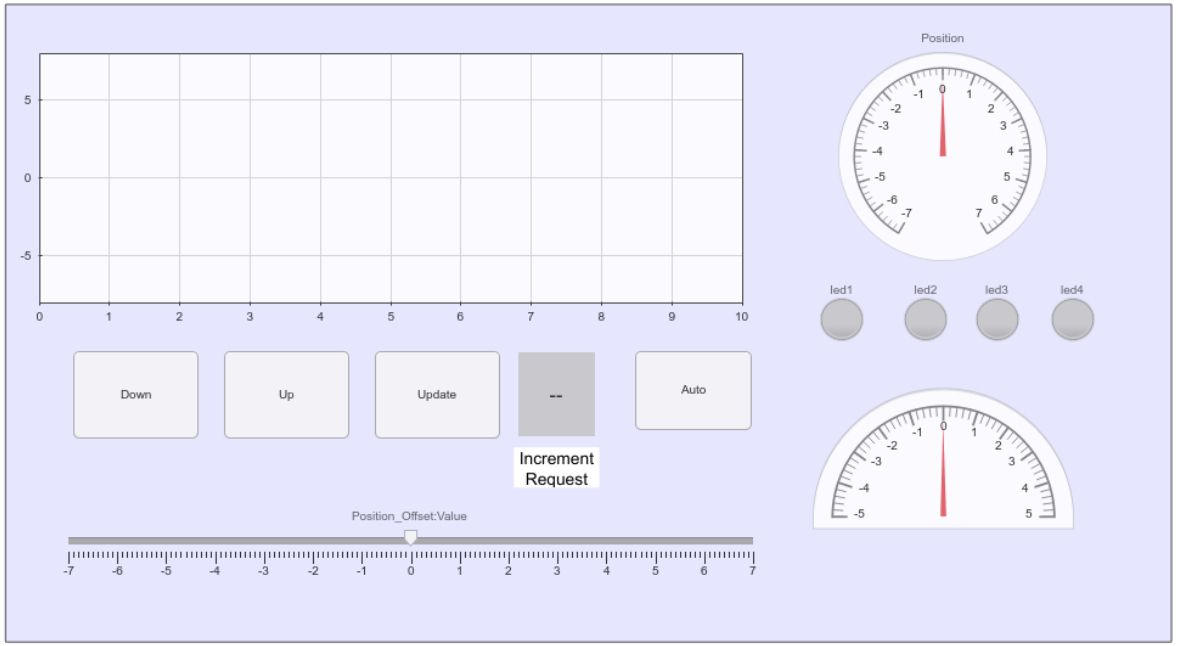


Modeling & Simulation



Modeling & Simulation

Manual Test Harness

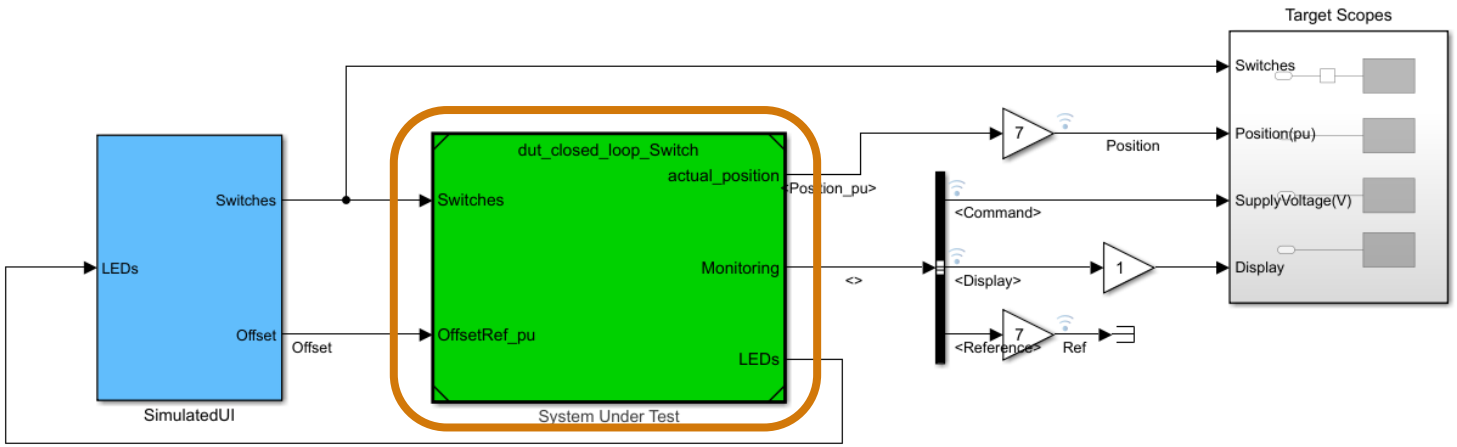


RCP

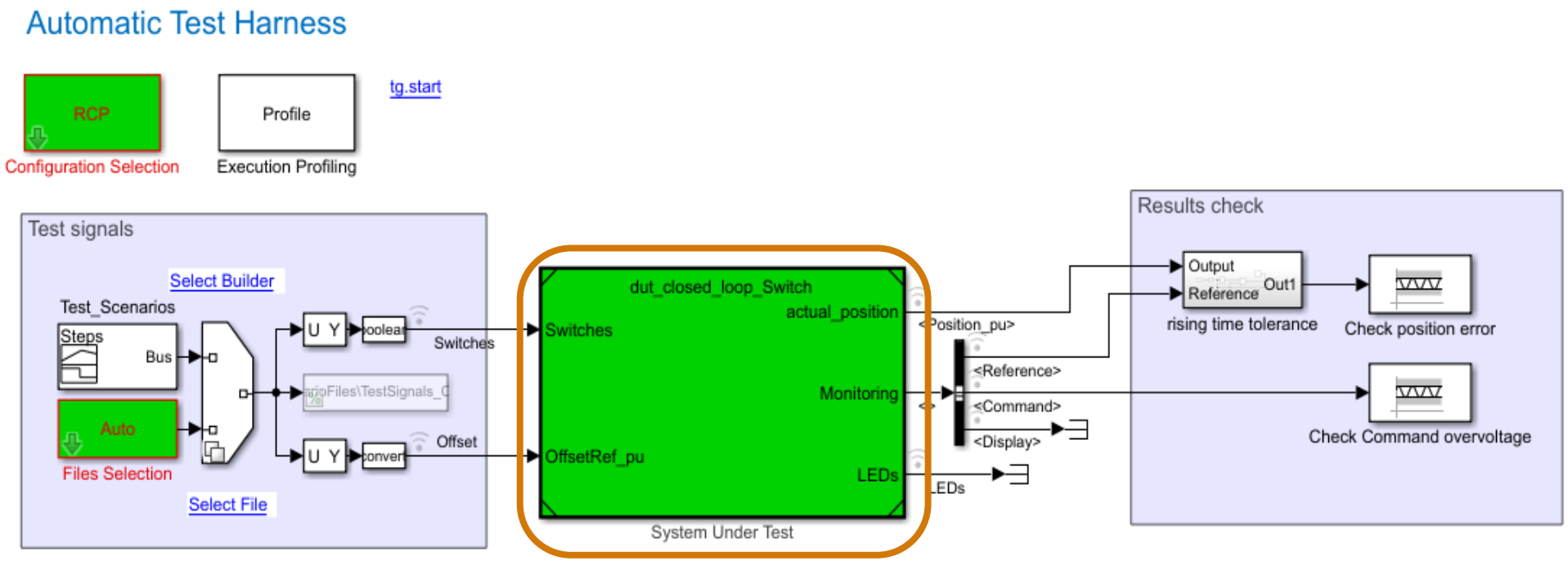
Configuration Selection

User Manual:

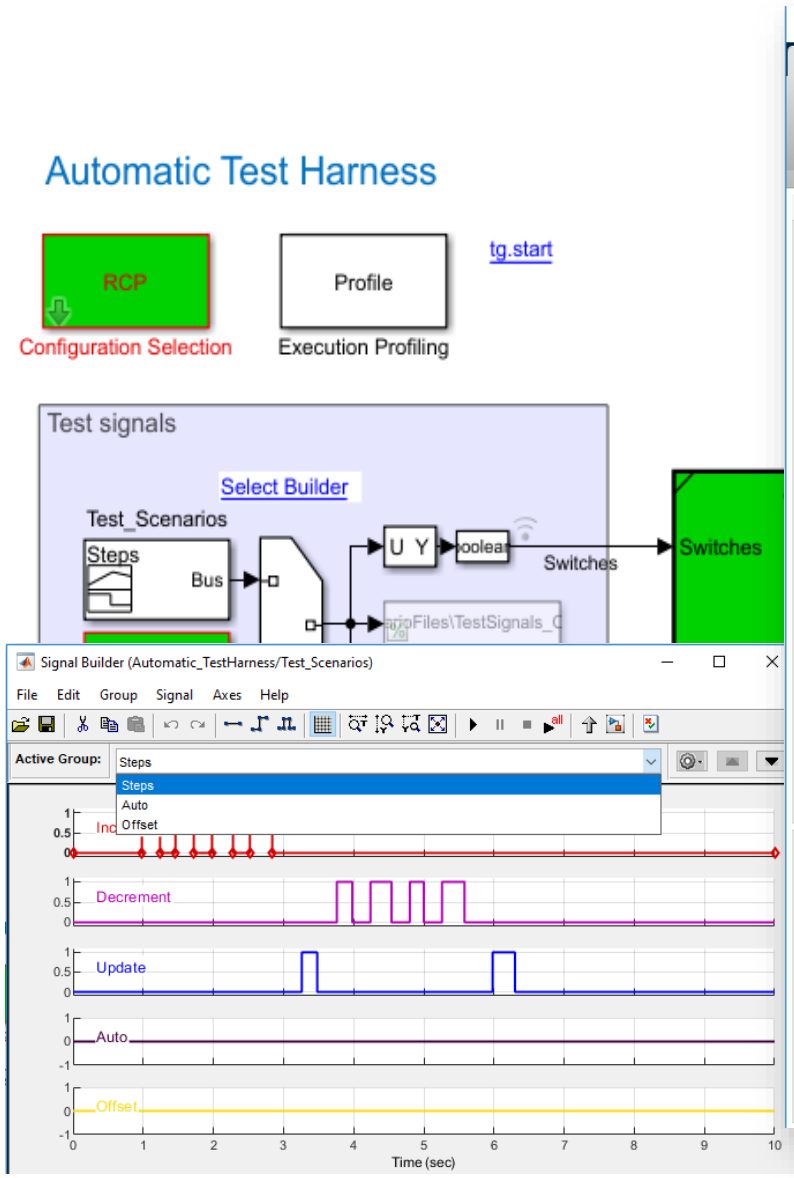
> Click Down/Up button to set the number of step increment
> Click Update to Move servo to the desired step increment.
> Double click Update to Reset servo to offset value
> Click Auto Button to start or stop Auto move

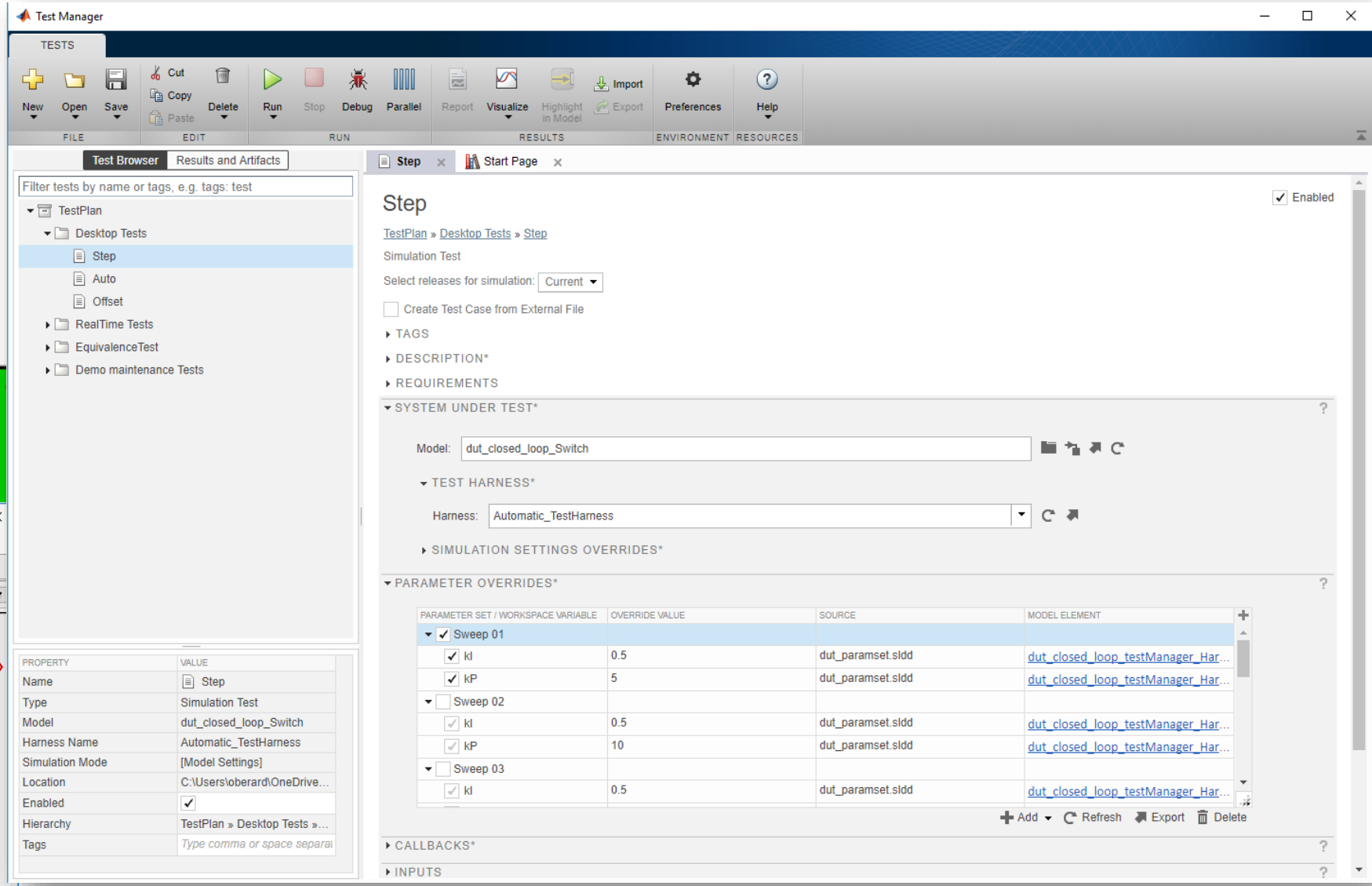


Modeling & Simulation

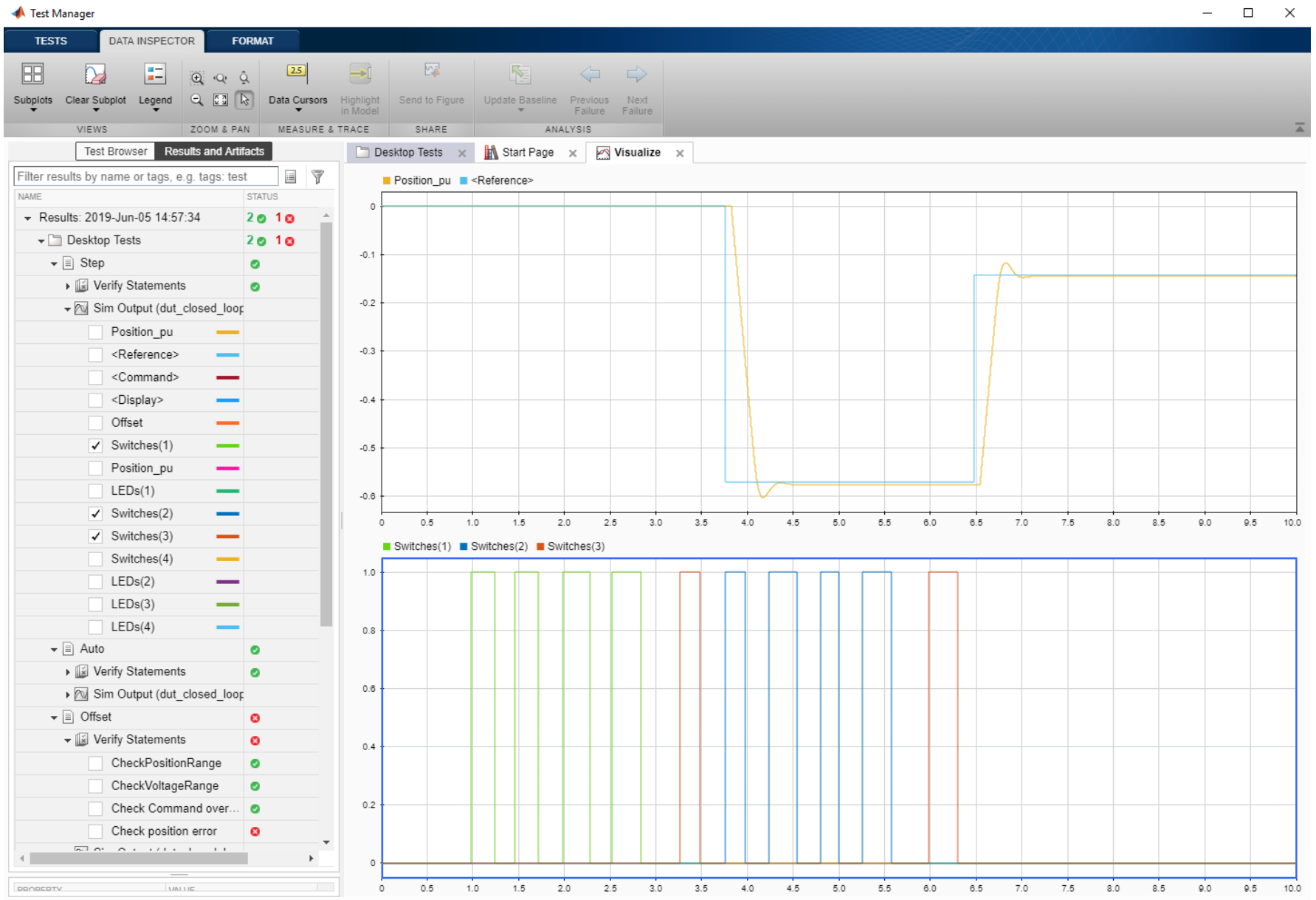


Modeling & Simulation

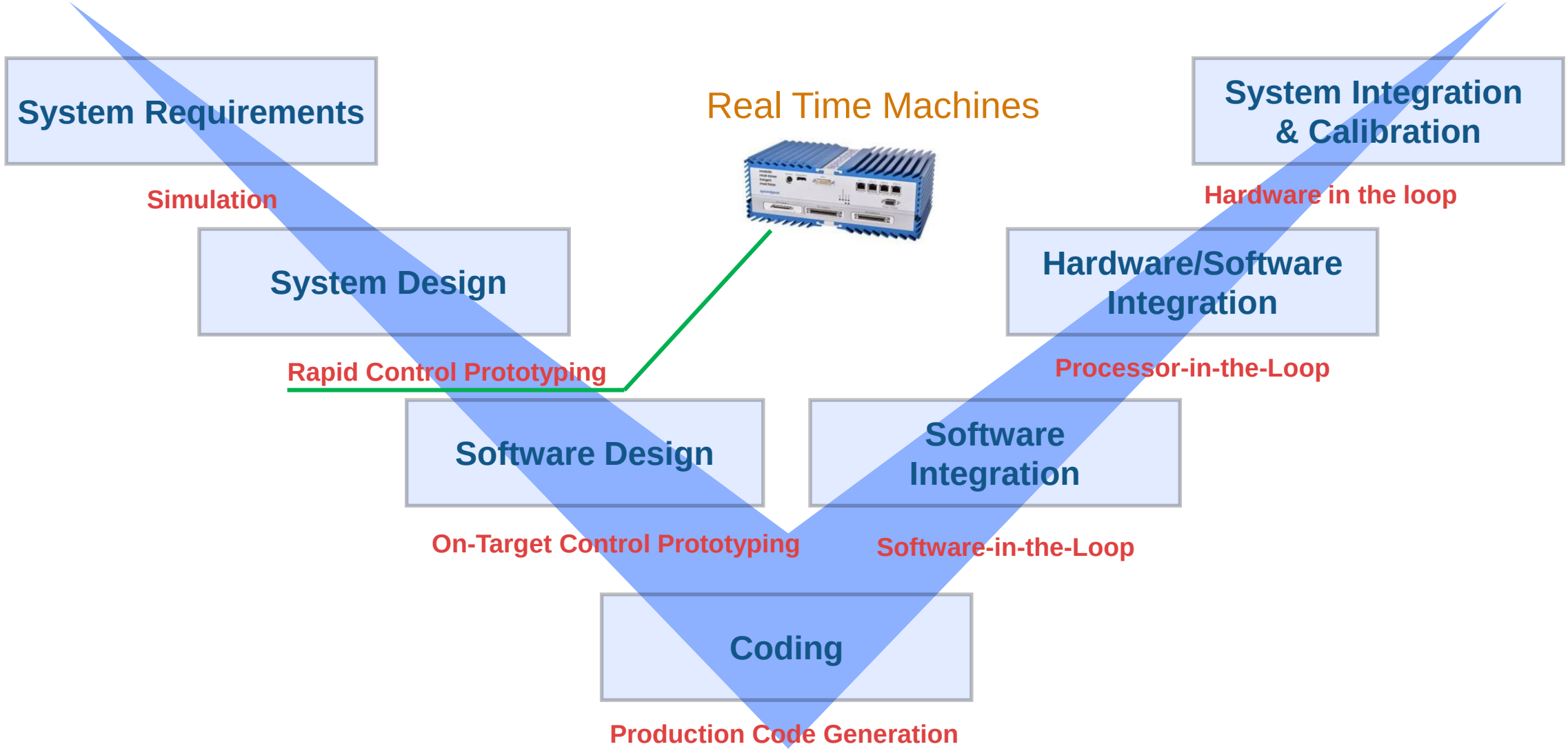




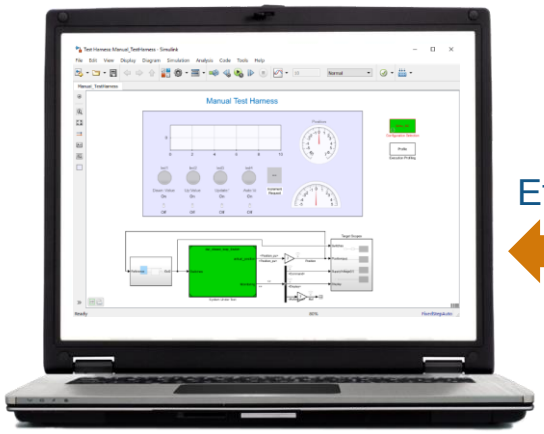
Modeling & Simulation - Run & Analyse tests



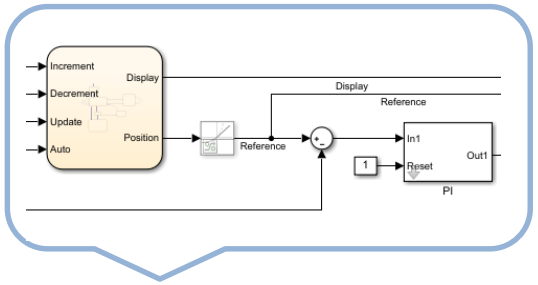
Rapid Control Prototyping



Rapid Control Prototyping



Ethernet

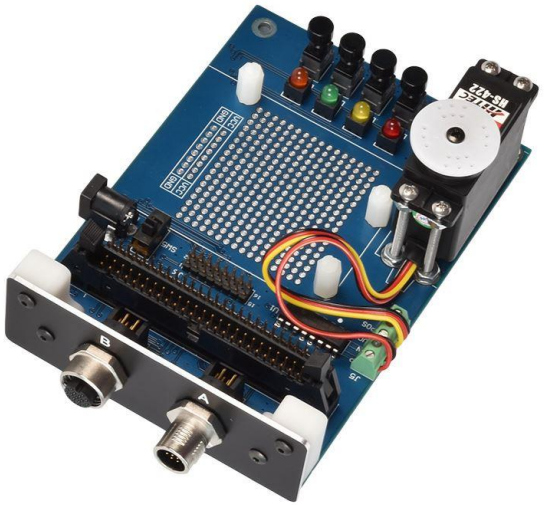


Controller

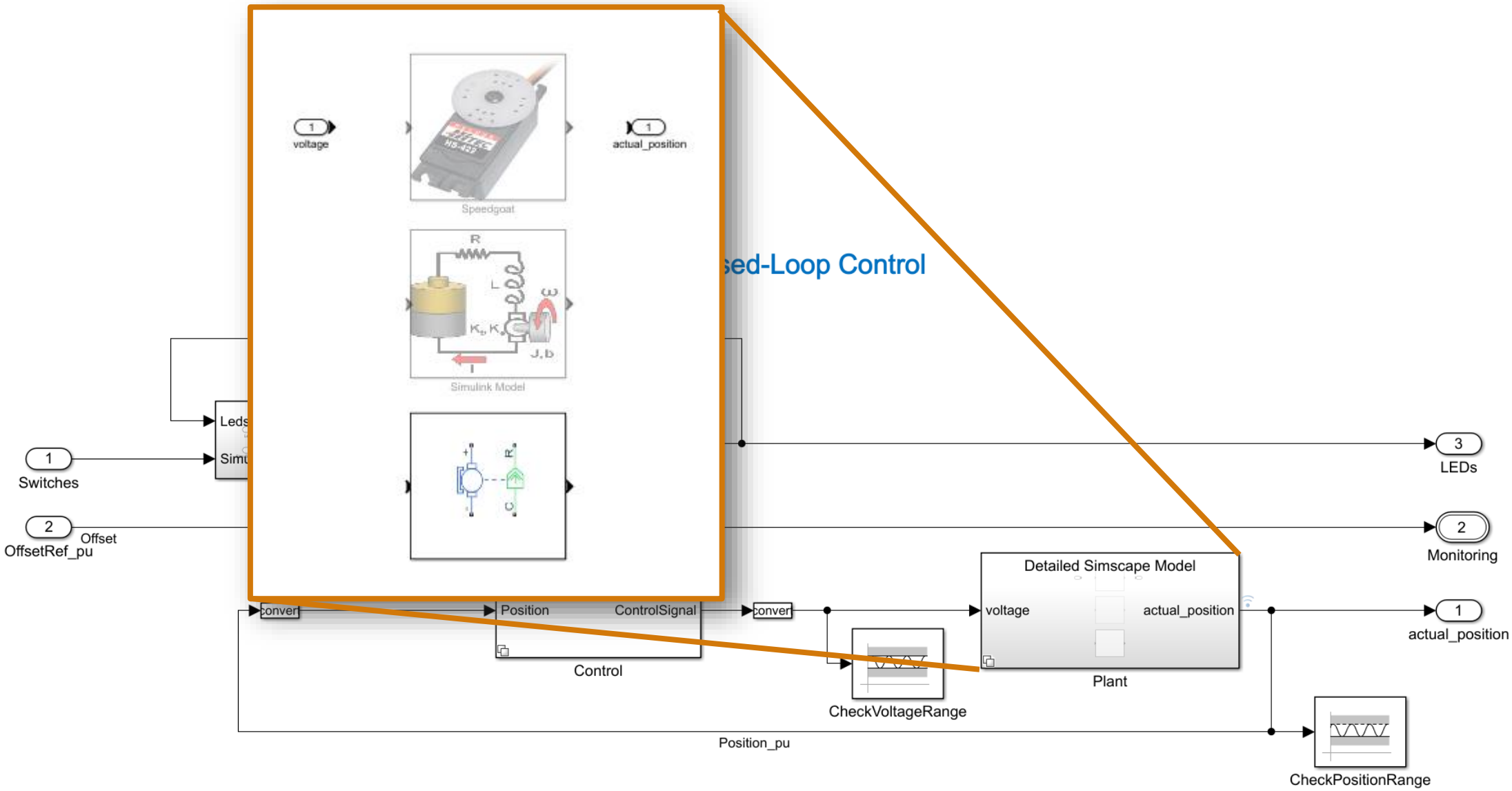
Command/Leds states

Position/Switches states

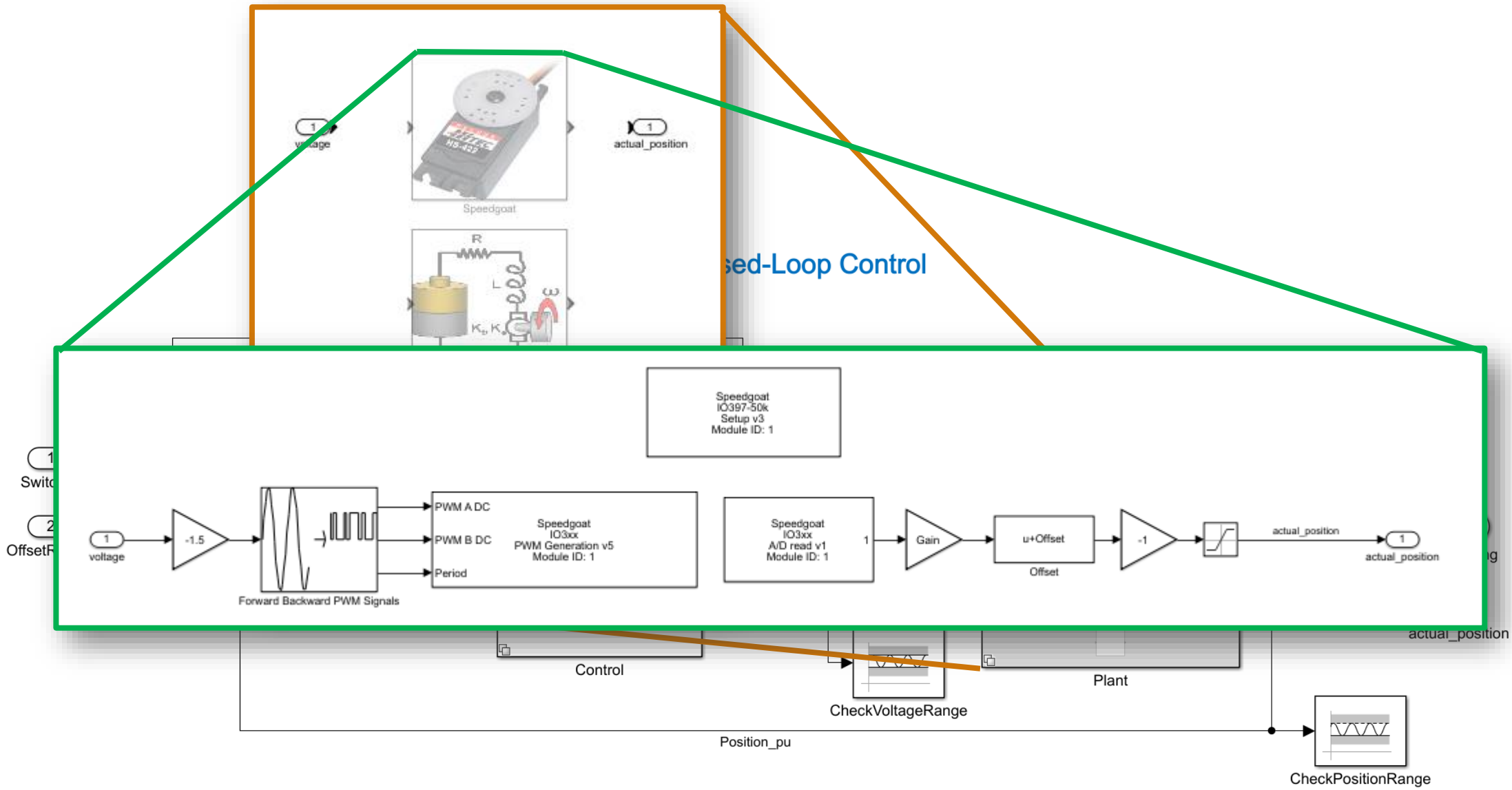
Plant Prototype



Rapid Control Prototyping

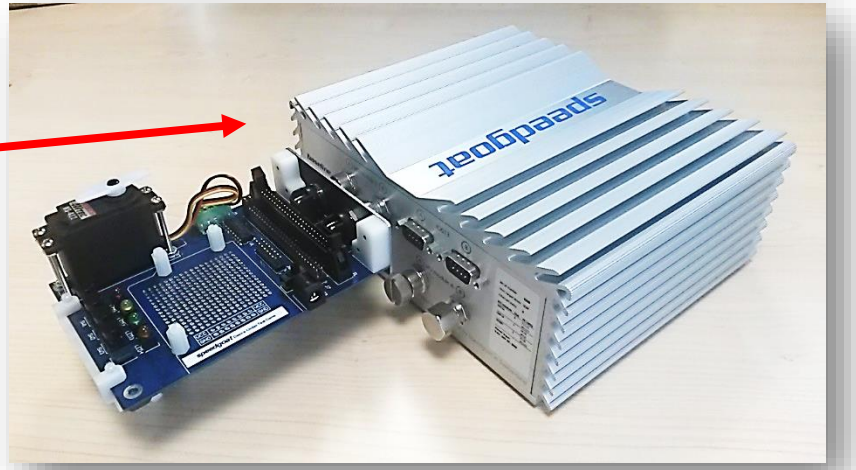


Rapid Control Prototyping



Rapid Control Prototyping

Compile, Load, Execute



Test Harness: Manual_TestHarness - Simulink

File Edit View Display Diagram Simulation Analysis Code Tools Help

Build Model

Manual_TestHarness

Manual Test Harness

Position

led1 led2 led3 led4

Down Up Update -- Auto

Increment Request

Position_Offset:Value

RCP

Configuration Selection

User Manual:

> Click Down/Up button to set the number of step increment
> Click Update to Move servo to the desired step increment.
> Double click Update to Reset servo to offset value
> Click Auto Button to start or stop Auto move

Control, Monitor & Log in Real Time

Target Scopes

Switches Position(pu) SupplyVoltage(V) Display

dut_closed_loop_Switch

Switches LEDs Offset SimulatedUI

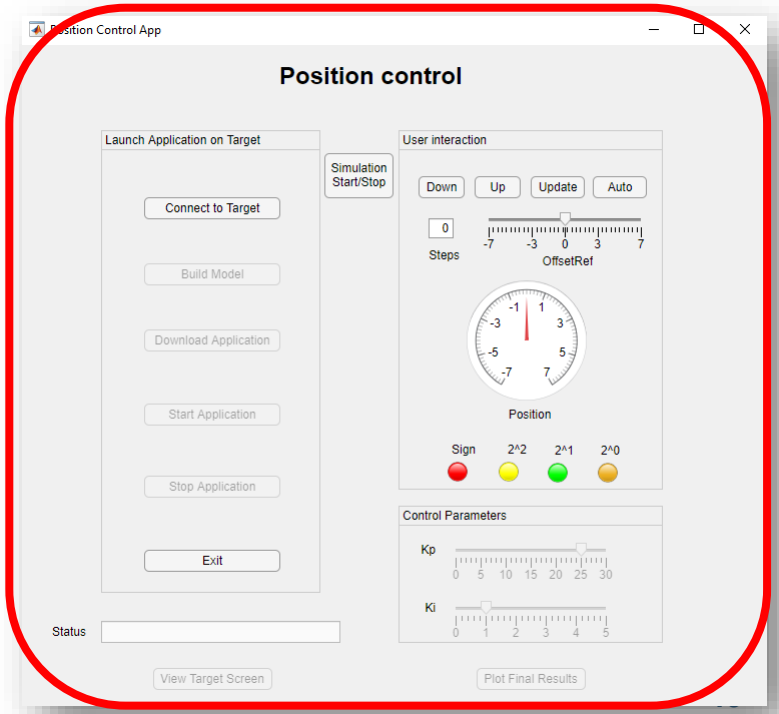
Switches actual_position Monitoring OffsetRef_pu LEDs System Under Test

<Position_pu> <Command> <Display> <References> Ref

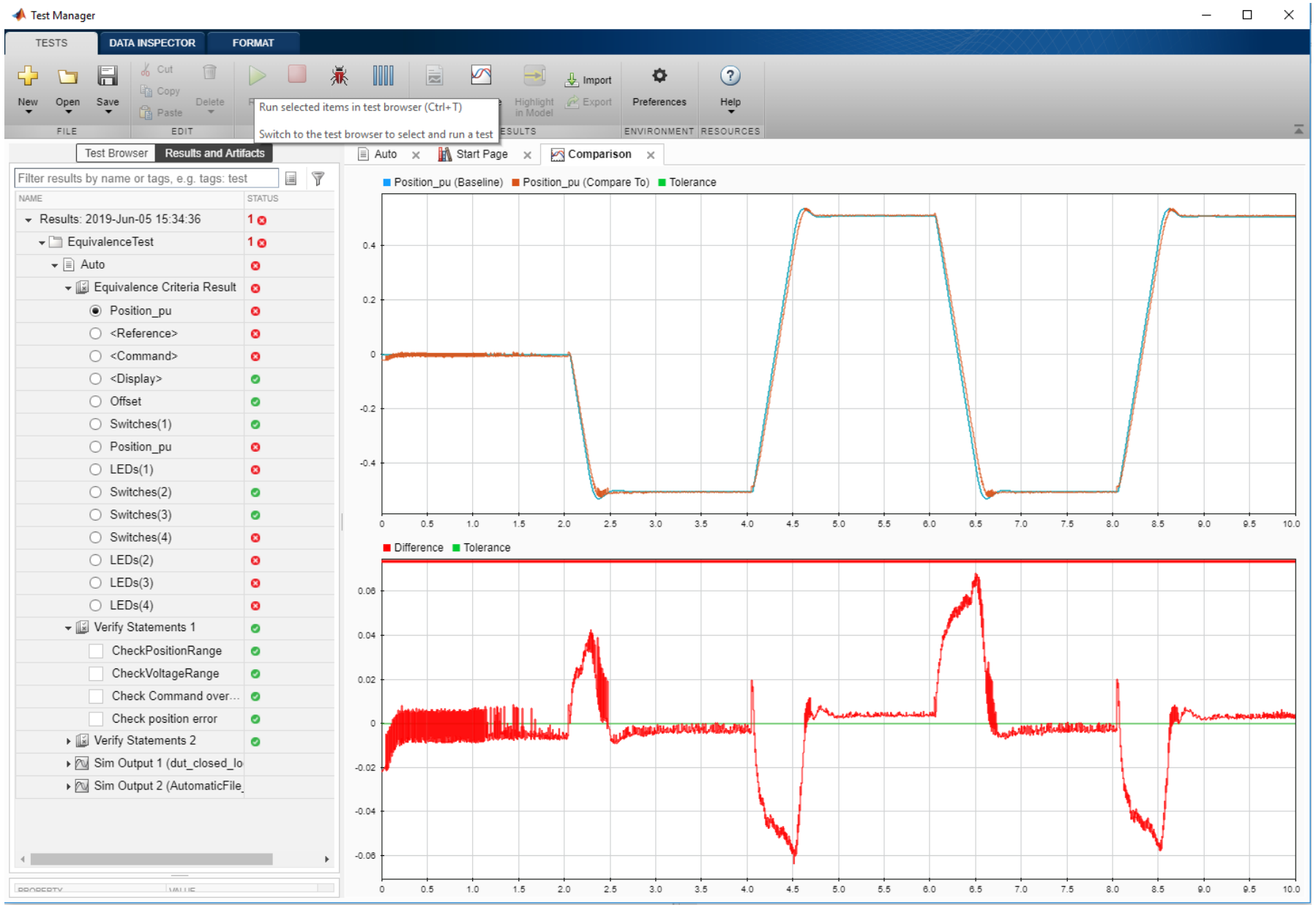
Position

Ready 83% FixedStepAuto

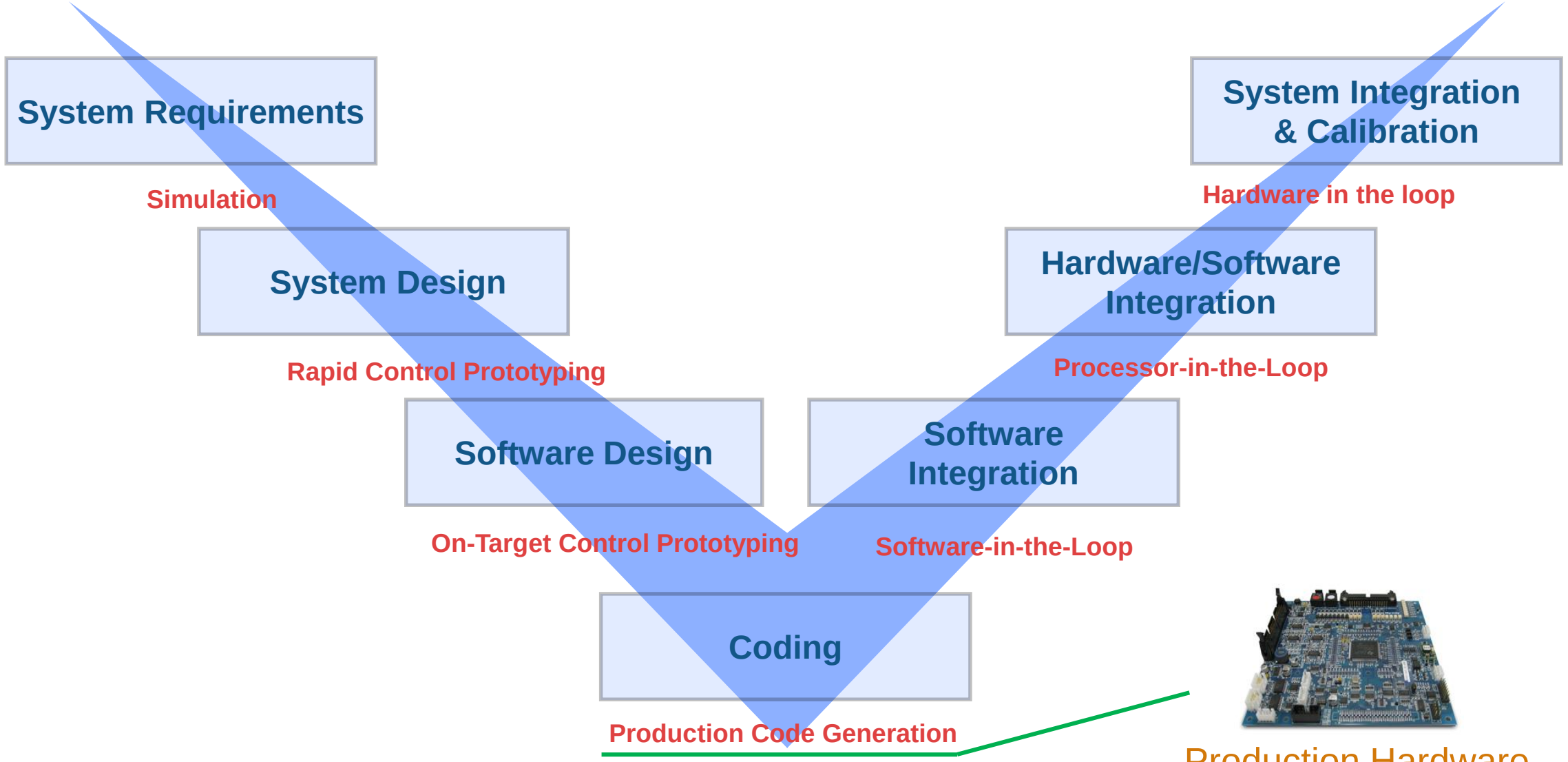
App designer



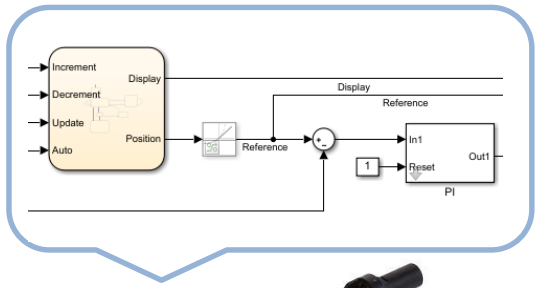
Rapid Control Prototyping - Run & Analyse tests

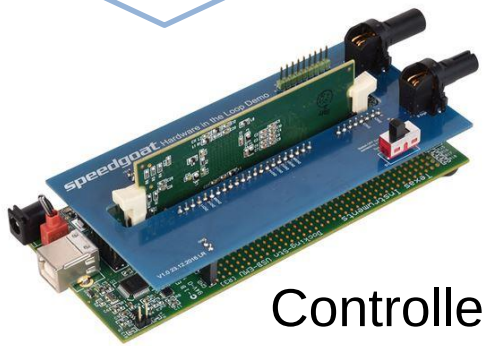


Production Code



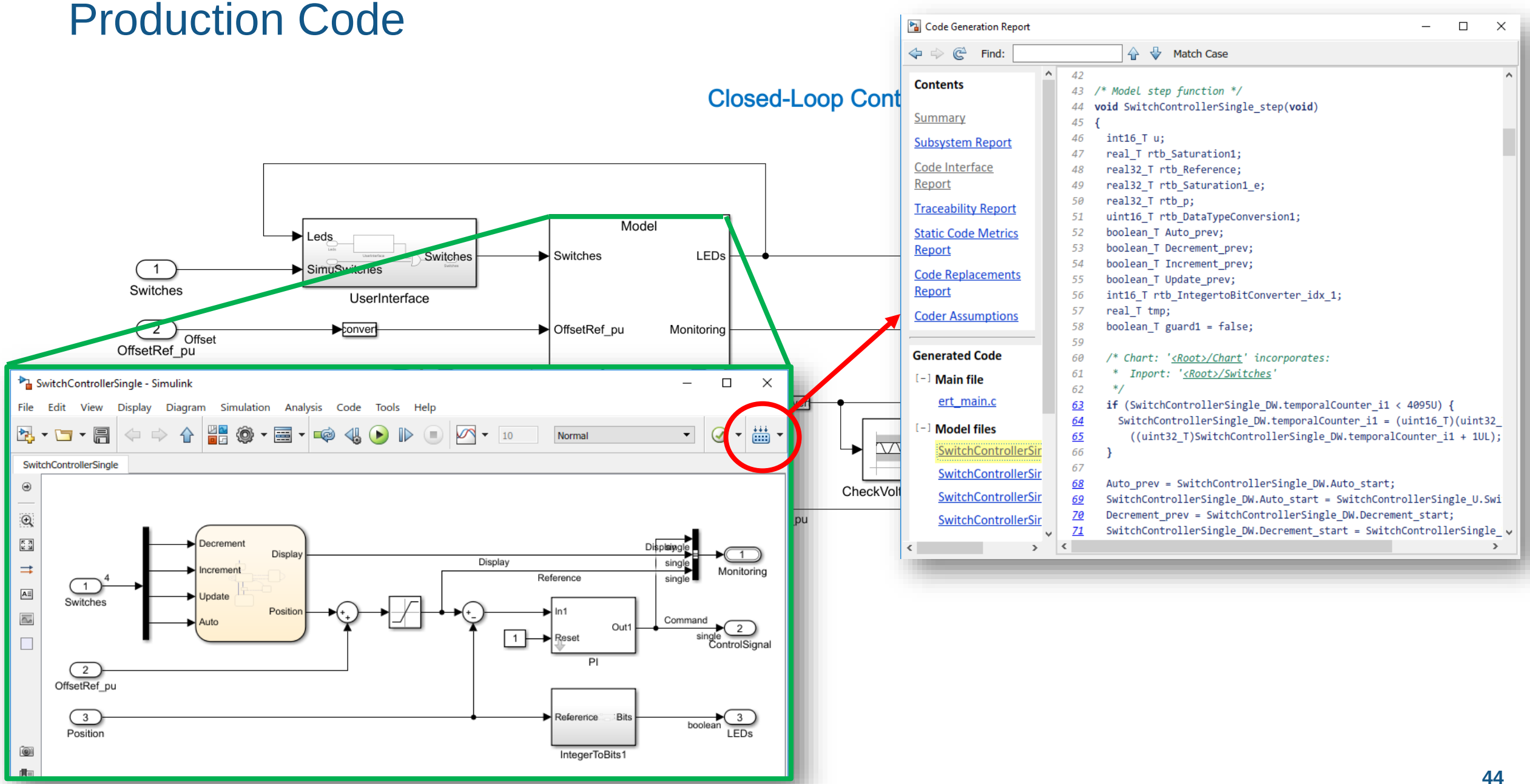
Production Code





Controller

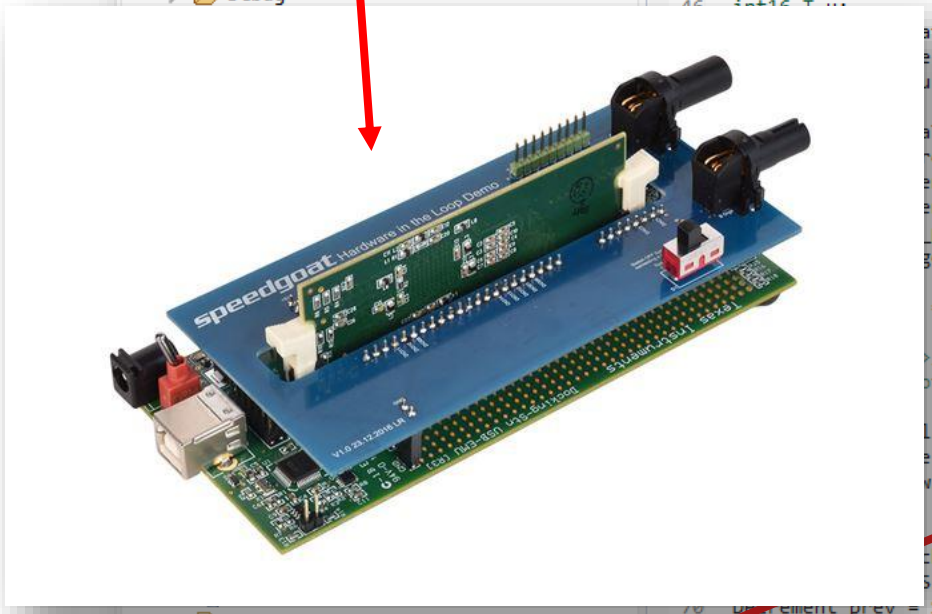
Production Code



Production Code

Compile & Load to target from IDE

Target IDE



workspace_v8 - SwitchControllerSingle/SwitchControllerSingle.c - Code Composer Studio

File Edit View Navigate Project Run Scripts Window Help

Project Explorer

SwitchControllerSingle [Active - Debug]

Includes

Debug

rtGetInf.c

rtGetNaN.c

SwitchControllerSingle.c

TexasControlBoard

SwitchControllerSingle.c

```
43 /* Model step function */
44 void SwitchControllerSingle_step(void)
45 {
ation1;
erence;
uration1_e;

aTypeConversion1;
rev;
ent_prev;
ent_prev;
_prev;
gertoBitConverter_idx_1;

= false;

>/Chart' incorporates:
ot>/Switches'

llerSingle_DW.temporalCounter_i1 < 4095U) {
rSingle_DW.temporalCounter_i1 = (uint16_T)(uint32
itchControllerSingle_DW.temporalCounter_i1 + 1UL)

chControllerSingle_DW.Auto_start;
Single_DW.Auto_start = SwitchControllerSingle_U.Switches[3];
Decrement_prev = SwitchControllerSingle_DW.Decrement_start;
SwitchControllerSingle_DW.Decrement_start = SwitchControllerSingle_U.Switches
  [0];
Increment_prev = SwitchControllerSingle_DW.Increment_start;
SwitchControllerSingle_DW.Increment_start = SwitchControllerSingle_U.Switches
  [1];
Update_prev = SwitchControllerSingle_DW.Update_start;
SwitchControllerSingle_DW.Update_start = SwitchControllerSingle_U.Switches[2];
```

Code Generation Report

Find: Match Case

Contents

Summary

Subsystem Report

Code Interface Report

Traceability Report

Static Code Metrics Report

Code Replacements Report

Coder Assumptions

Generated Code

[-] Main file

ert_main.c

[-] Model files

SwitchControllerSingle.c

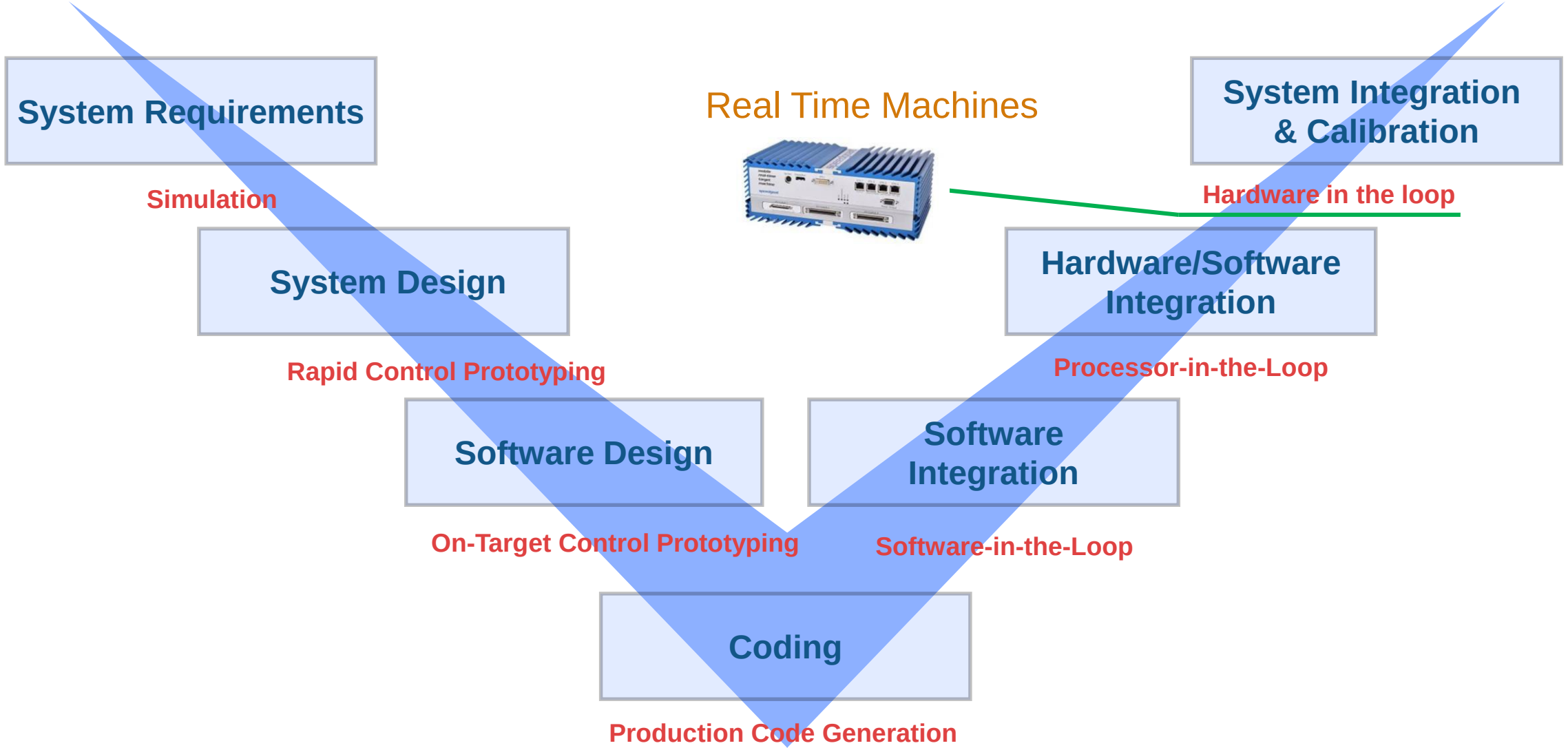
SwitchControllerSingle.c

SwitchControllerSingle.c

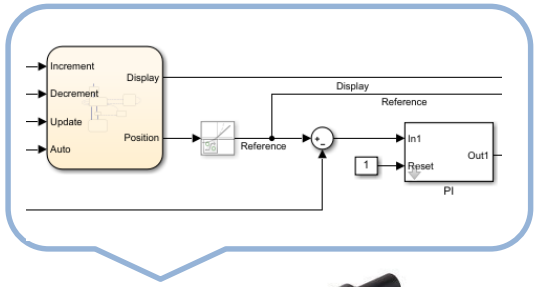
SwitchControllerSingle.c

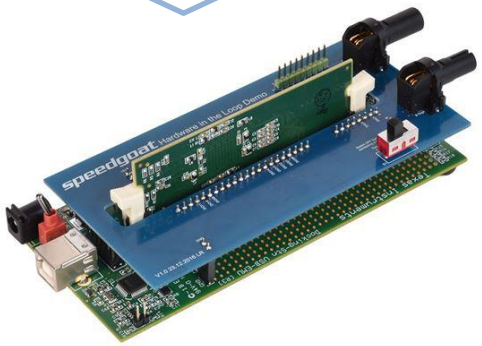
```
42
43 /* Model step function */
44 void SwitchControllerSingle_step(void)
45 {
46   int16_T u;
47   real_T rtb_Saturation1;
48   real32_T rtb_Reference;
49   real32_T rtb_Saturation1_e;
50   real32_T rtb_p;
51   uint16_T rtb_DataTypeConversion1;
52   boolean_T Auto_prev;
53   boolean_T Decrement_prev;
54   boolean_T Increment_prev;
55   boolean_T Update_prev;
56   int16_T rtb_IntegertoBitConverter_idx_1;
57   real_T tmp;
58   boolean_T guard1 = false;
59
60   /* Chart: '<Root>/Chart' incorporates:
61    *  Inport: '<Root>/Switches'
62    */
63   if (SwitchControllerSingle_DW.temporalCounter_i1 < 4095U) {
64     SwitchControllerSingle_DW.temporalCounter_i1 = (uint16_T)(uint32_
65       ((uint32_T)SwitchControllerSingle_DW.temporalCounter_i1 + 1UL);
66   }
67
68   Auto_prev = SwitchControllerSingle_DW.Auto_start;
69   SwitchControllerSingle_DW.Auto_start = SwitchControllerSingle_U.Swi
70   Decrement_prev = SwitchControllerSingle_DW.Decrement_start;
71   SwitchControllerSingle_DW.Decrement_start = SwitchControllerSingle_
```

Hardware in The Loop



Hardware in The Loop



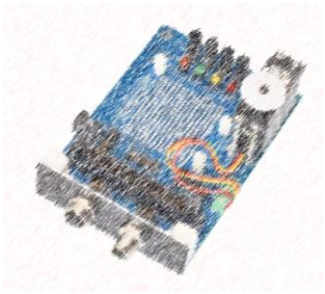


Controller

Command/Leds states

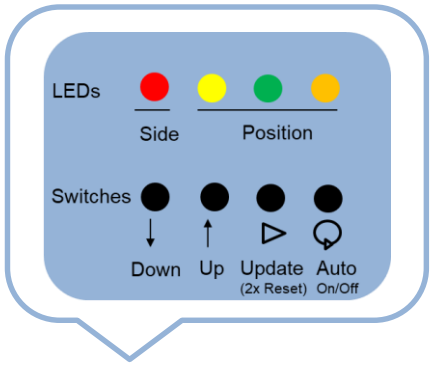
Position/Switches states

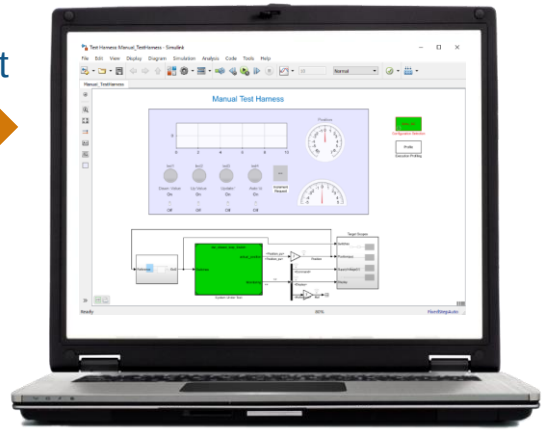
Simulated Plant



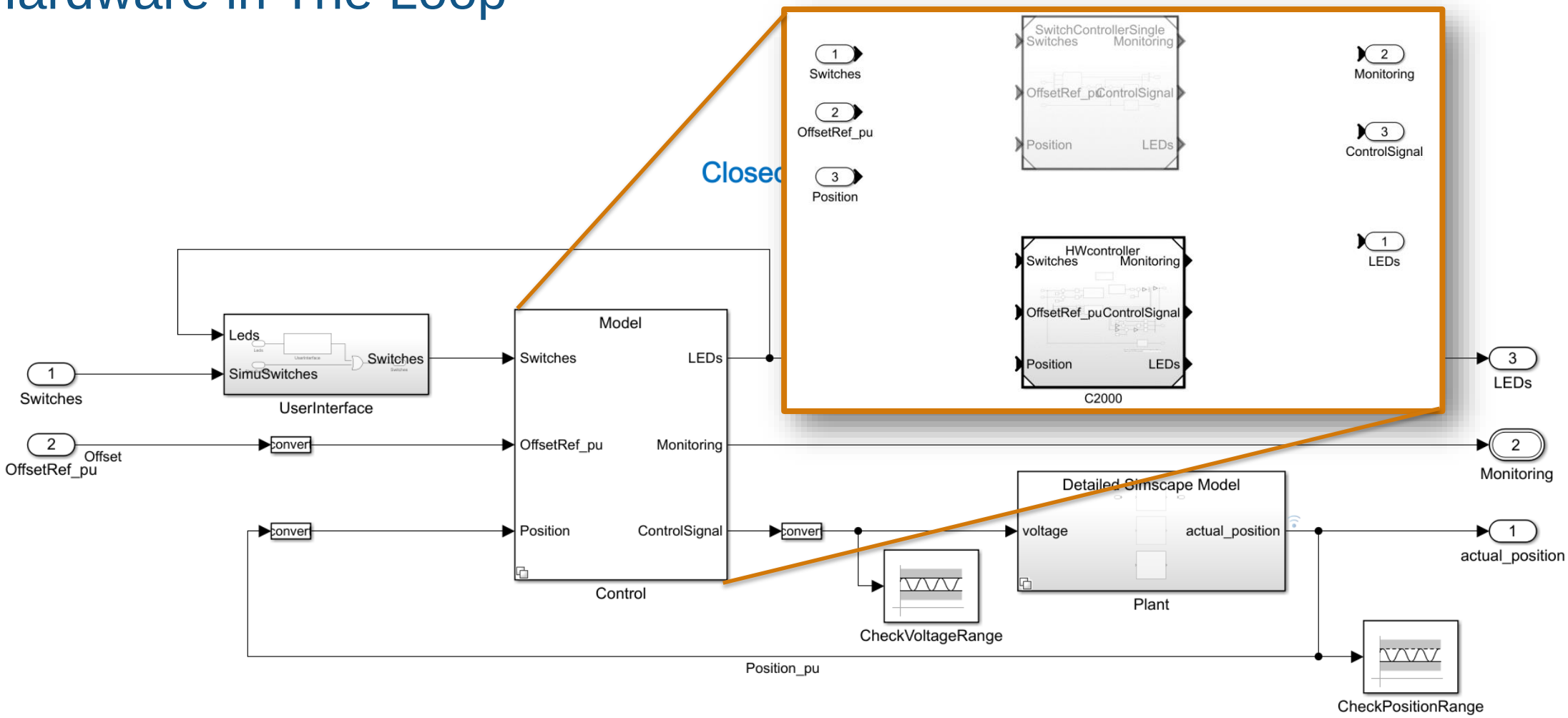
Ethernet

User Interface

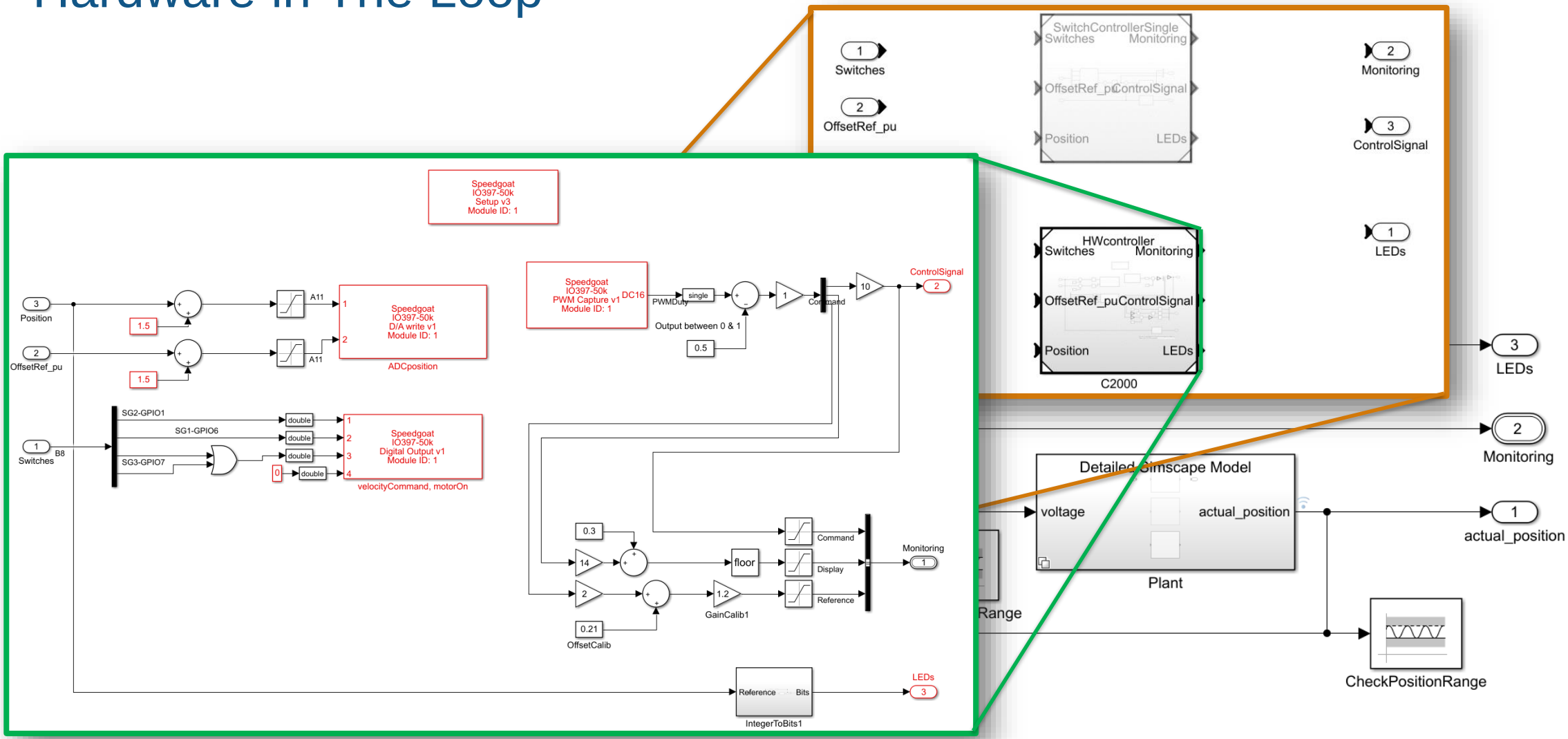




Hardware in The Loop

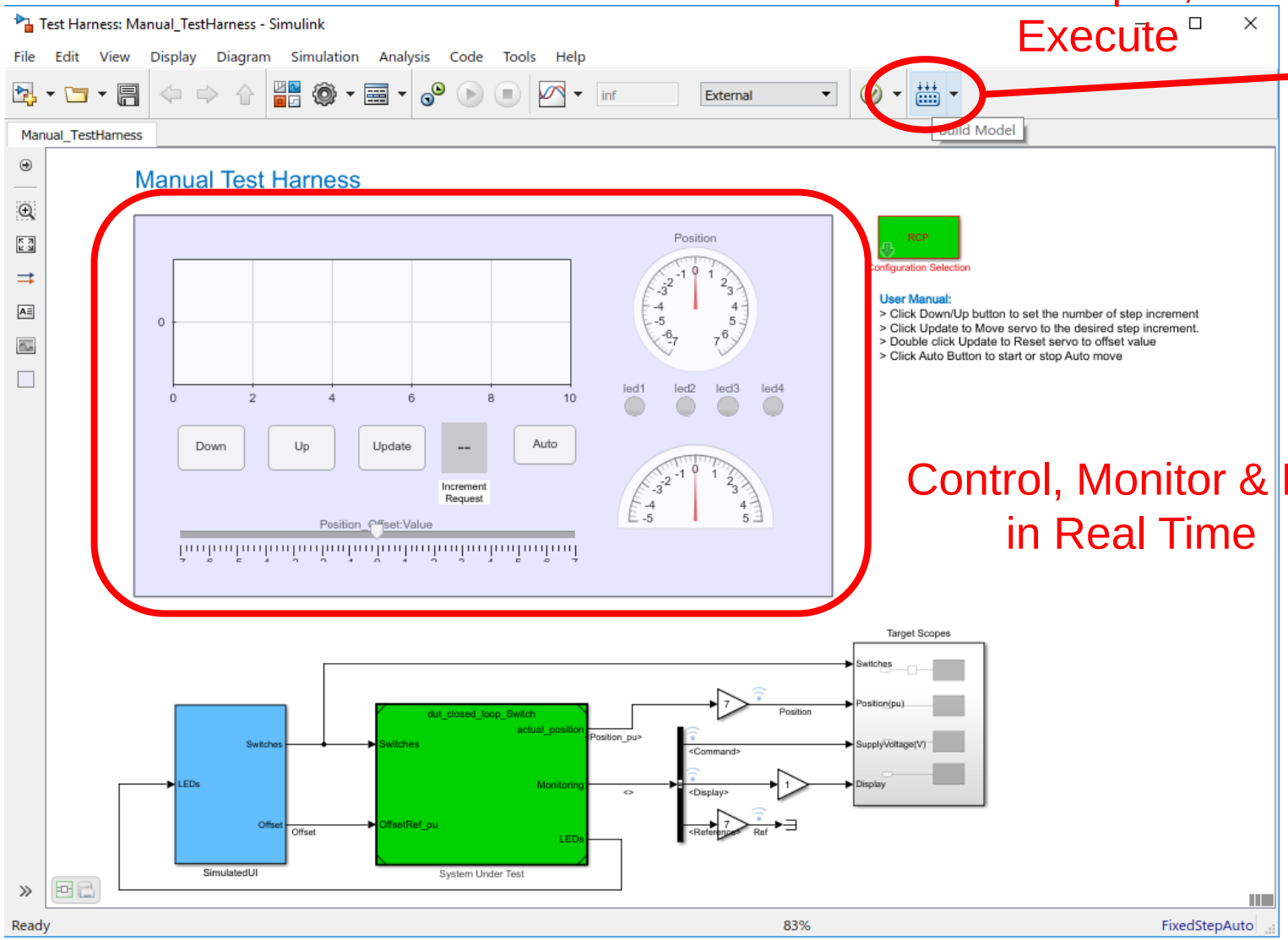


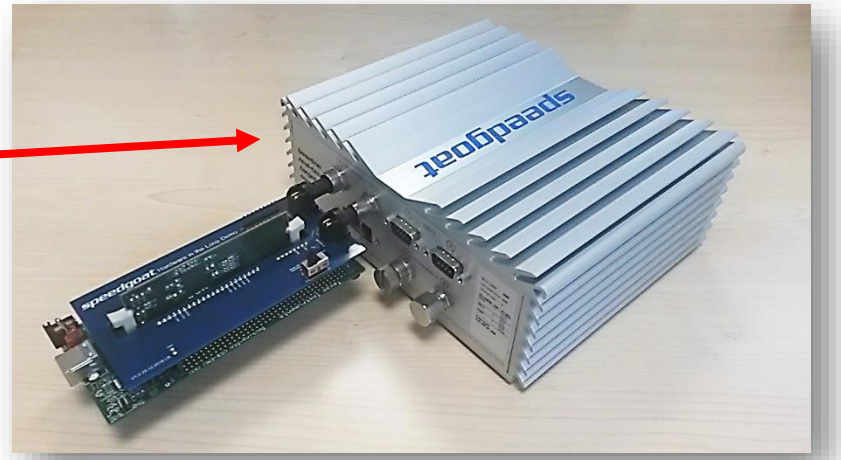
Hardware in The Loop



Hardware in The Loop

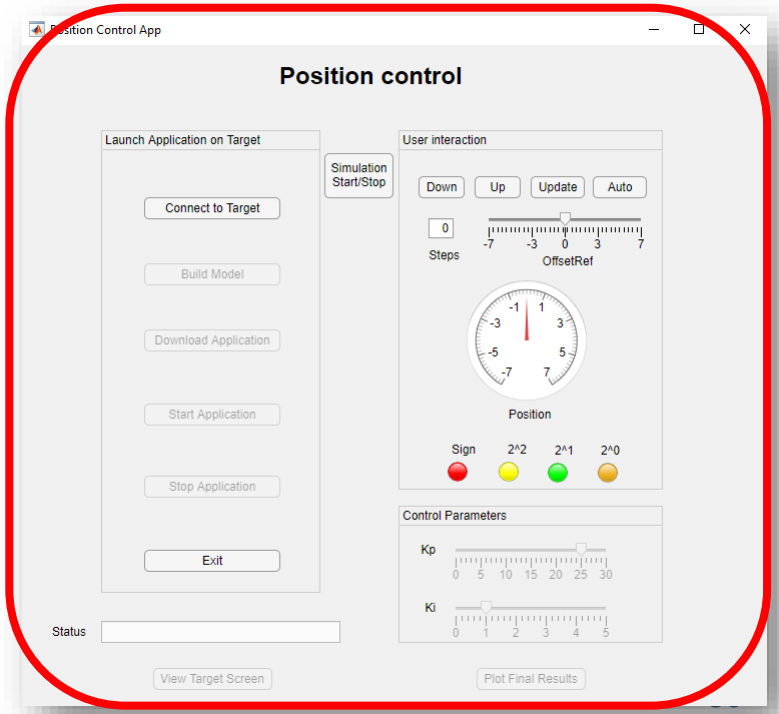
Compile, Load, Execute



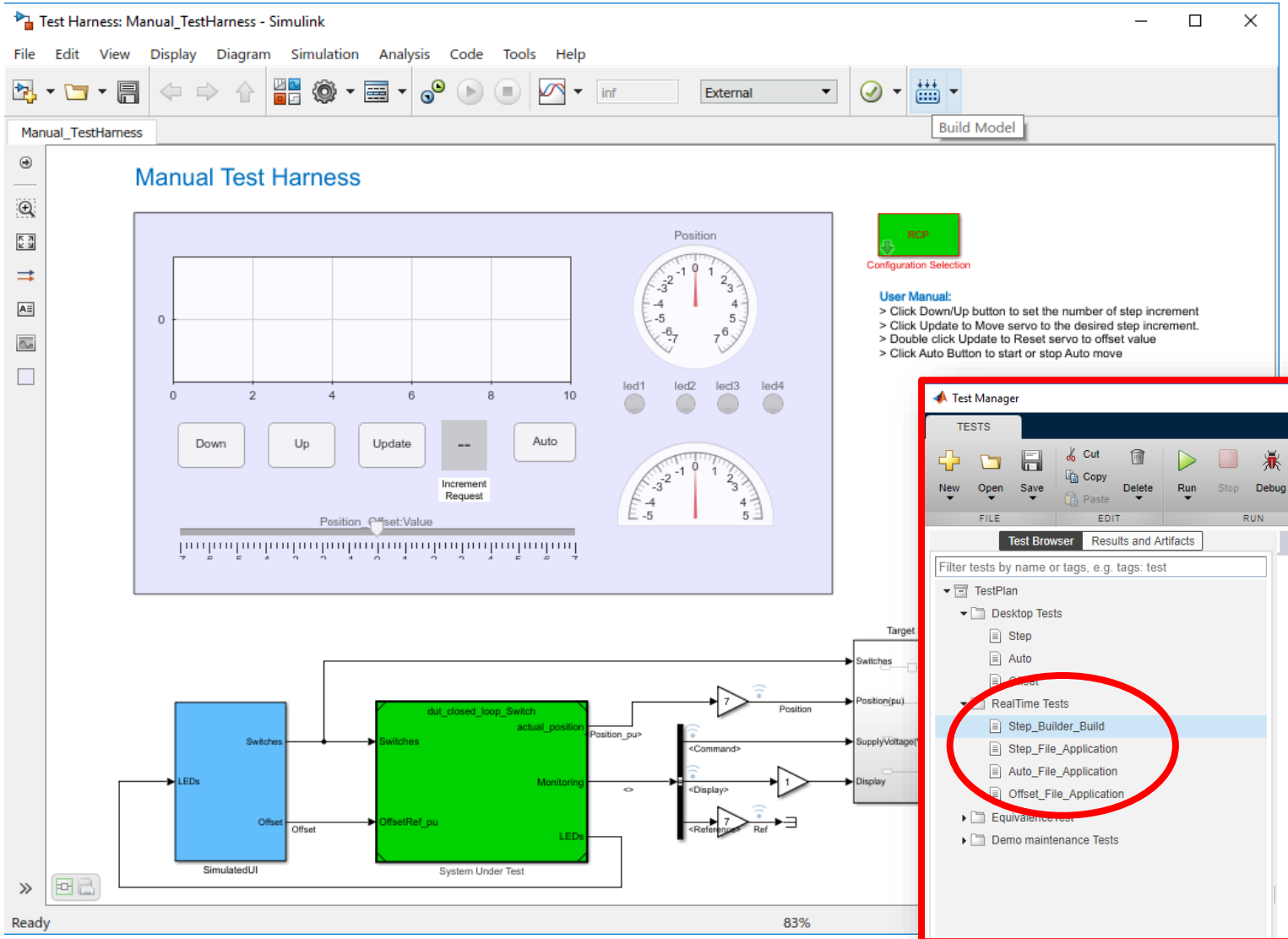


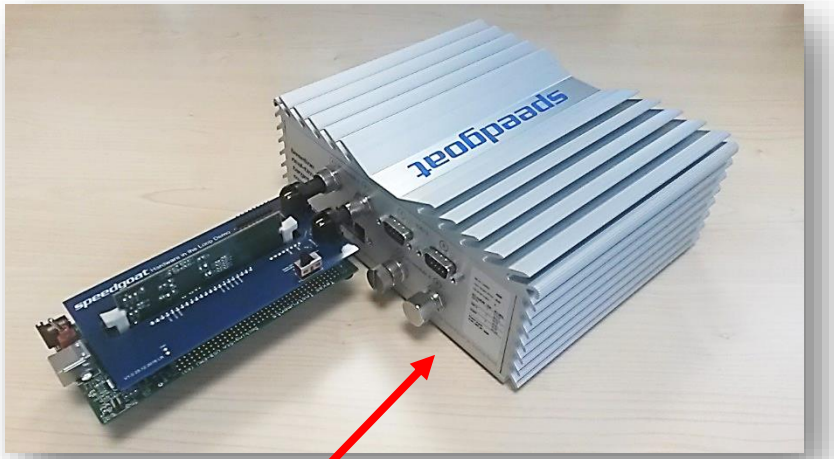
Control, Monitor & Log in Real Time

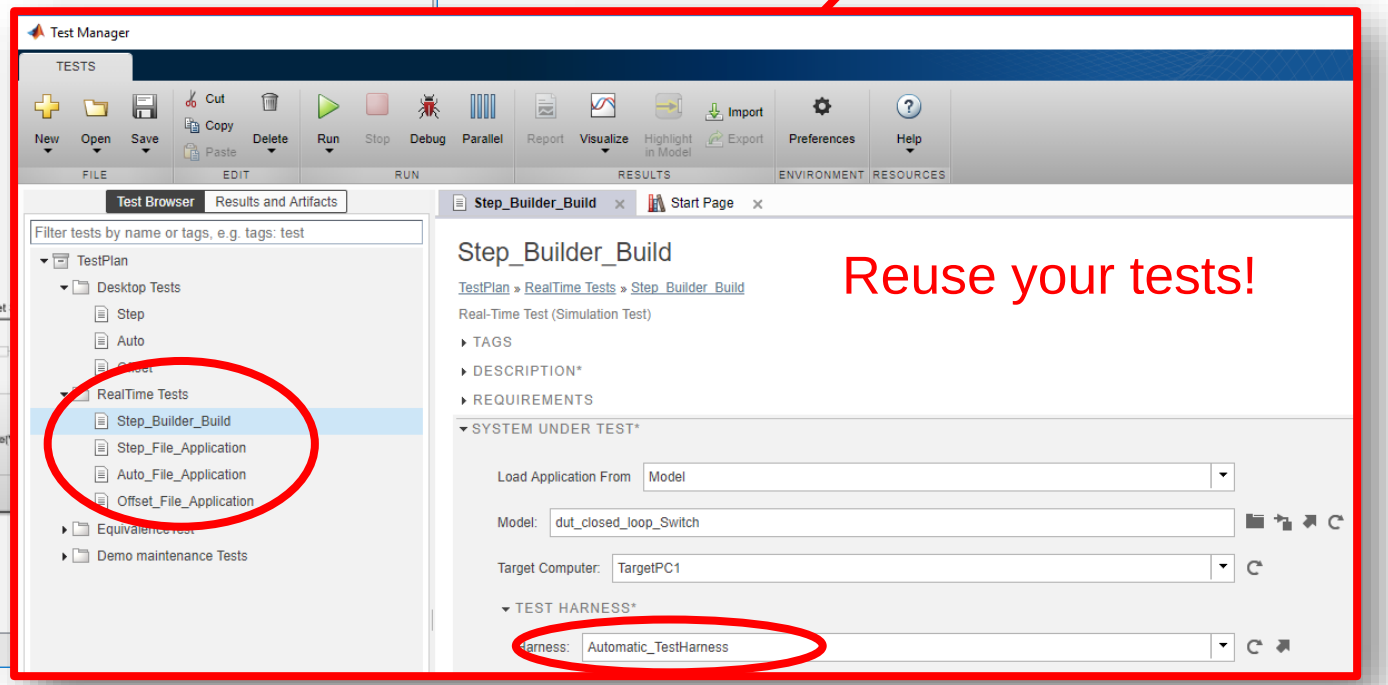
App designer



Hardware in The Loop







Conclusions

- Hardware deployment use cases
  - Control Prototyping
  - Production code
  - Hardware verification
- Fully integrated solutions
  - Simulink Real Time & Speedgoat Real-Time machines
  - Hardware support packages
  - From specification to Hardware verification in a single environment
- Retrieve your time to do **more & better innovative** products

What's next…

- Contact us to…
  - Define together your MBD implementation plan
  - Build together your project and model architecture
  - Train yourselve & your team

- More informations online…
  - Embedded systems page
  - Simulink Real Time & Speedgoat Solutions
  - Supported Hardware page