Lightning Talks

ACCU 2012

Tom Gilb – Quantifying Music

Dominic Robinson – The Beard Heuristic

Jim Hague – Setting up an ACCU local group

Claudius Link – Complexity: Human Behaviour in Complex Situations

Erik Schlyter – Teenage Mutant Niinja Turtles Pattern

Diomidis Spinellis – name !shame: Rational Naming

Anders Schau Knatten – AUTOMATE ALL THE THINGS

Andy Balaam – Implementing Tail-call Optimisation in C++

Klaus Marquardt – Learning From School

Ed Sykes & Raj Singh – Posse Programming

Bernhard Merkle – I Use A Dead Language

I use a dead language…

Bernhard Merkle

ACCU 2012

Lightning ~~talk~~

Disclaimer:

- I do embedded development and <u>still</u> feel the metal

embedded software development is hard… ;-(

I work for

SICK

sensors

I work for

SICK

sensors

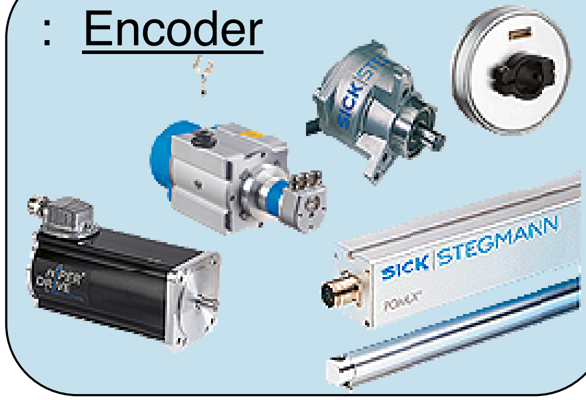
YOU: 'Eh...??? (!@#$ ^)'

SICK Sensors...

: Industrial Sensors

: Advanced Industrial Sensors

: Encoder

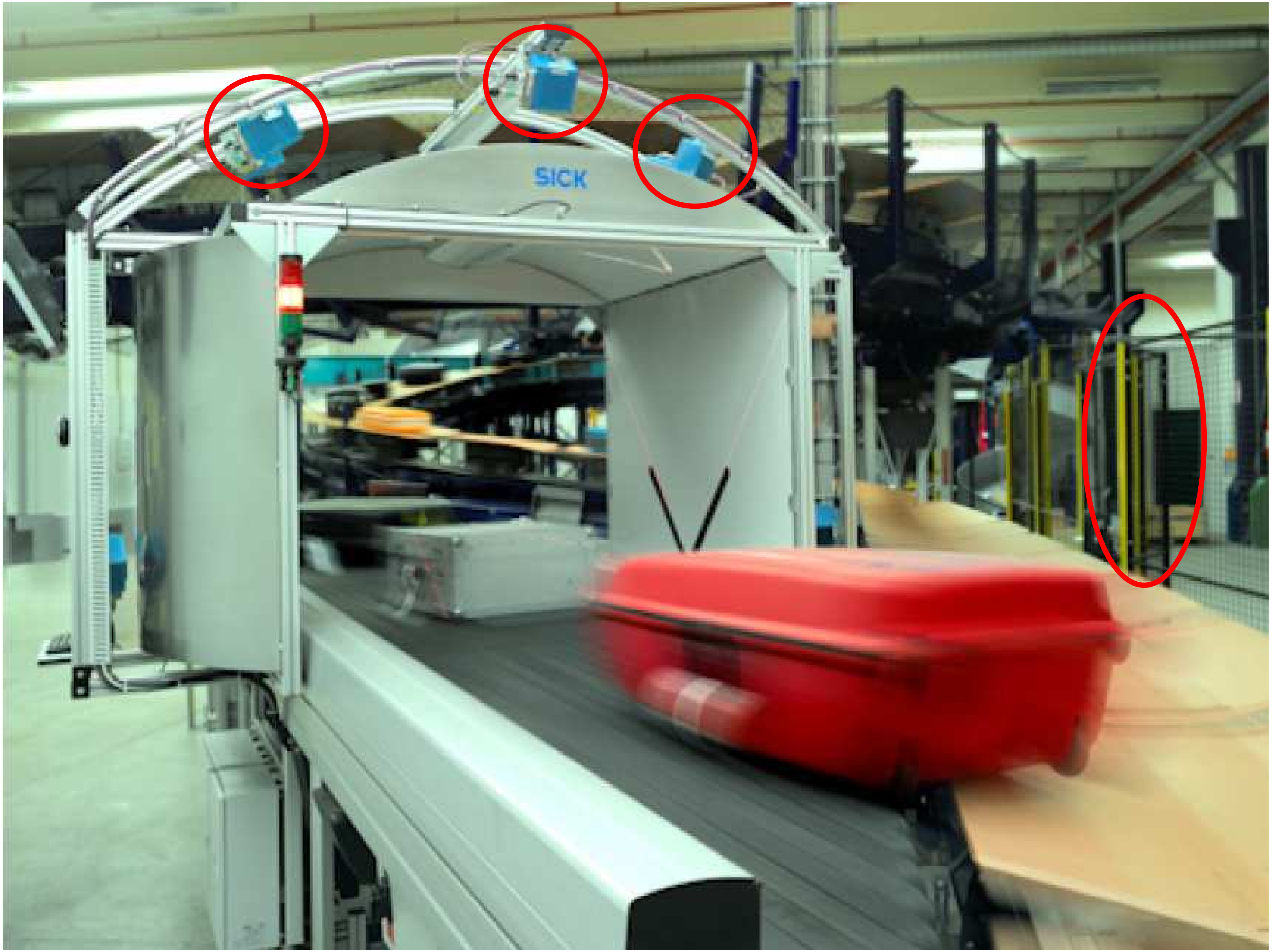


: Industrial Safety Systems

: Auto Ident

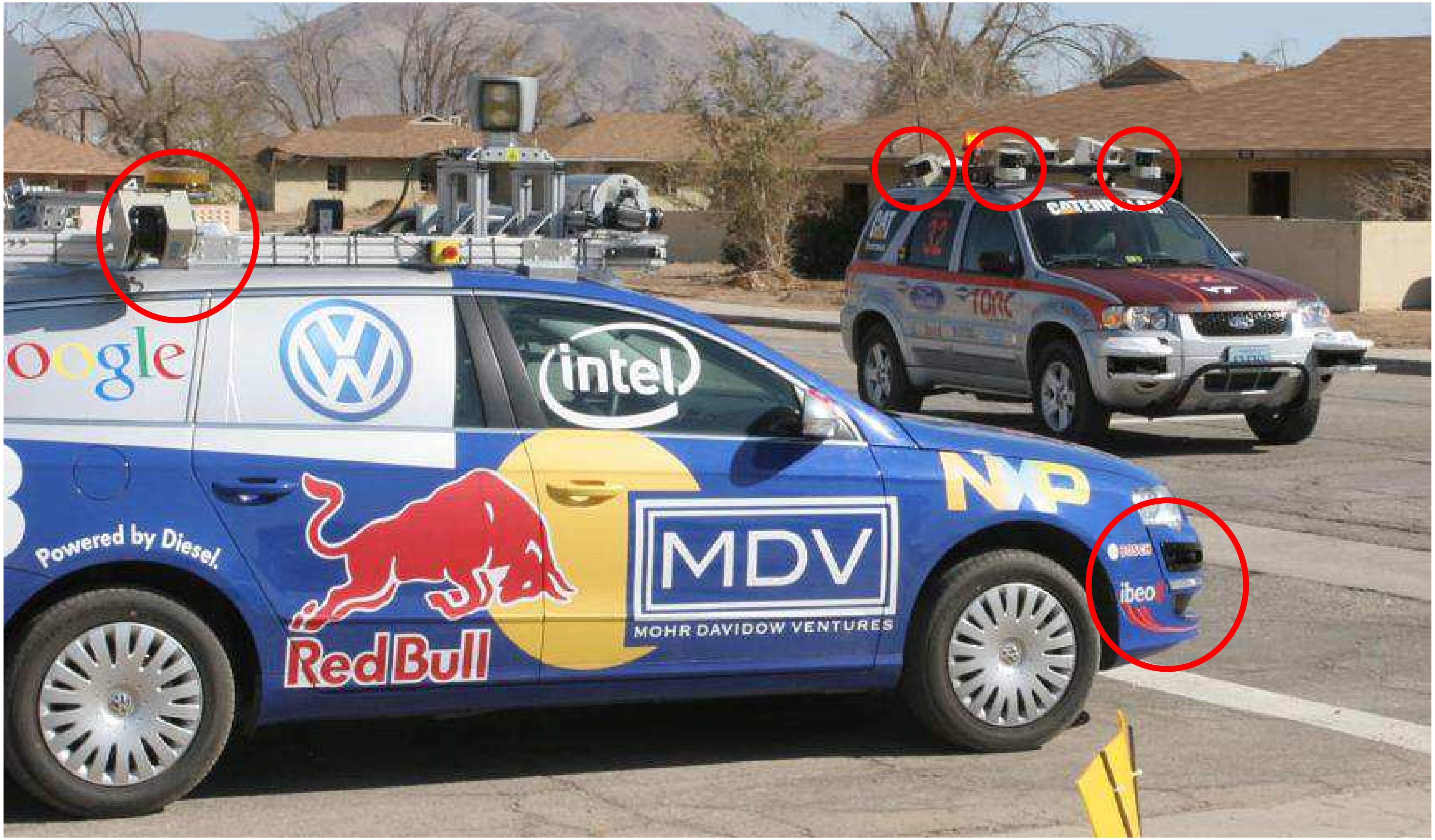
: Analyzers & Process Instrumentation

Lufthansa
STAR ALLIANCE
Lufthansa
Quick Check-in
Lufthansa
Quick Check-in
Lufthansa
Quick Check-in

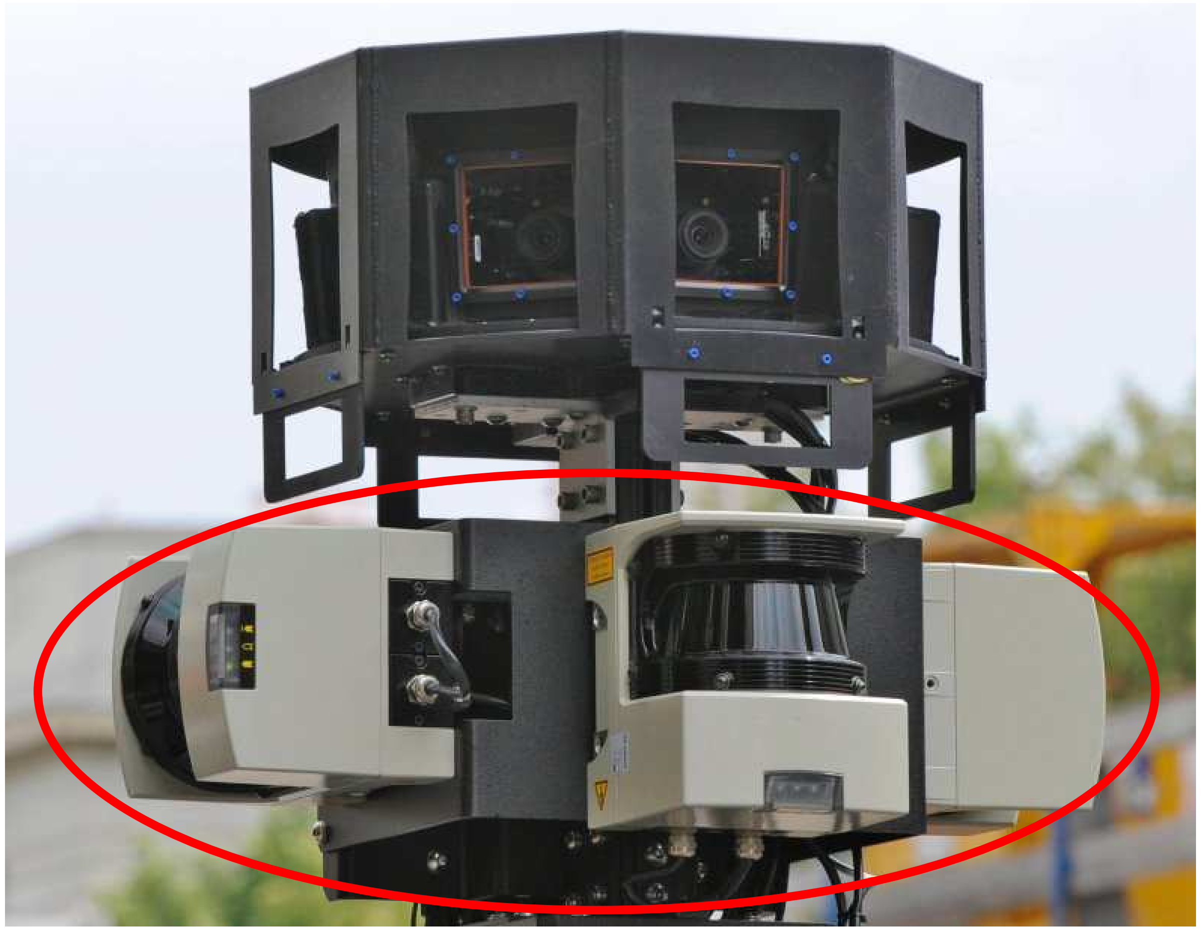

SICK

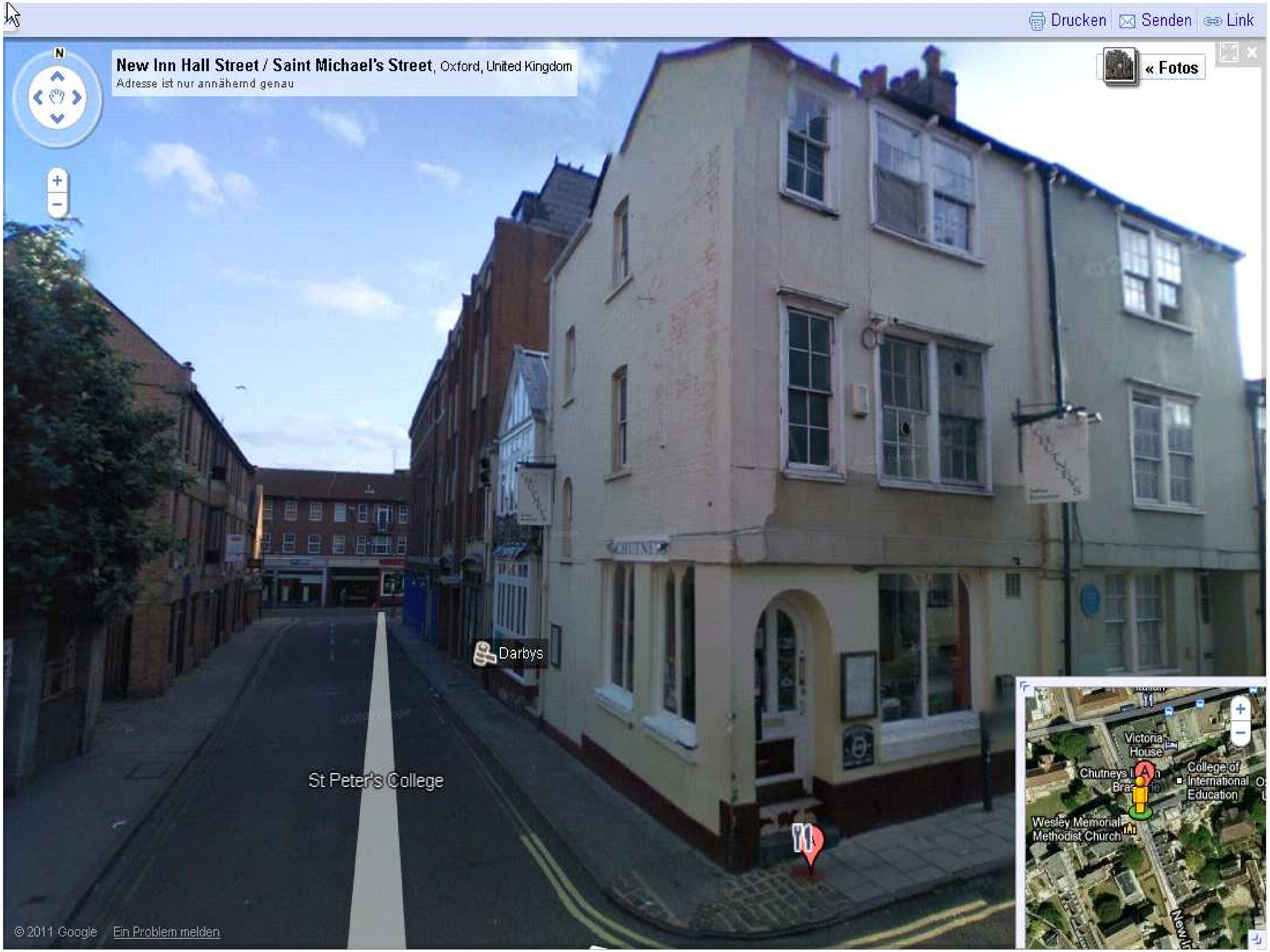
DARPA: Urban/Grand Challenge

SICK



Google Street View




Drucken
Senden
Link
New Inn Hall Street / Saint Michael's Street, Oxford, United Kingdom
Adresse ist nur annähernd genau
« Fotos
Darbys
St Peter's College
© 2011 Google
Ein Problem melden
Victoria House
Chutneys
College of International Education
Wesley Memorial Methodist Church

embedded
software
development

is <u>still</u> hard…

;-(

Challenges:

Abstraction without Runtime Cost

C considered unsafe

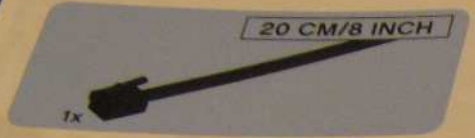
(High Level) Program Annotations

Static Checks
+
Verification

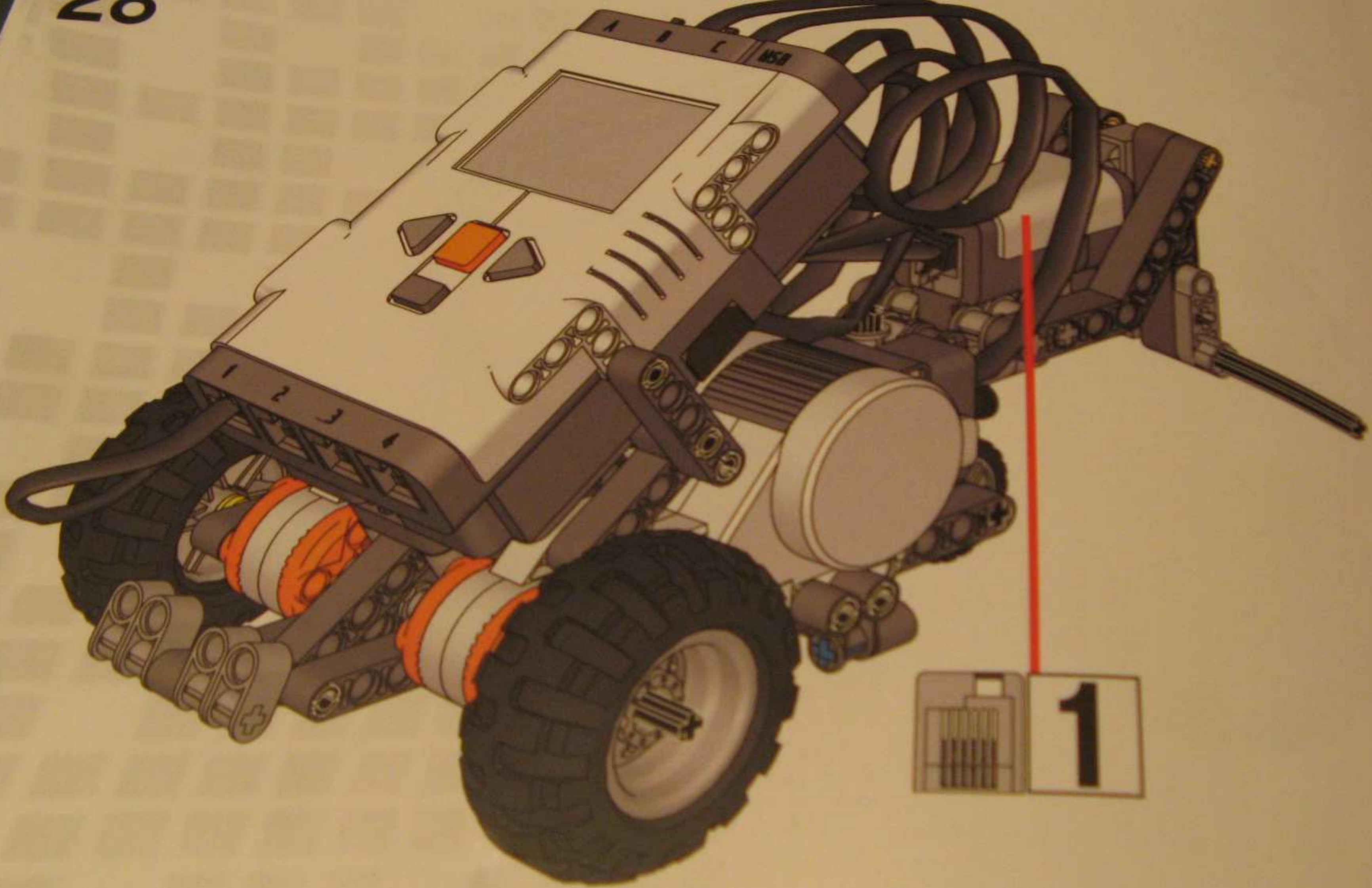
Product Lines Support

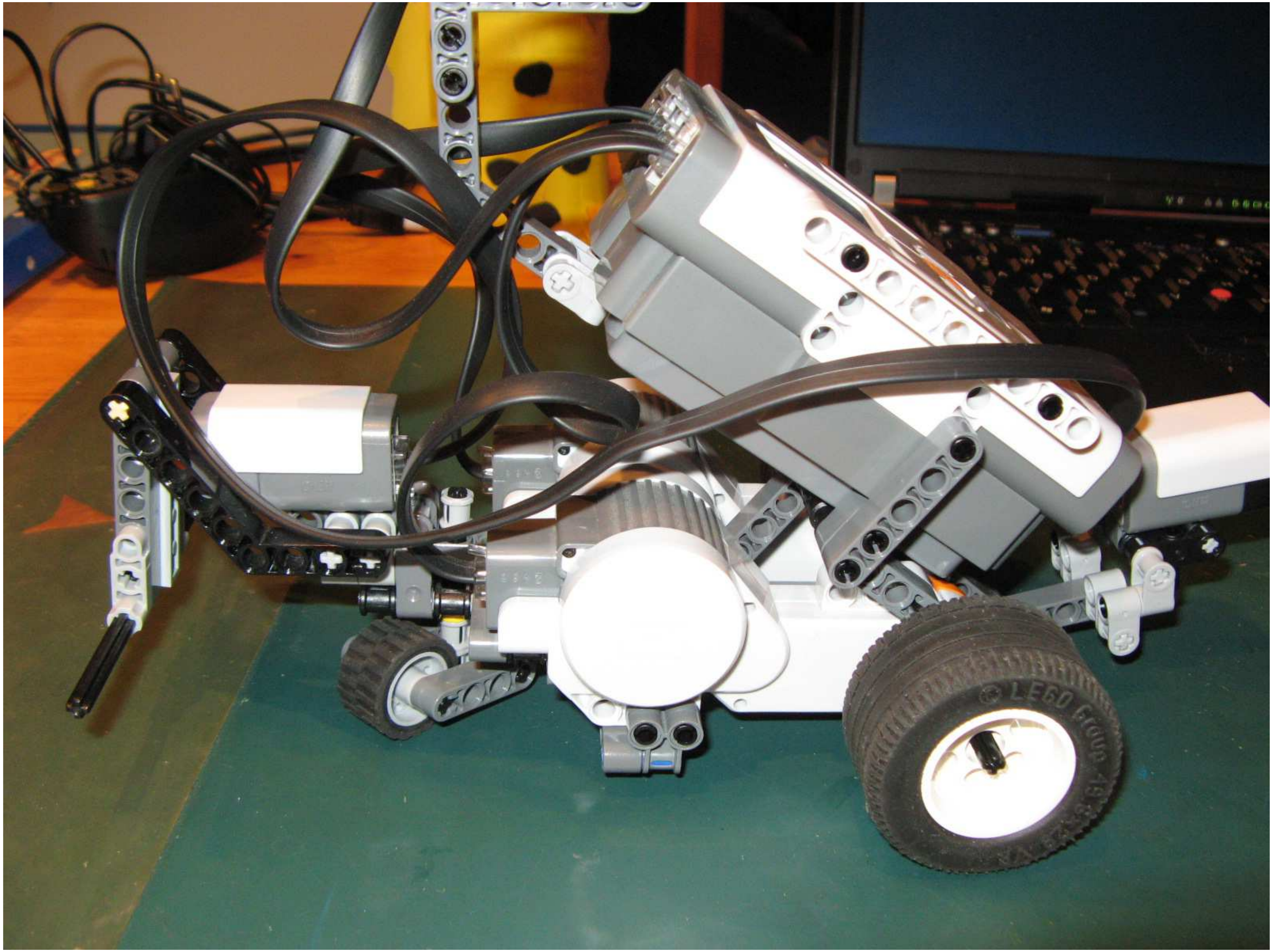
(variants without #ifdef)

but lets have some fun...



28





I have stolen the NXT robot from...

(my son)...

[redacted]

What if…

What if…

you could change languages

What if…

you could change languages

like you can change programs?

What if…

What if…

you could
<u>use</u>
the DSL
YOU want ?

What if…

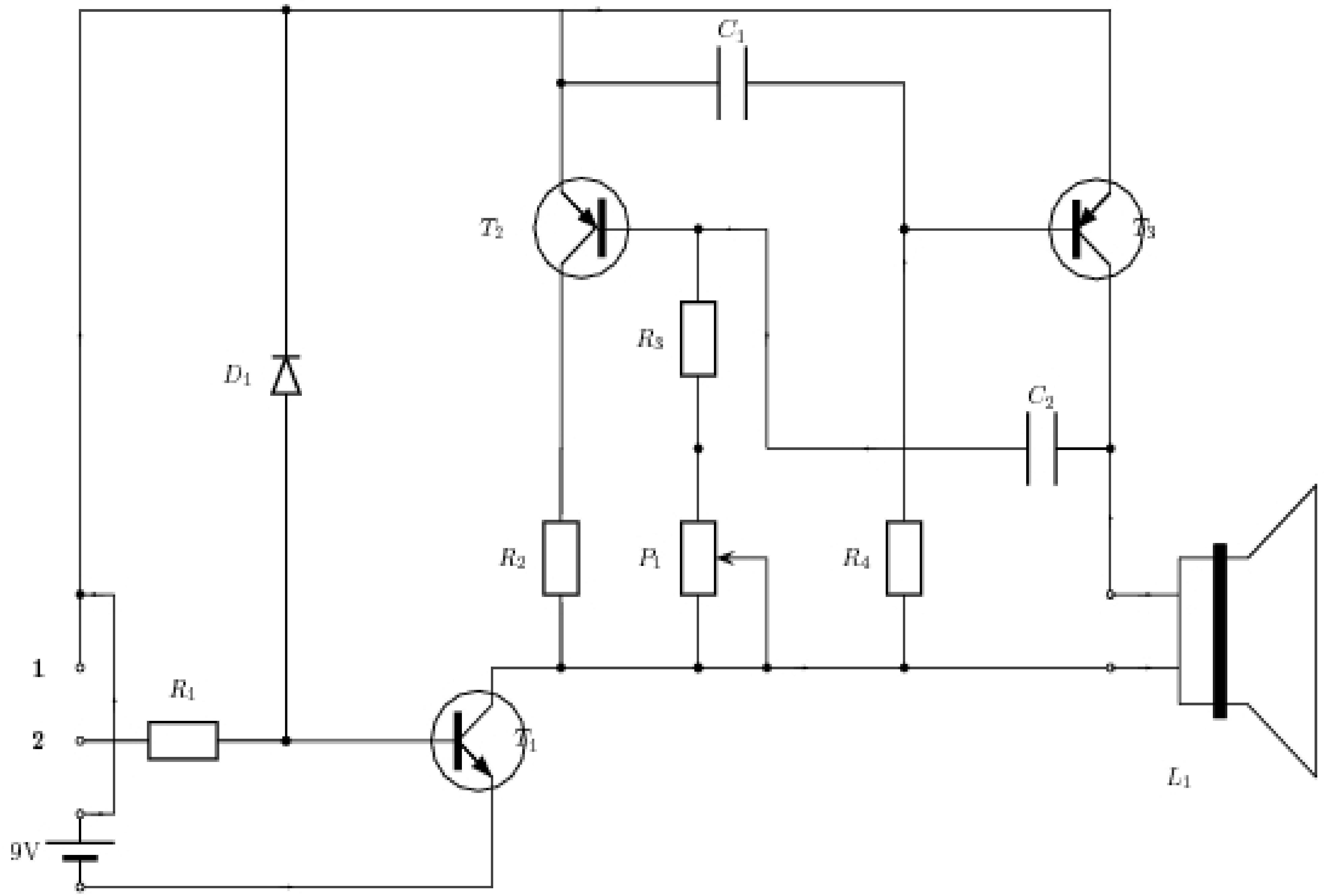
you could
use
the DSL
YOU want ?

(within your GPL)

SELECT firstname, lastName **from** employee **where** age = 42;

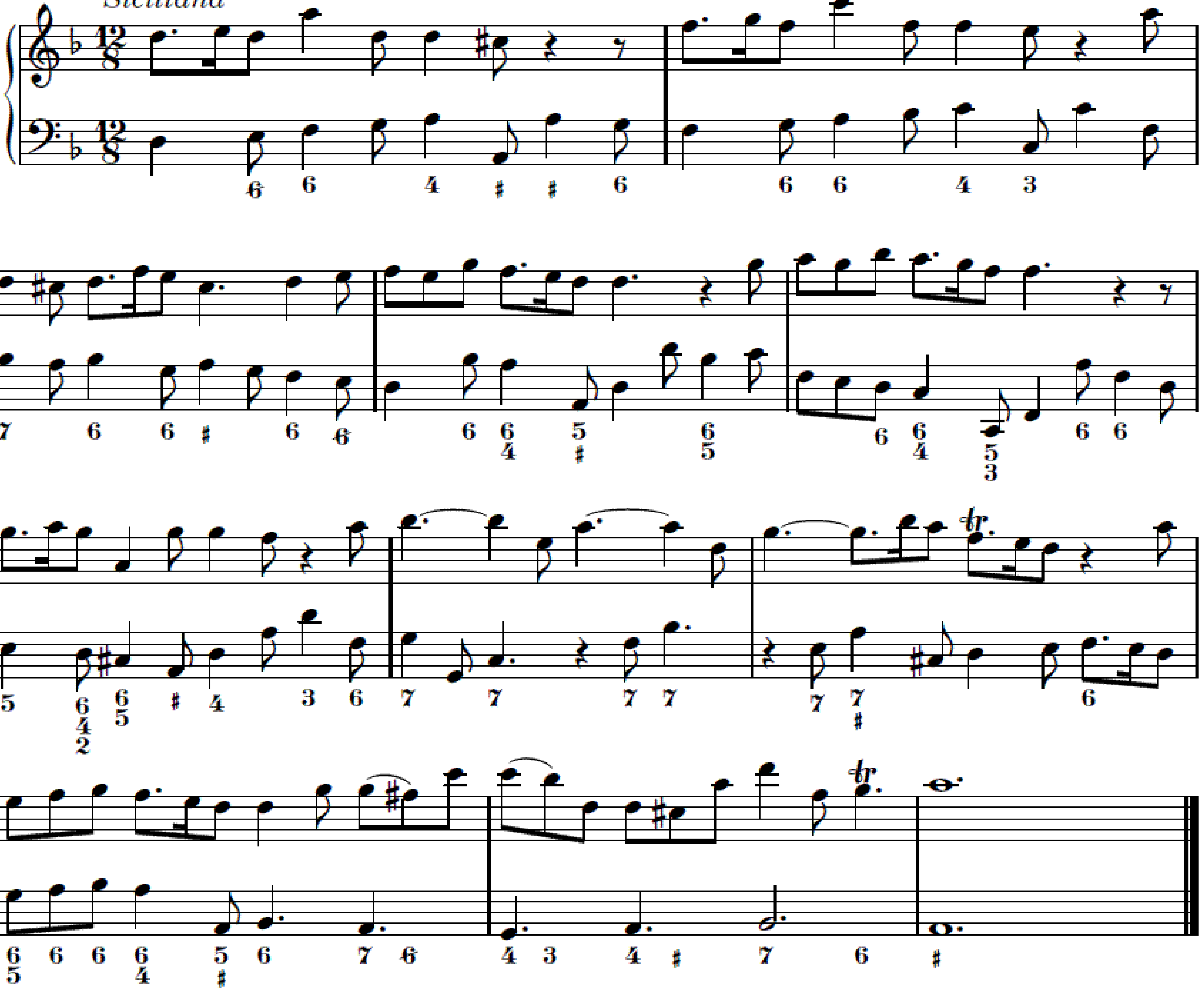
SELECT firstname, lastName **from** employee **where** age = 42;

([+-]?[0-9]*) | ([A-Z][a-z]+)



C_1
T_2
T_3
R_3
C_2
D_1
R_2
P_1
R_4
1
R_1
2
T_1
L_1
9V

Siciliana



What if…

What if…

you could
<u>build</u>
the DSL
YOU want ?

mbeddr C
Approach

An extensible C with support for embedded developemnt

SDK for building your own Language Extensions!

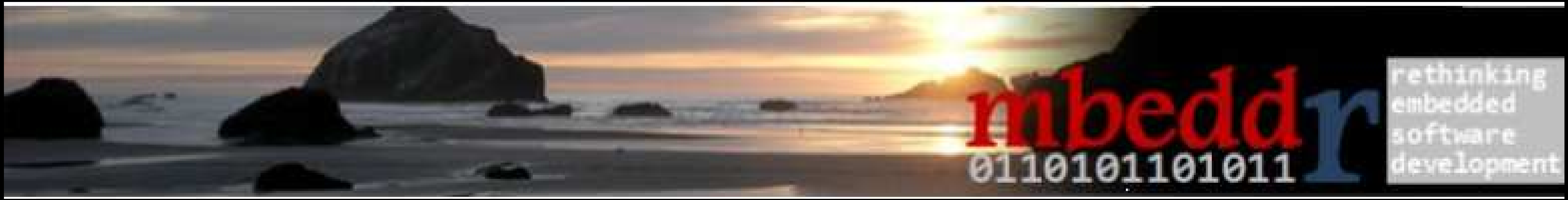
Build with open source

JetBrains

MPS

Open Source

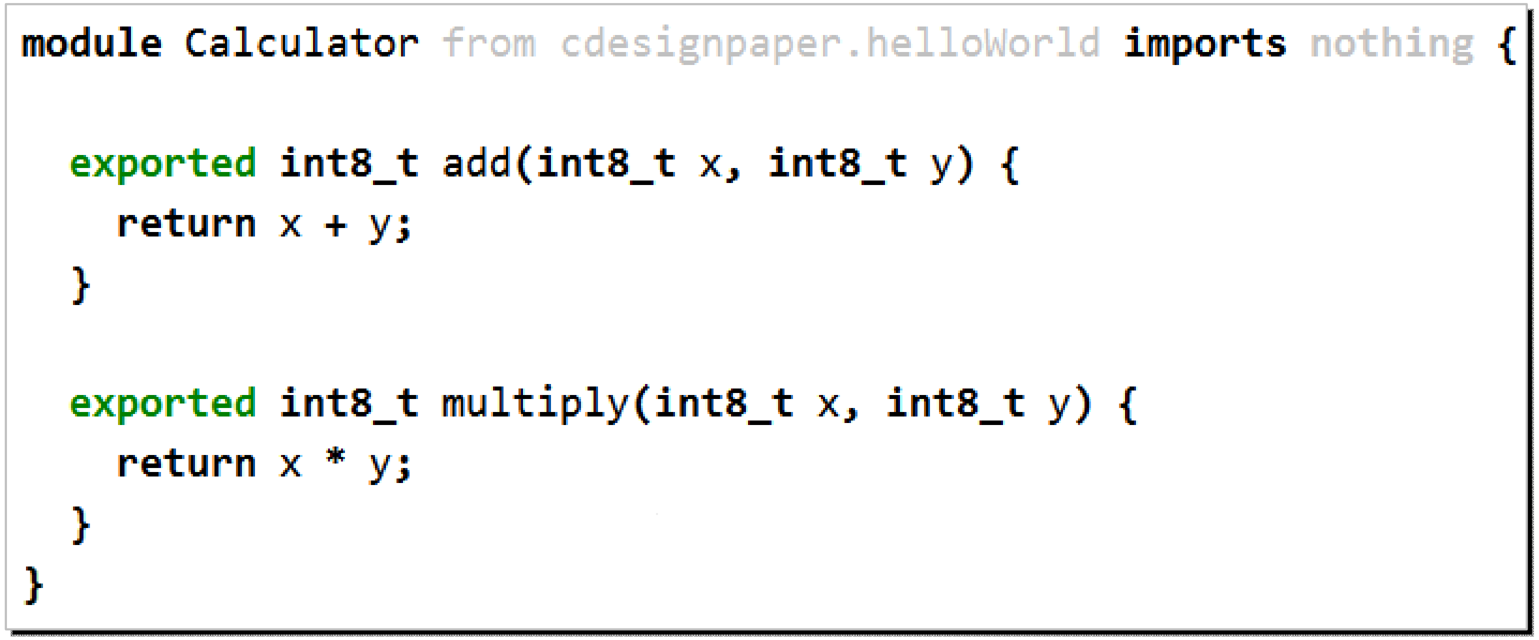
Language Workbench



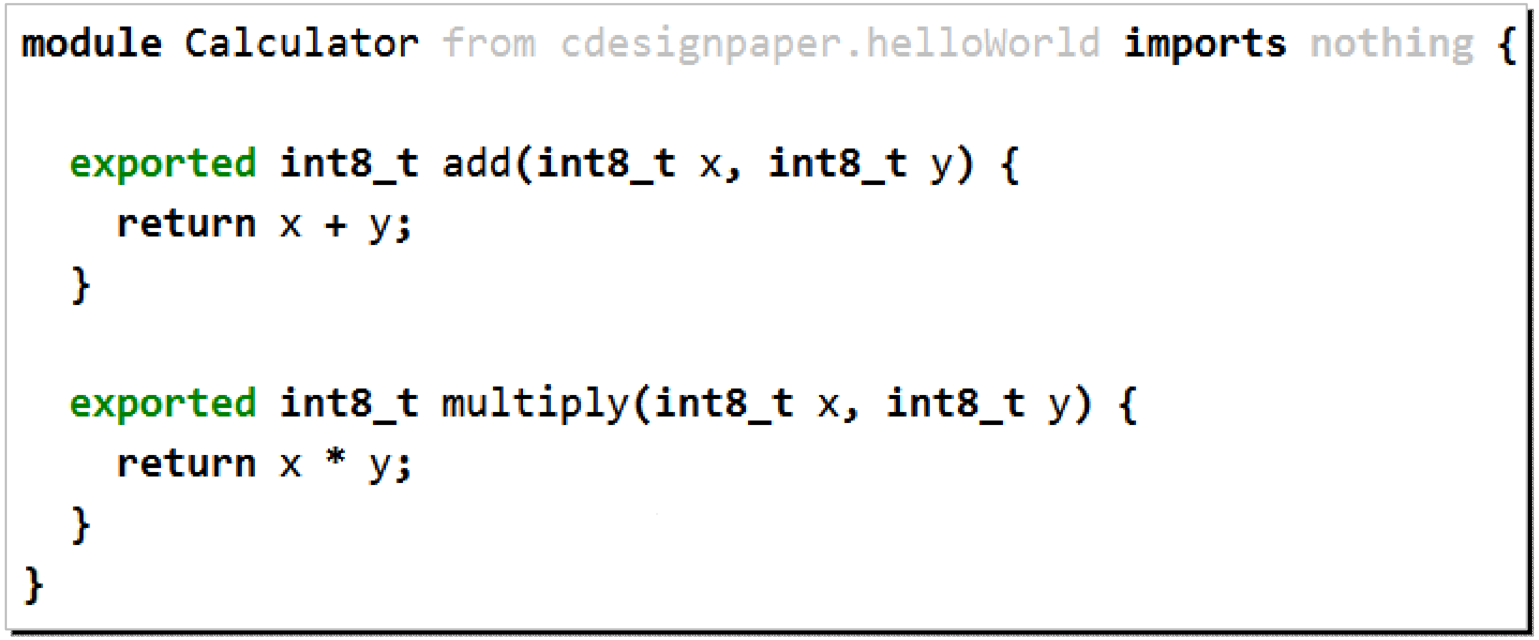
Incremental Extension of C

C
(cleaned-up)

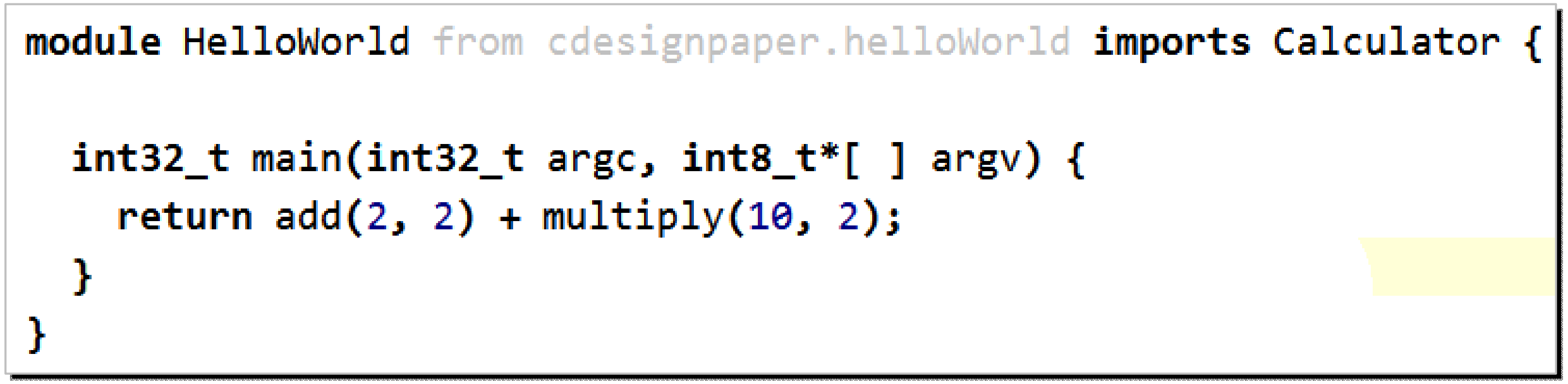
removed bad stuff…



```
module Calculator from cdesignpaper.helloWorld imports nothing {

  exported int8_t add(int8_t x, int8_t y) {
    return x + y;
  }

  exported int8_t multiply(int8_t x, int8_t y) {
    return x * y;
  }
}
```



```
module Calculator from cdesignpaper.helloWorld imports nothing {

  exported int8_t add(int8_t x, int8_t y) {
    return x + y;
  }

  exported int8_t multiply(int8_t x, int8_t y) {
    return x * y;
  }
}
```

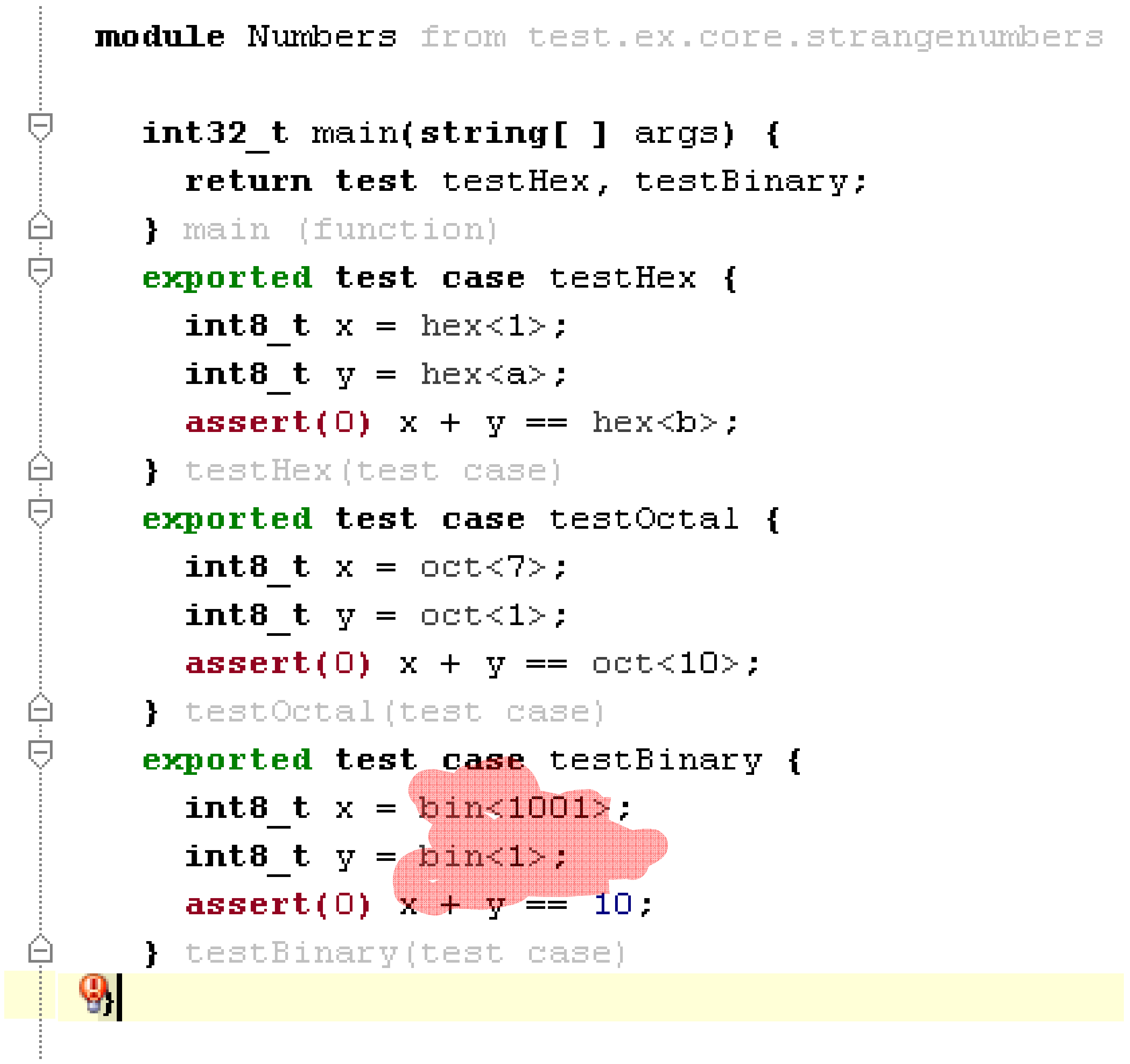


```
module HelloWorld from cdesignpaper.helloWorld imports Calculator {

  int32_t main(int32_t argc, int8_t*[ ] argv) {
    return add(2, 2) + multiply(10, 2);
  }
}
```

Binary literals in C++11 ?
(after years)

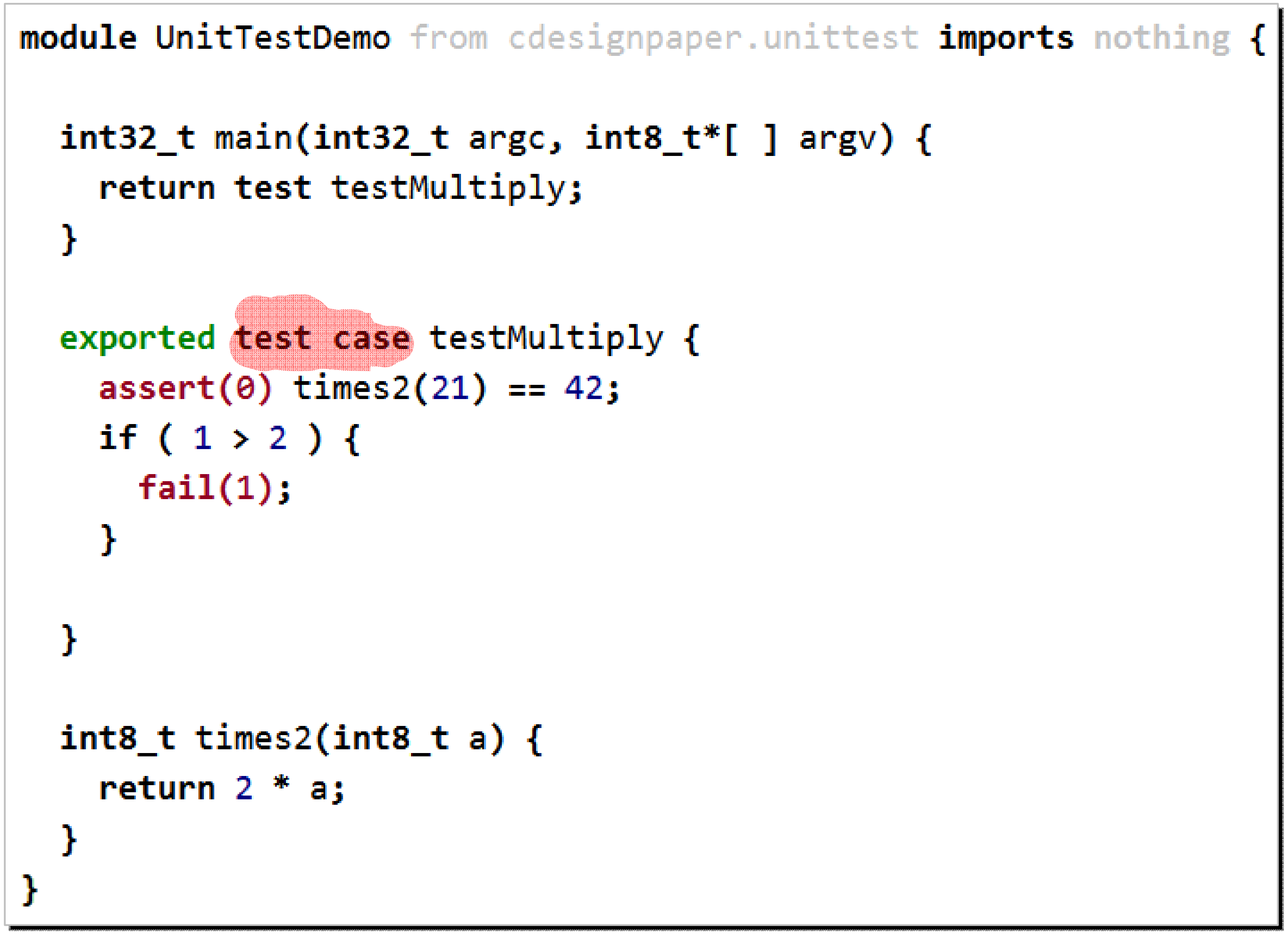
Binary literals in C++11 ?
(after years)

Supported in mbeddr C
(2 hours to implement...)

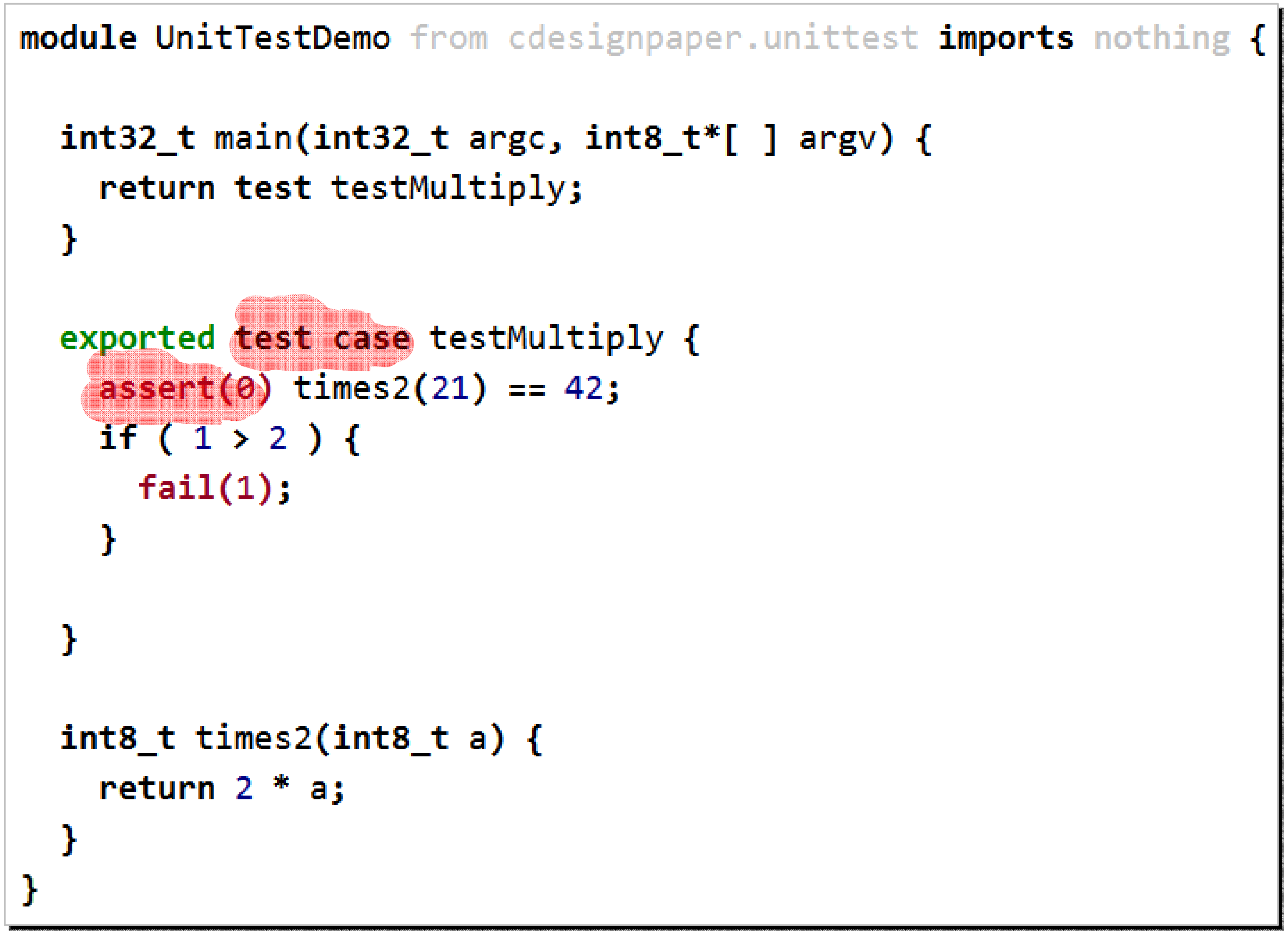


```
module Numbers from test.ex.core.strangenumbers

  int32_t main(string[ ] args) {
    return test testHex, testBinary;
  } main (function)
  exported test case testHex {
    int8_t x = hex<1>;
    int8_t y = hex<a>;
    assert(0) x + y == hex<b>;
  } testHex(test case)
  exported test case testOctal {
    int8_t x = oct<7>;
    int8_t y = oct<1>;
    assert(0) x + y == oct<10>;
  } testOctal(test case)
  exported test case testBinary {
    int8_t x = bin<1001>;
    int8_t y = bin<1>;
    assert(0) x + y == 10;
  } testBinary(test case)
}
```

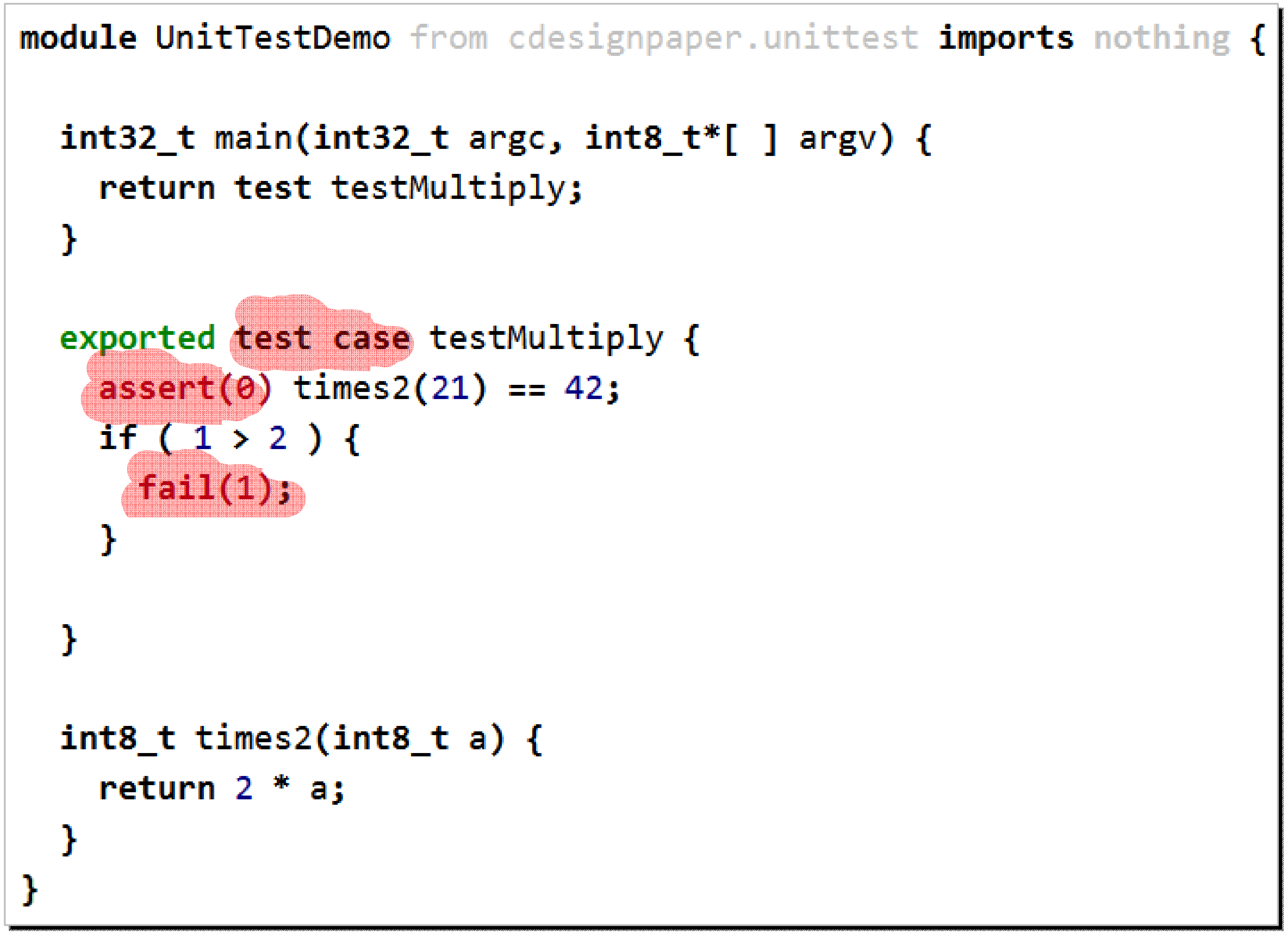
Native Support for Unit Testing



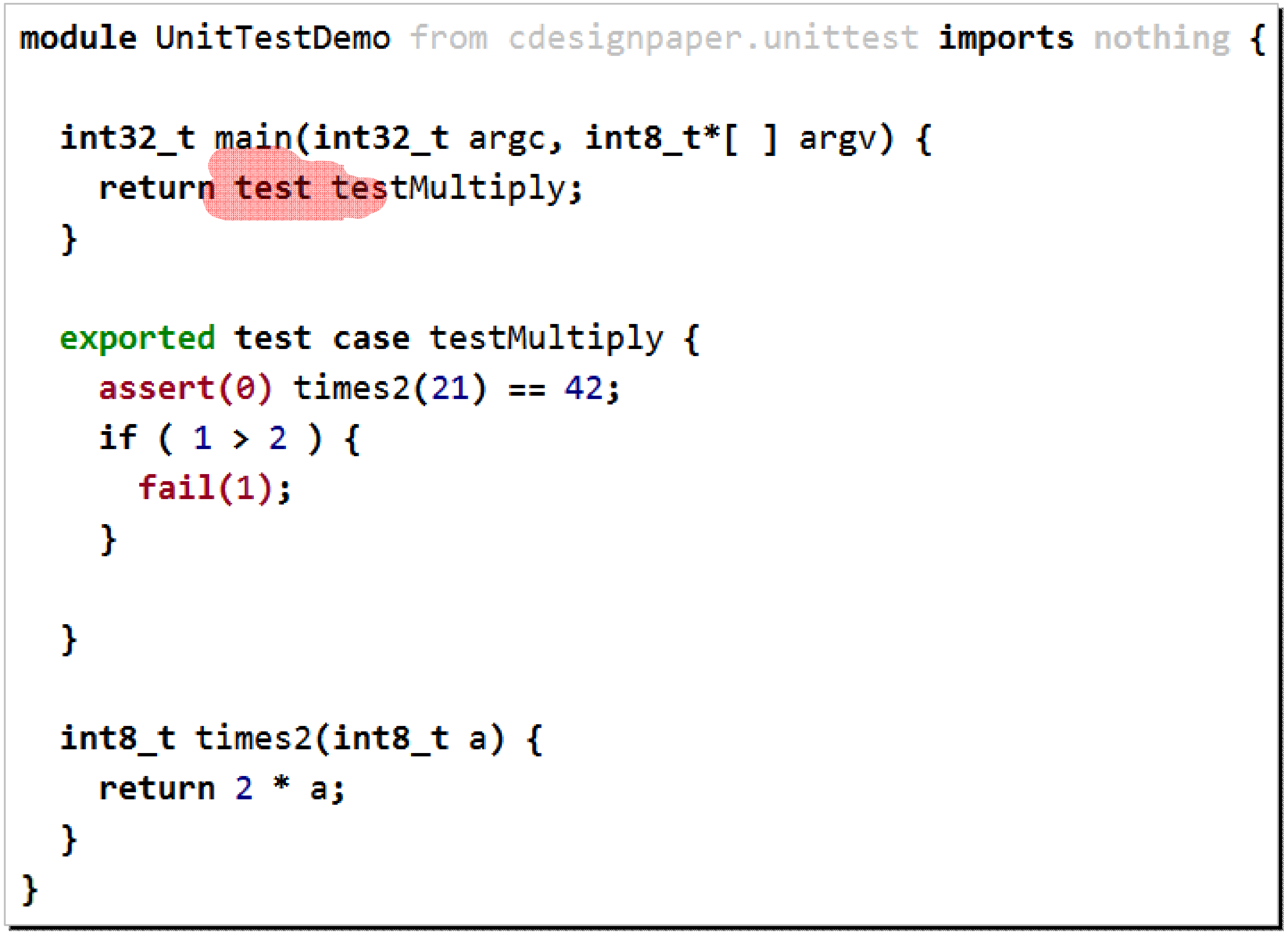
```
module UnitTestDemo from cdesignpaper.unittest imports nothing {

  int32_t main(int32_t argc, int8_t*[ ] argv) {
    return test testMultiply;
  }

  exported test case testMultiply {
    assert(0) times2(21) == 42;
    if ( 1 > 2 ) {
      fail(1);
    }

  }

  int8_t times2(int8_t a) {
    return 2 * a;
  }
}
```



```
module UnitTestDemo from cdesignpaper.unittest imports nothing {

  int32_t main(int32_t argc, int8_t*[ ] argv) {
    return test testMultiply;
  }

  exported test case testMultiply {
    assert(0) times2(21) == 42;
    if ( 1 > 2 ) {
      fail(1);
    }

  }

  int8_t times2(int8_t a) {
    return 2 * a;
  }
}
```



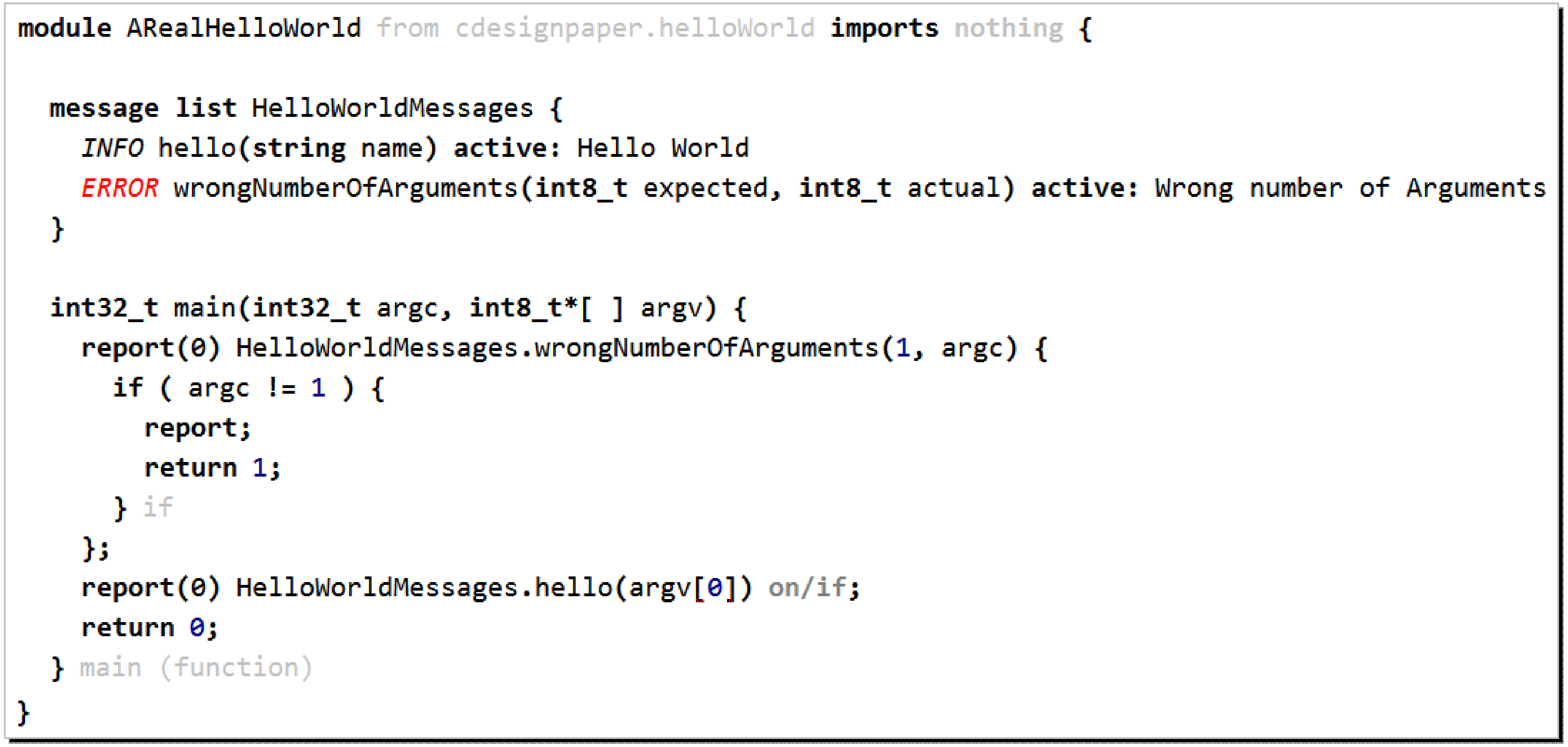
```
module UnitTestDemo from cdesignpaper.unittest imports nothing {

  int32_t main(int32_t argc, int8_t*[ ] argv) {
    return test testMultiply;
  }

  exported test case testMultiply {
    assert(0) times2(21) == 42;
    if ( 1 > 2 ) {
      fail(1);
    }

  }

  int8_t times2(int8_t a) {
    return 2 * a;
  }
}
```

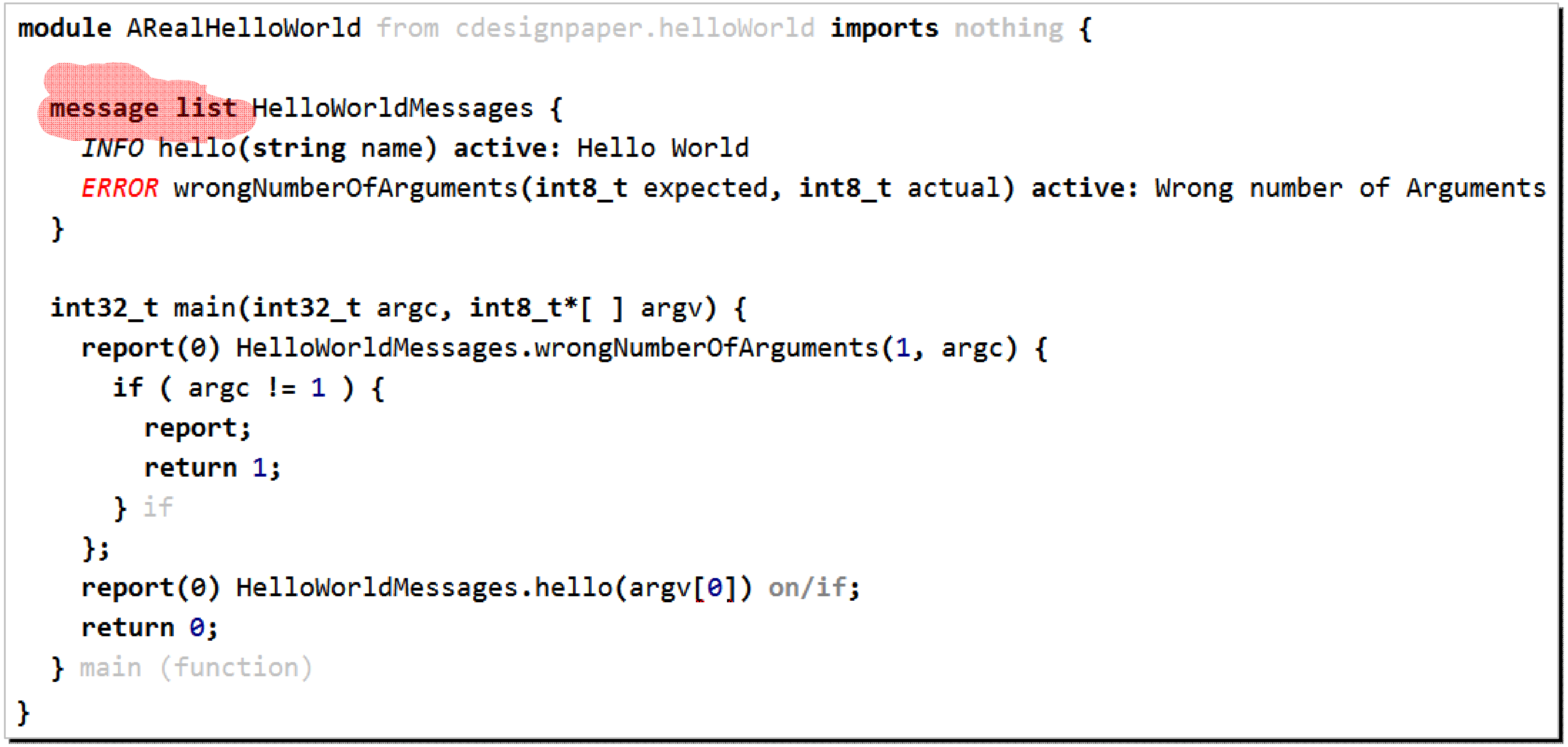


```
module UnitTestDemo from cdesignpaper.unittest imports nothing {

  int32_t main(int32_t argc, int8_t*[ ] argv) {
    return test testMultiply;
  }

  exported test case testMultiply {
    assert(0) times2(21) == 42;
    if ( 1 > 2 ) {
      fail(1);
    }

  }

  int8_t times2(int8_t a) {
    return 2 * a;
  }
}
```

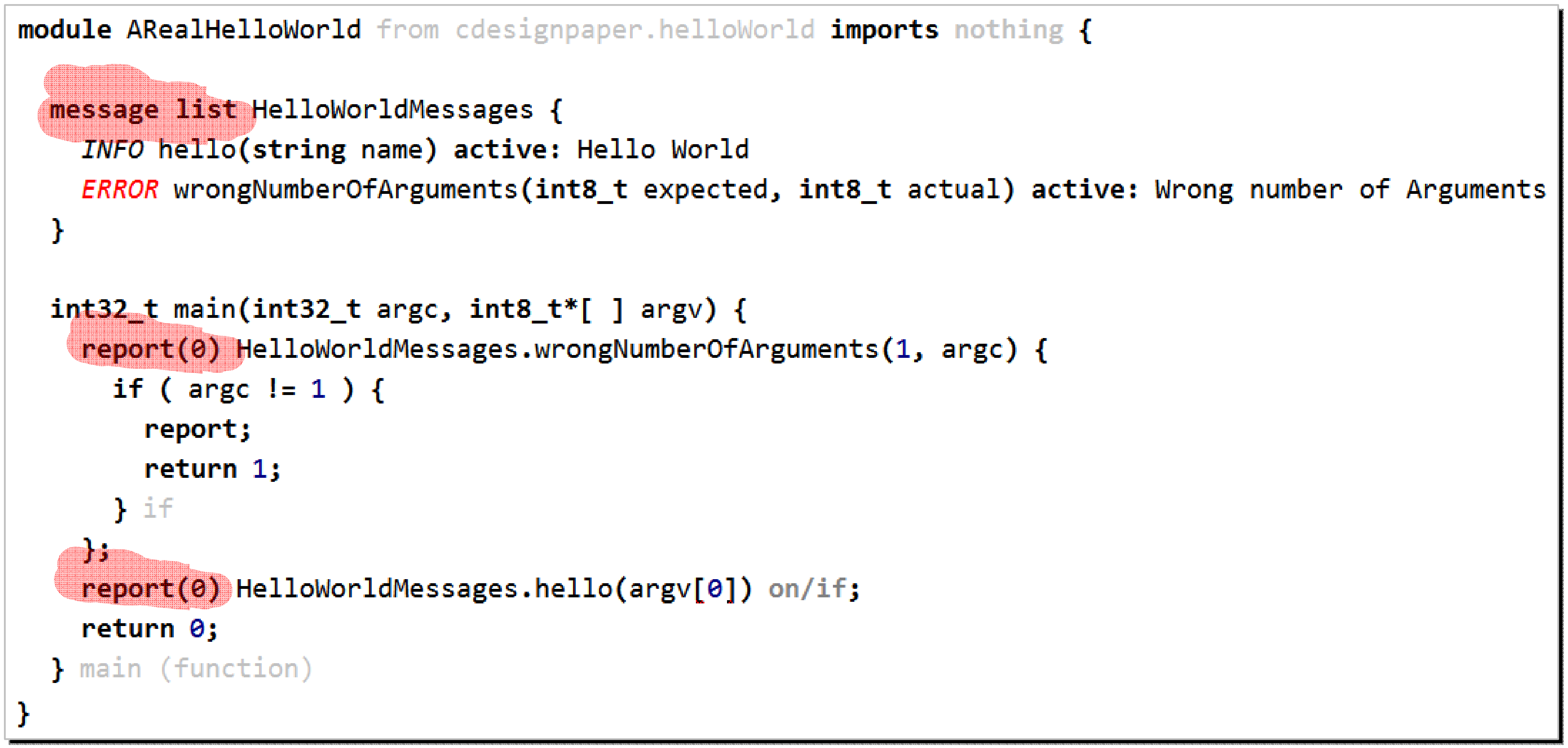
Native Support

for Unit Testing and Logging

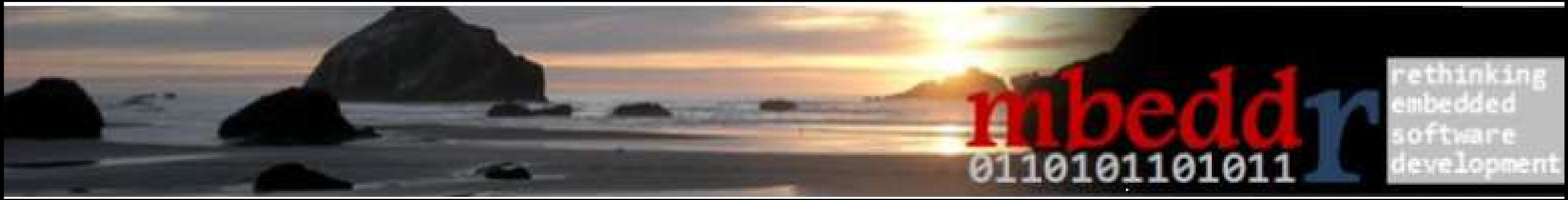

```
module ARealHelloWorld from cdesignpaper.helloWorld imports nothing {

  message list HelloWorldMessages {
    INFO hello(string name) active: Hello World
    ERROR wrongNumberOfArguments(int8_t expected, int8_t actual) active: Wrong number of Arguments
  }

  int32_t main(int32_t argc, int8_t*[ ] argv) {
    report(0) HelloWorldMessages.wrongNumberOfArguments(1, argc) {
      if ( argc != 1 ) {
        report;
        return 1;
      } if
    };
    report(0) HelloWorldMessages.hello(argv[0]) on/if;
    return 0;
  } main (function)
}
```



```
module ARealHelloWorld from cdesignpaper.helloWorld imports nothing {

  message list HelloWorldMessages {
    INFO hello(string name) active: Hello World
    ERROR wrongNumberOfArguments(int8_t expected, int8_t actual) active: Wrong number of Arguments
  }

  int32_t main(int32_t argc, int8_t*[ ] argv) {
    report(0) HelloWorldMessages.wrongNumberOfArguments(1, argc) {
      if ( argc != 1 ) {
        report;
        return 1;
      } if
    };
    report(0) HelloWorldMessages.hello(argv[0]) on/if;
    return 0;
  } main (function)
}
```



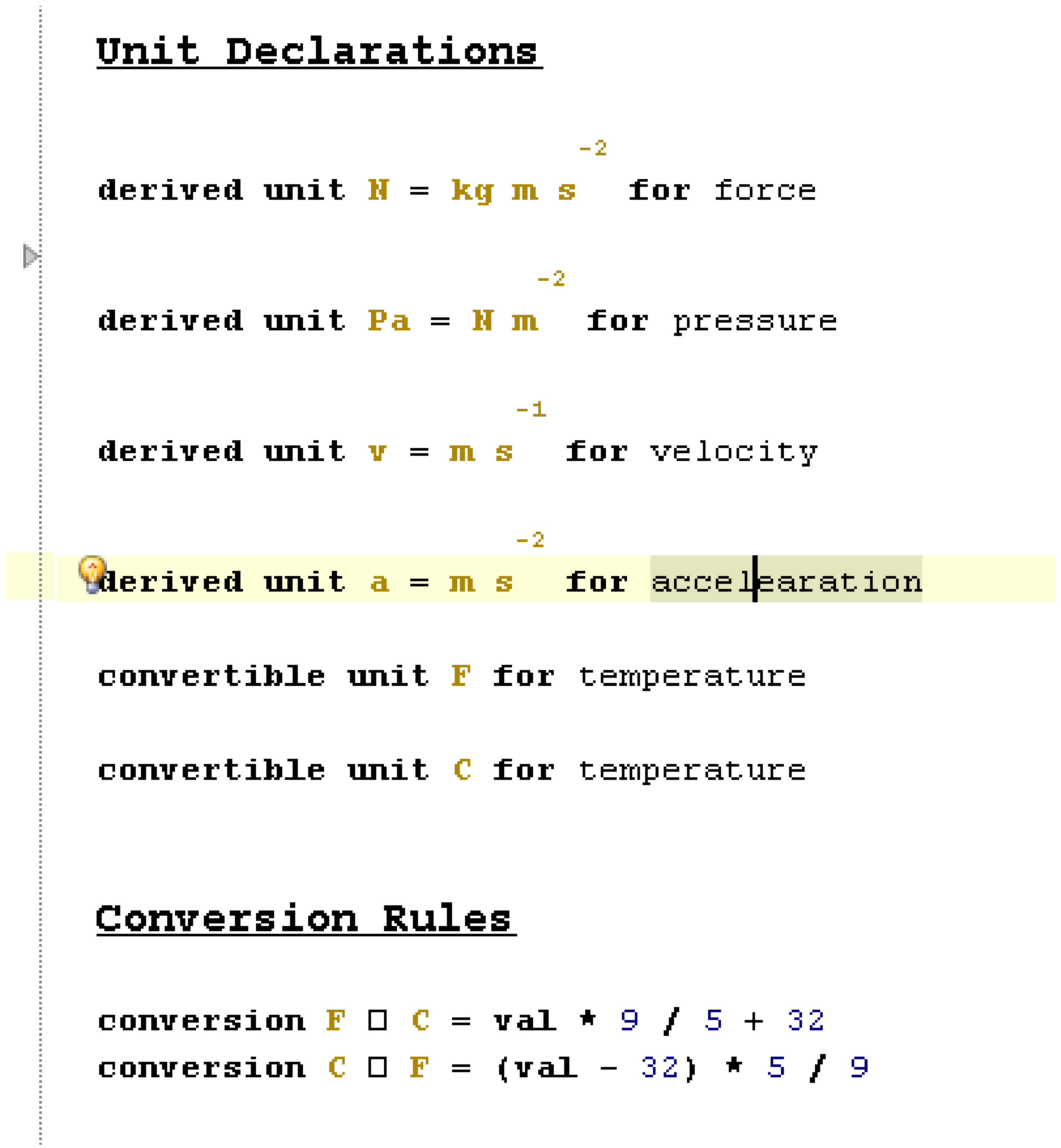
```
module ARealHelloWorld from cdesignpaper.helloWorld imports nothing {

  message list HelloWorldMessages {
    INFO hello(string name) active: Hello World
    ERROR wrongNumberOfArguments(int8_t expected, int8_t actual) active: Wrong number of Arguments
  }

  int32_t main(int32_t argc, int8_t*[ ] argv) {
    report(0) HelloWorldMessages.wrongNumberOfArguments(1, argc) {
      if ( argc != 1 ) {
        report;
        return 1;
      } if
    };
    report(0) HelloWorldMessages.hello(argv[0]) on/if;
    return 0;
  } main (function)
}
```



Incremental Extension of C

Physical Units

Without Template meta programming Hell !!!



Unit Declarations

derived unit N = kg m s^{-2} for force

derived unit Pa = N m^{-2} for pressure

derived unit v = m s^{-1} for velocity

derived unit a = m s^{-2} for accelearation

convertible unit F for temperature

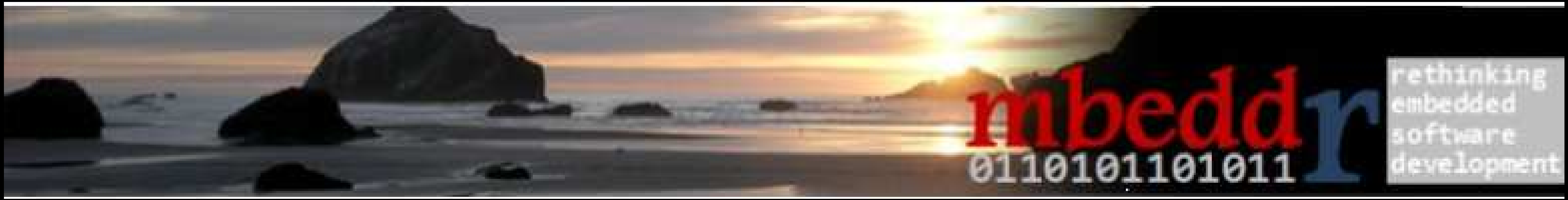
convertible unit C for temperature

Conversion Rules

conversion F □ C = val * 9 / 5 + 32
conversion C □ F = (val - 32) * 5 / 9

And
error messages
at
EDITING time !

```
void testBasicUnits() {
  int8_t/N/ n = 3 N;
                                 -2
  int8_t/N/ n3 = 3 kg m s   ;
  int8_t/N/ n4 = 3 N * 4s / 3s;
  int8_t/N m/ n5 = n4 * 3 m;

  int8_t/cd/ aLuminousIntensity = 0cd;
  int8_t/m/ length;
  int8_t/s/ time;
               -1
  int8_t/m s   / speed = length / time;

  int8_t/v/ thisShouldNotWork = <node length + time has err
```

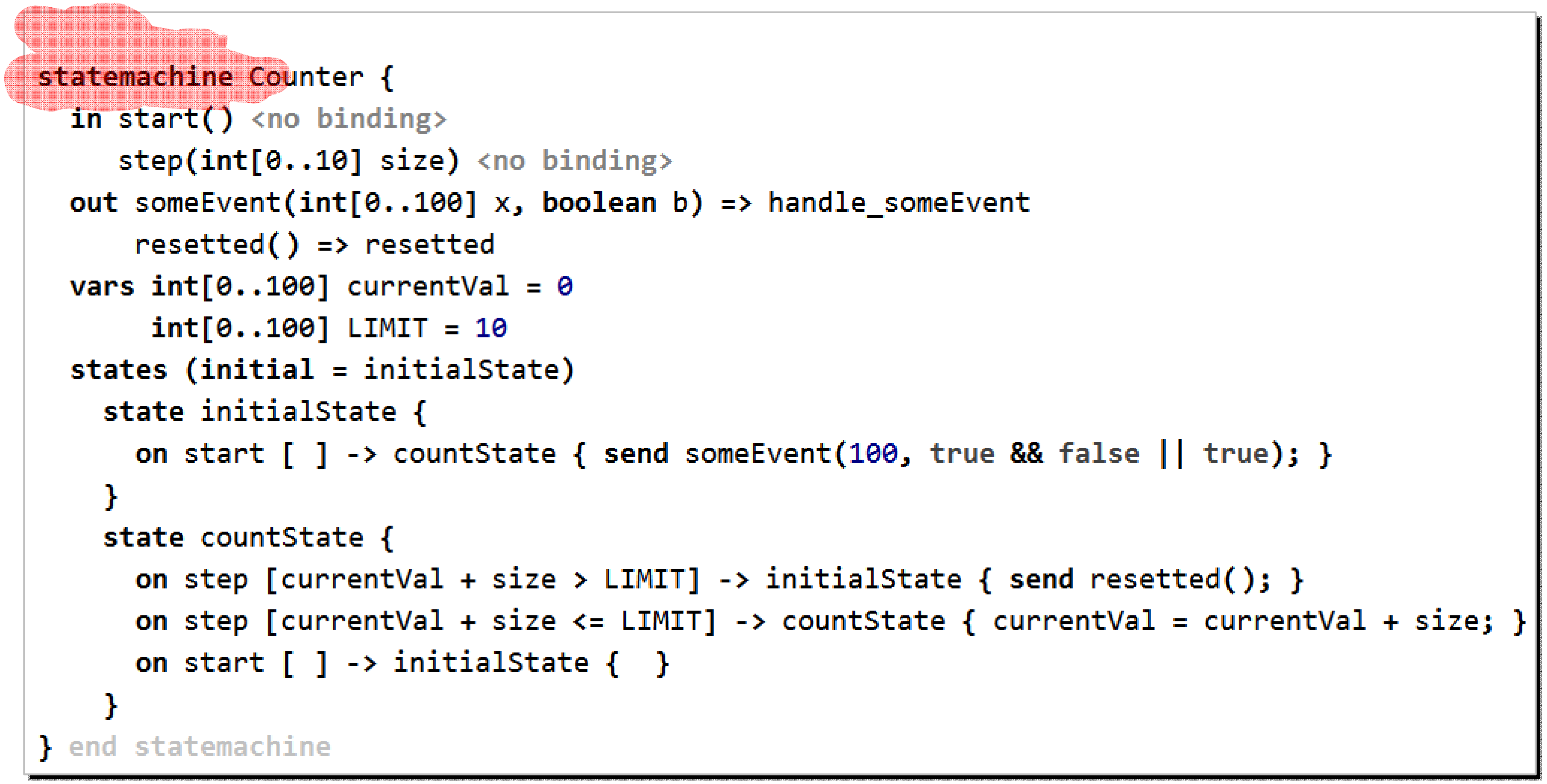


Incremental Extension of C

State Machines

Physical Units

State Machines
as
first class
concepts

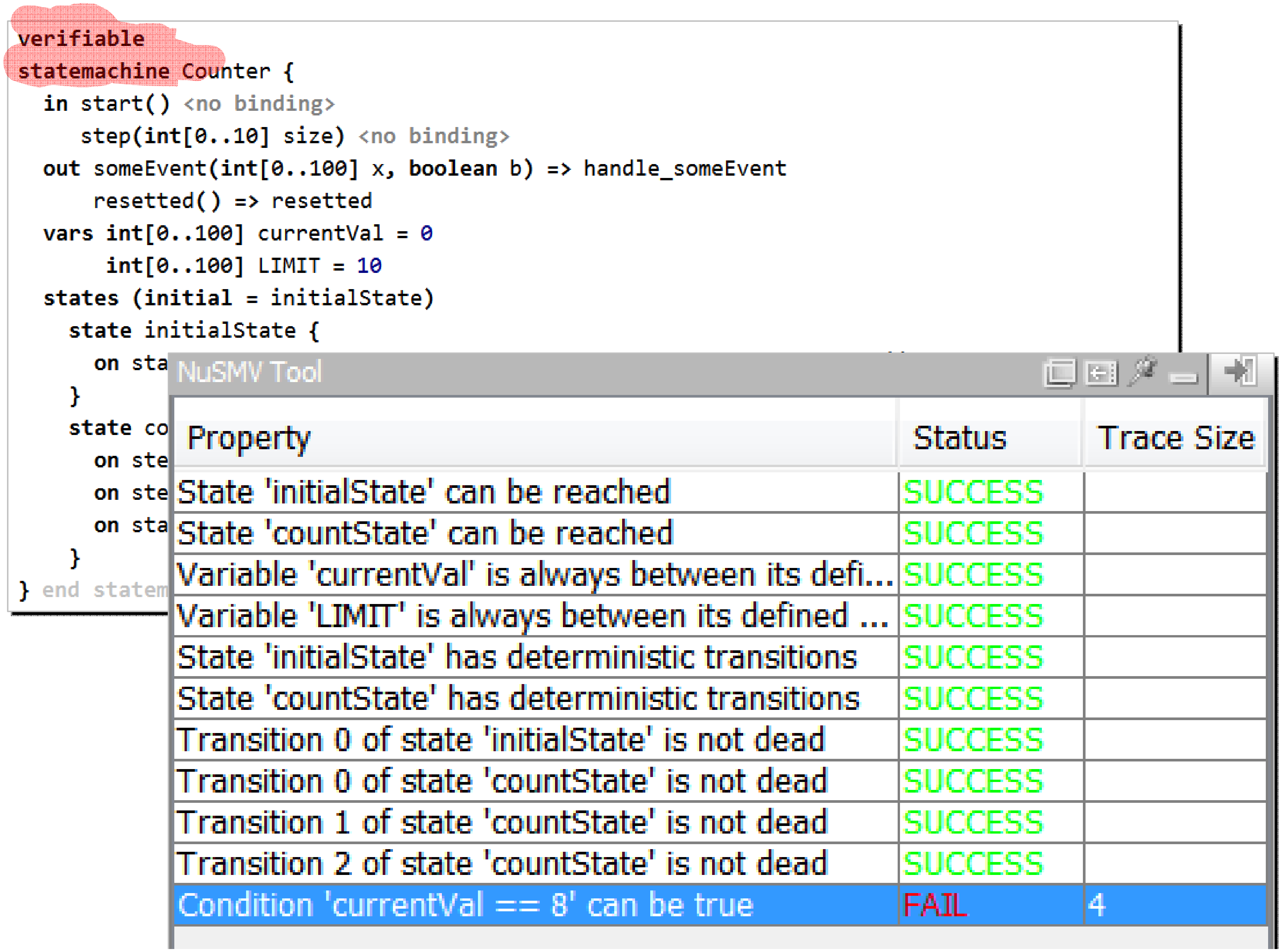


```
statemachine Counter {
  in start() <no binding>
     step(int[0..10] size) <no binding>
  out someEvent(int[0..100] x, boolean b) => handle_someEvent
      resetted() => resetted
  vars int[0..100] currentVal = 0
       int[0..100] LIMIT = 10
  states (initial = initialState)
    state initialState {
      on start [ ] -> countState { send someEvent(100, true && false || true); }
    }
    state countState {
      on step [currentVal + size > LIMIT] -> initialState { send resetted(); }
      on step [currentVal + size <= LIMIT] -> countState { currentVal = currentVal + size; }
      on start [ ] -> initialState {  }
    }
} end statemachine
```

State Machines

+

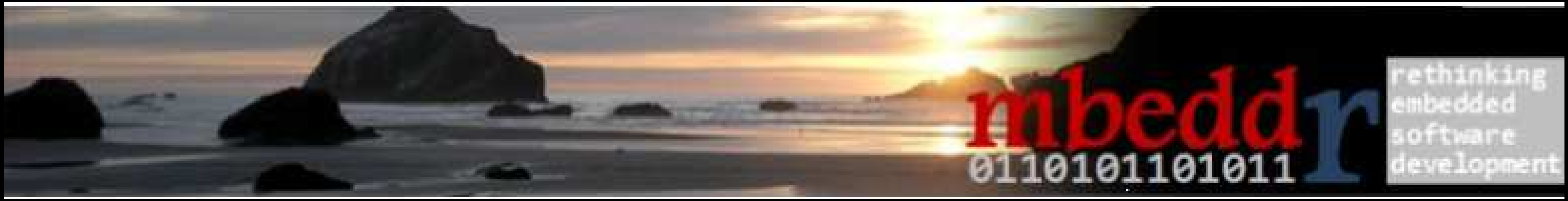
Model Checking



```
verifiable
statemachine Counter {
  in start() <no binding>
     step(int[0..10] size) <no binding>
  out someEvent(int[0..100] x, boolean b) => handle_someEvent
      resetted() => resetted
  vars int[0..100] currentVal = 0
       int[0..100] LIMIT = 10
  states (initial = initialState)
    state initialState {
      on sta
    }
    state co
      on ste
      on ste
      on sta
    }
} end statem
```

NuSMV Tool

Property	Status	Trace Size
State 'initialState' can be reached	SUCCESS	
State 'countState' can be reached	SUCCESS	
Variable 'currentVal' is always between its defi...	SUCCESS	
Variable 'LIMIT' is always between its defined ...	SUCCESS	
State 'initialState' has deterministic transitions	SUCCESS	
State 'countState' has deterministic transitions	SUCCESS	
Transition 0 of state 'initialState' is not dead	SUCCESS	
Transition 0 of state 'countState' is not dead	SUCCESS	
Transition 1 of state 'countState' is not dead	SUCCESS	
Transition 2 of state 'countState' is not dead	SUCCESS	
Condition 'currentVal == 8' can be true	FAIL	4



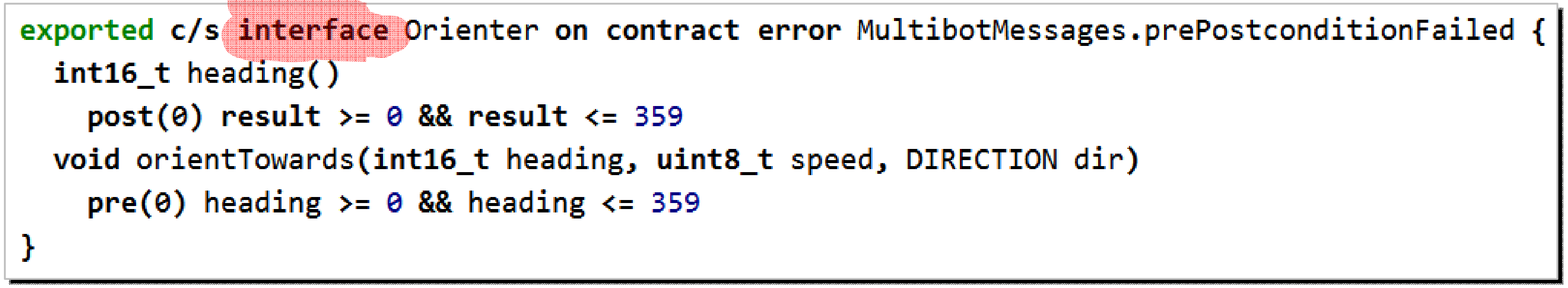
Incremental Extension of C

Components

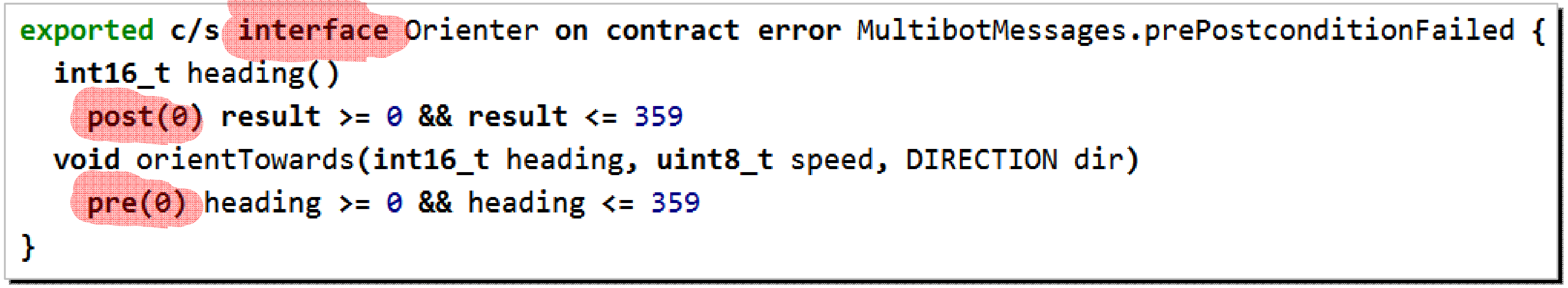
State Machines

Physical Units

Components
Interfaces

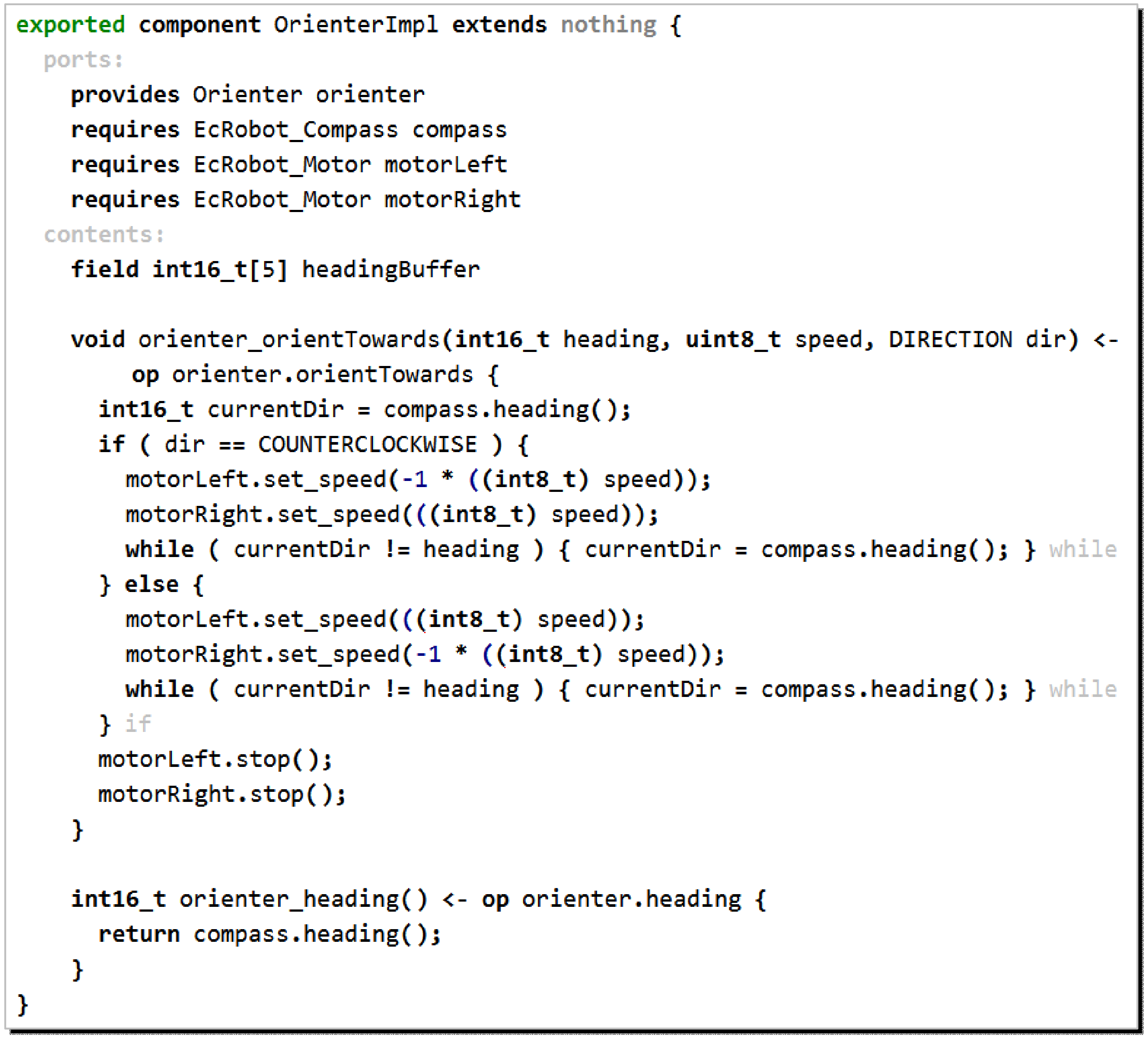


```
exported c/s interface Orienter on contract error MultibotMessages.prePostconditionFailed {
  int16_t heading()
    post(0) result >= 0 && result <= 359
  void orientTowards(int16_t heading, uint8_t speed, DIRECTION dir)
    pre(0) heading >= 0 && heading <= 359
}
```



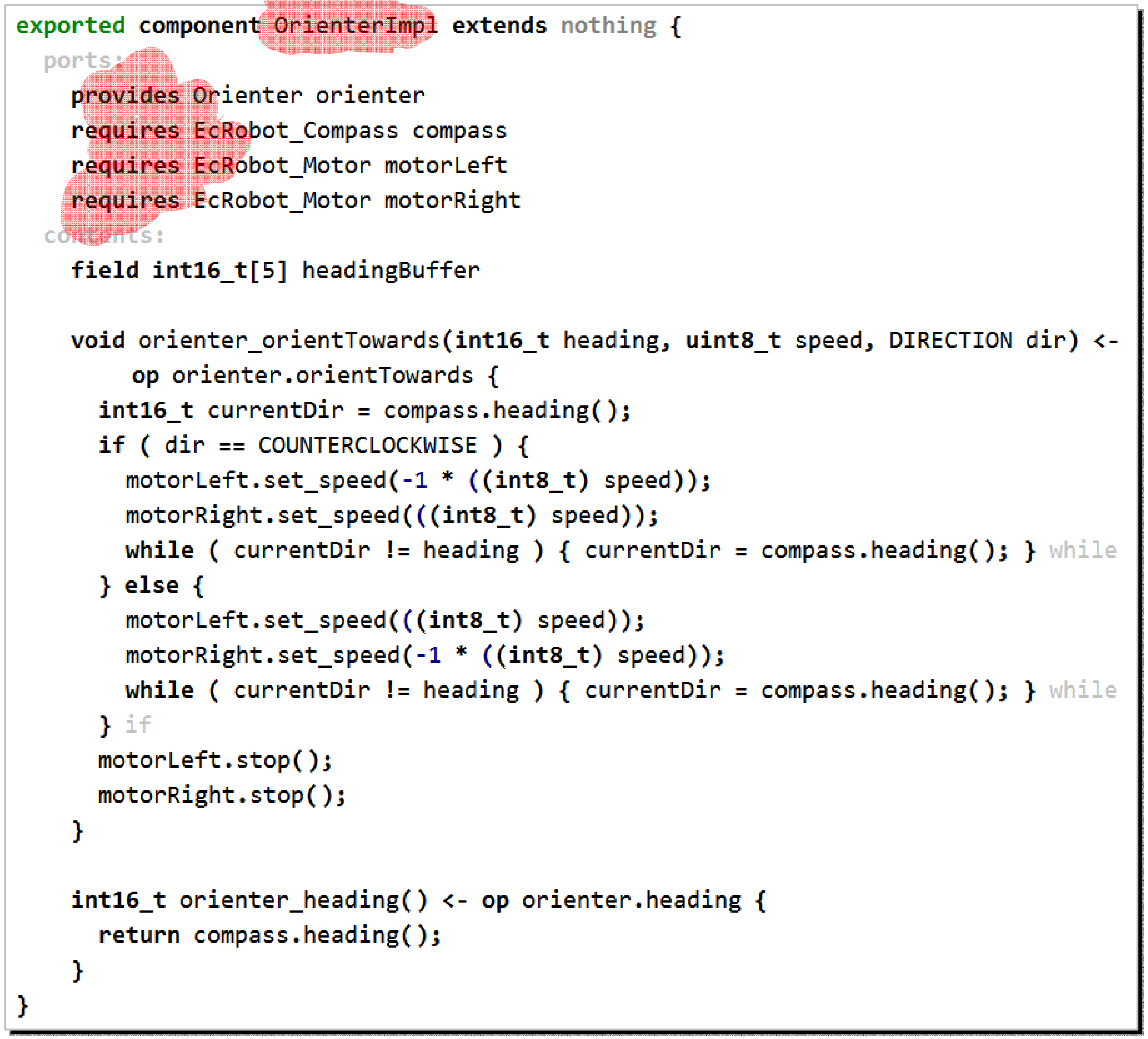
```
exported c/s interface Orienter on contract error MultibotMessages.prePostconditionFailed {
  int16_t heading()
    post(0) result >= 0 && result <= 359
  void orientTowards(int16_t heading, uint8_t speed, DIRECTION dir)
    pre(0) heading >= 0 && heading <= 359
}
```

Components
Interfaces
with Contracts !

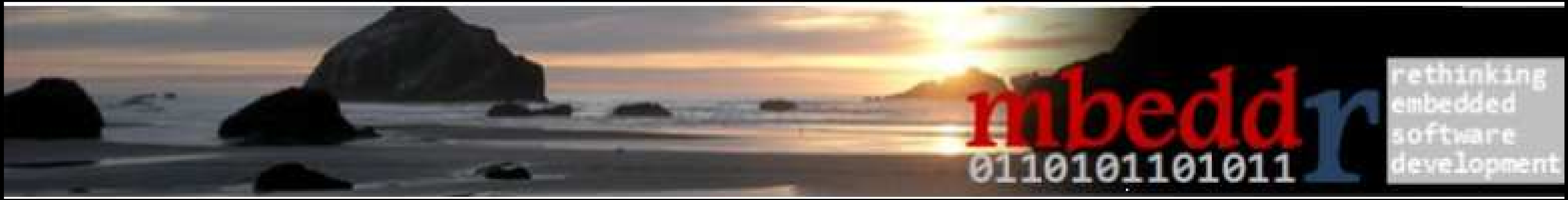


```
exported component OrienterImpl extends nothing {
  ports:
    provides Orienter orienter
    requires EcRobot_Compass compass
    requires EcRobot_Motor motorLeft
    requires EcRobot_Motor motorRight
  contents:
    field int16_t[5] headingBuffer

    void orienter_orientTowards(int16_t heading, uint8_t speed, DIRECTION dir) <-
         op orienter.orientTowards {
      int16_t currentDir = compass.heading();
      if ( dir == COUNTERCLOCKWISE ) {
        motorLeft.set_speed(-1 * ((int8_t) speed));
        motorRight.set_speed(((int8_t) speed));
        while ( currentDir != heading ) { currentDir = compass.heading(); } while
      } else {
        motorLeft.set_speed(((int8_t) speed));
        motorRight.set_speed(-1 * ((int8_t) speed));
        while ( currentDir != heading ) { currentDir = compass.heading(); } while
      } if
      motorLeft.stop();
      motorRight.stop();
    }

    int16_t orienter_heading() <- op orienter.heading {
      return compass.heading();
    }
}
```

Interface + Implementation



```
exported component OrienterImpl extends nothing {
  ports:
    provides Orienter orienter
    requires EcRobot_Compass compass
    requires EcRobot_Motor motorLeft
    requires EcRobot_Motor motorRight
  contents:
    field int16_t[5] headingBuffer

    void orienter_orientTowards(int16_t heading, uint8_t speed, DIRECTION dir) <-
         op orienter.orientTowards {
      int16_t currentDir = compass.heading();
      if ( dir == COUNTERCLOCKWISE ) {
        motorLeft.set_speed(-1 * ((int8_t) speed));
        motorRight.set_speed(((int8_t) speed));
        while ( currentDir != heading ) { currentDir = compass.heading(); } while
      } else {
        motorLeft.set_speed(((int8_t) speed));
        motorRight.set_speed(-1 * ((int8_t) speed));
        while ( currentDir != heading ) { currentDir = compass.heading(); } while
      } if
      motorLeft.stop();
      motorRight.stop();
    }

    int16_t orienter_heading() <- op orienter.heading {
      return compass.heading();
    }
}
```



Incremental Extension of C

Components
Productlines
State Machines
Physical Units

Product Line Variability

Define
a
feature model

```
feature model DeploymentConfiguration
  root ? {
    logging
    test
    valueTest [int8_t value]
  }
```

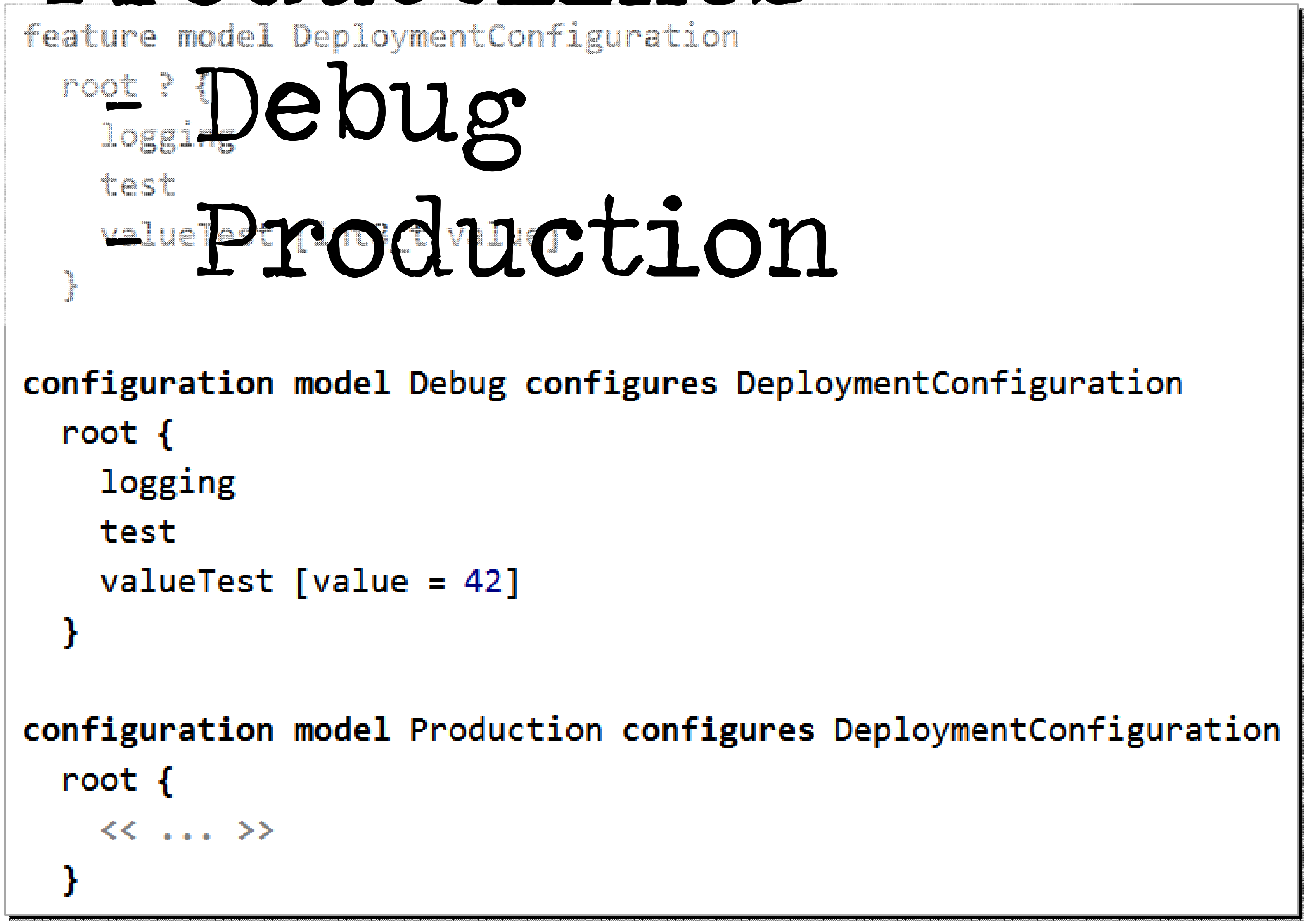
```
feature model DeploymentConfiguration
  root ? {
    logging
    test
    valueTest [int8_t value]
  }
```

Example Feature Model:

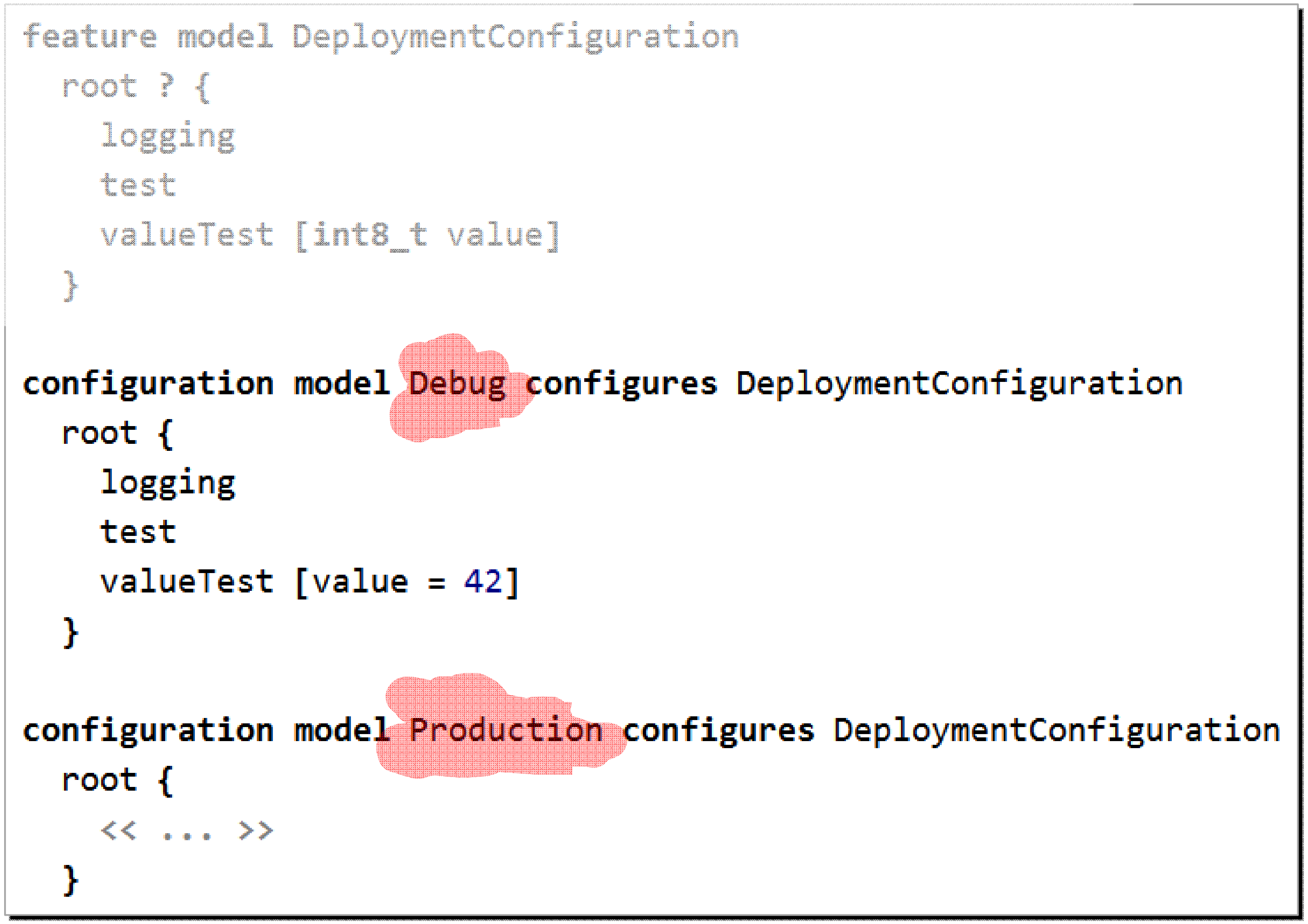
- logging
- testing
- a value

2 Productlines:

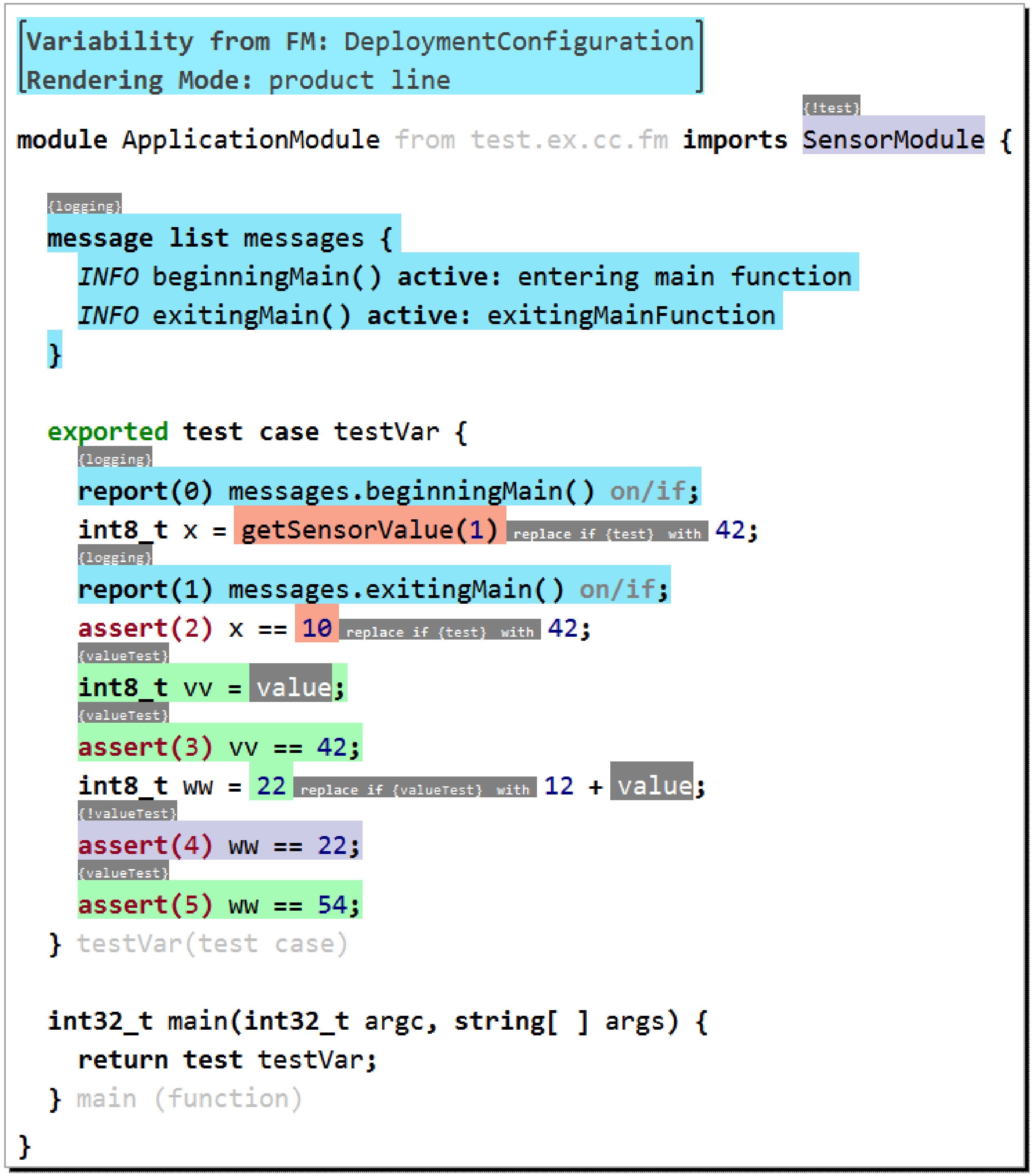
- Debug
- Production



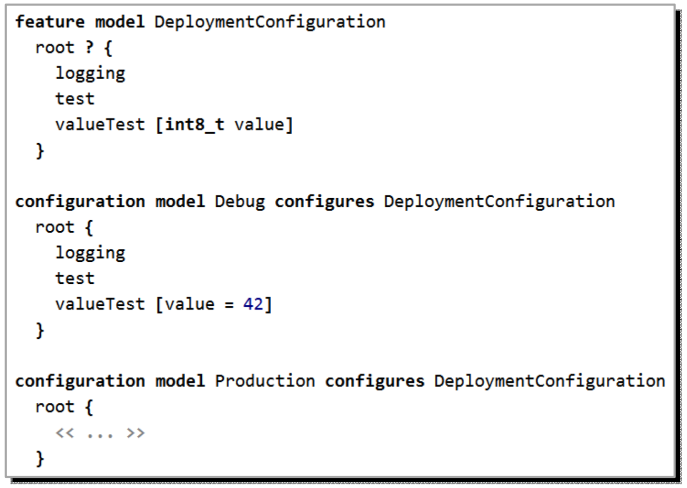
```
feature model DeploymentConfiguration
  root ? {
    logging
    test
    valueTest [int8 value]
  }

configuration model Debug configures DeploymentConfiguration
  root {
    logging
    test
    valueTest [value = 42]
  }

configuration model Production configures DeploymentConfiguration
  root {
    << ... >>
  }
```



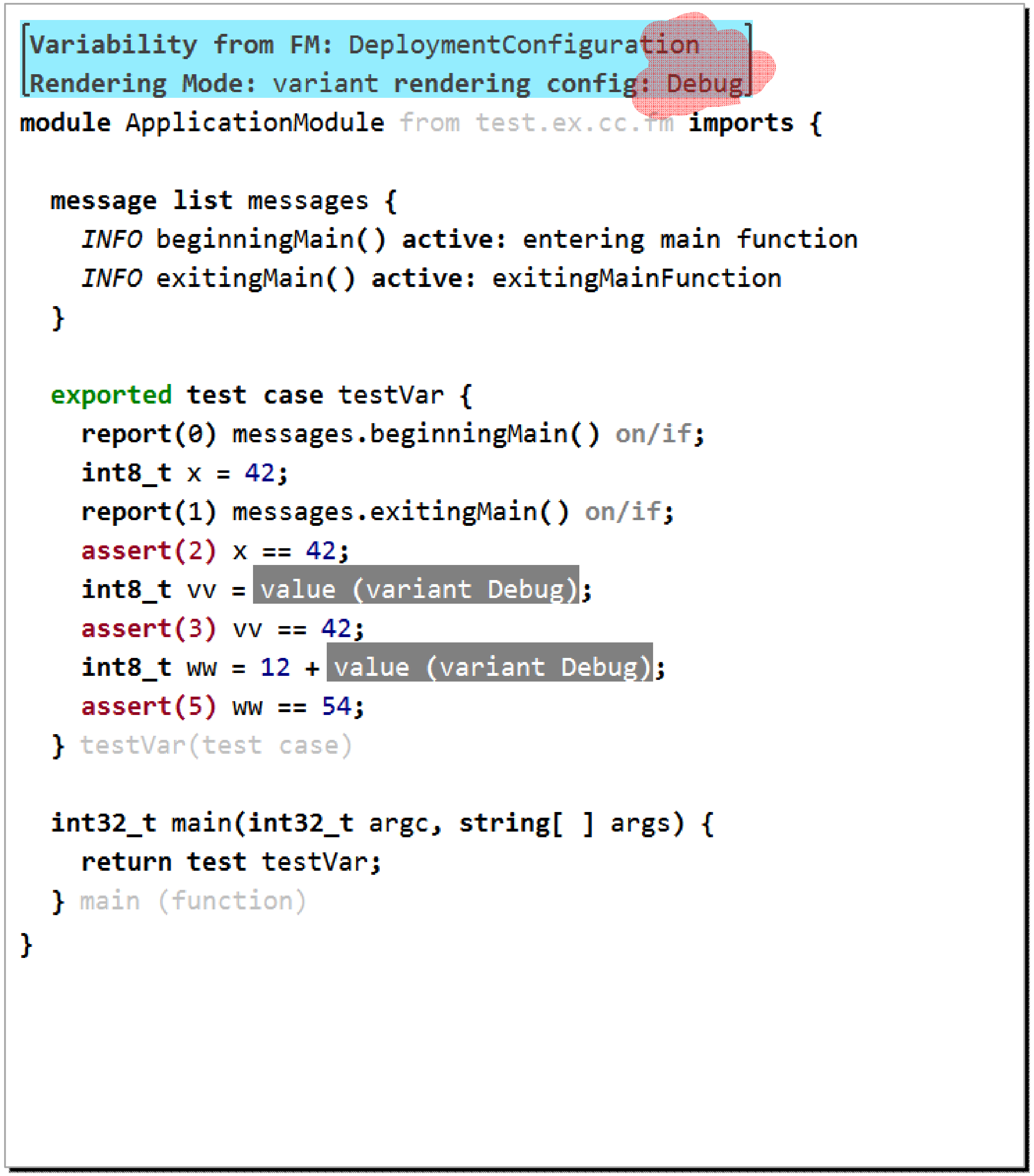
```
feature model DeploymentConfiguration
  root ? {
    logging
    test
    valueTest [int8_t value]
  }

configuration model Debug configures DeploymentConfiguration
  root {
    logging
    test
    valueTest [value = 42]
  }

configuration model Production configures DeploymentConfiguration
  root {
    << ... >>
  }
```



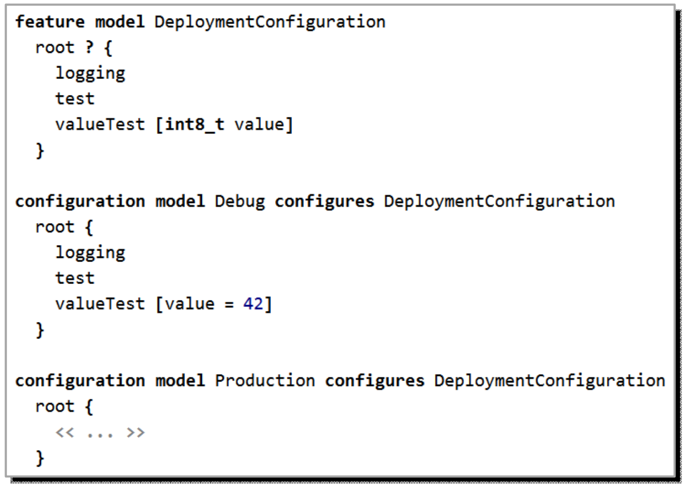
```
Variability from FM: DeploymentConfiguration
Rendering Mode: product line
                                                        {!test}
module ApplicationModule from test.ex.cc.fm imports SensorModule {

  {logging}
  message list messages {
    INFO beginningMain() active: entering main function
    INFO exitingMain() active: exitingMainFunction
  }

  exported test case testVar {
    {logging}
    report(0) messages.beginningMain() on/if;
    int8_t x = getSensorValue(1) replace if {test} with 42;
    {logging}
    report(1) messages.exitingMain() on/if;
    assert(2) x == 10 replace if {test} with 42;
    {valueTest}
    int8_t vv = value;
    {valueTest}
    assert(3) vv == 42;
    int8_t ww = 22 replace if {valueTest} with 12 + value;
    {!valueTest}
    assert(4) ww == 22;
    {valueTest}
    assert(5) ww == 54;
  } testVar(test case)

  int32_t main(int32_t argc, string[ ] args) {
    return test testVar;
  } main (function)
}
```



```
feature model DeploymentConfiguration
  root ? {
    logging
    test
    valueTest [int8_t value]
  }

configuration model Debug configures DeploymentConfiguration
  root {
    logging
    test
    valueTest [value = 42]
  }

configuration model Production configures DeploymentConfiguration
  root {
    << ... >>
  }
```

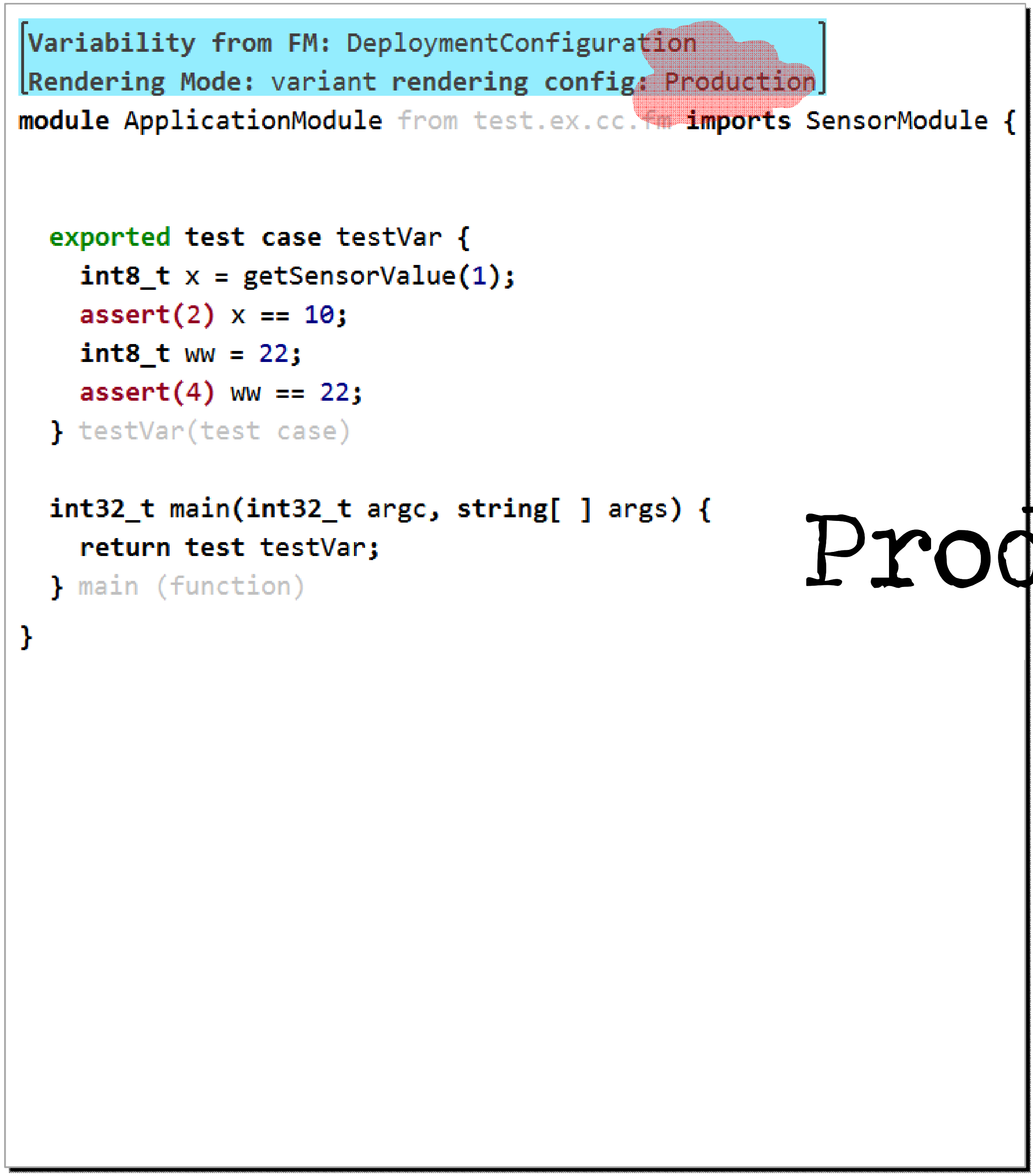


```
Variability from FM: DeploymentConfiguration
Rendering Mode: variant rendering config: Debug
module ApplicationModule from test.ex.cc.fm imports {

  message list messages {
    INFO beginningMain() active: entering main function
    INFO exitingMain() active: exitingMainFunction
  }

  exported test case testVar {
    report(0) messages.beginningMain() on/if;
    int8_t x = 42;
    report(1) messages.exitingMain() on/if;
    assert(2) x == 42;
    int8_t vv = value (variant Debug);
    assert(3) vv == 42;
    int8_t ww = 12 + value (variant Debug);
    assert(5) ww == 54;
  } testVar(test case)

  int32_t main(int32_t argc, string[ ] args) {
    return test testVar;
  } main (function)
}
```

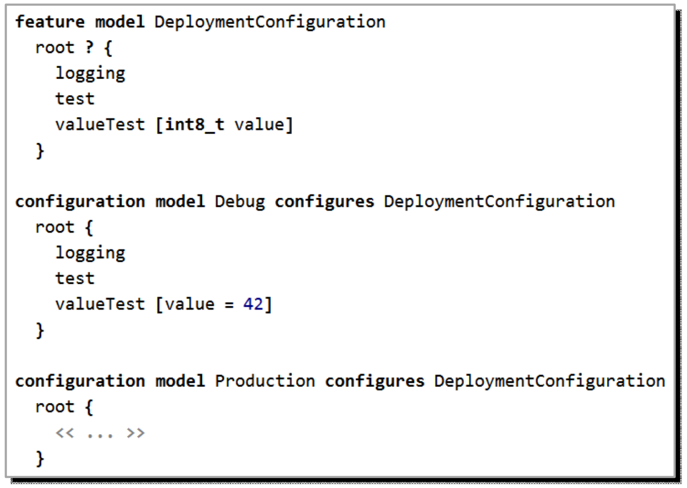


```
feature model DeploymentConfiguration
  root ? {
    logging
    test
    valueTest [int8_t value]
  }

configuration model Debug configures DeploymentConfiguration
  root {
    logging
    test
    valueTest [value = 42]
  }

configuration model Production configures DeploymentConfiguration
  root {
    << ... >>
  }
```

Debug FM



```
Variability from FM: DeploymentConfiguration
Rendering Mode: variant rendering config: Production
module ApplicationModule from test.ex.cc.fm imports SensorModule {


  exported test case testVar {
    int8_t x = getSensorValue(1);
    assert(2) x == 10;
    int8_t ww = 22;
    assert(4) ww == 22;
  } testVar(test case)

  int32_t main(int32_t argc, string[ ] args) {
    return test testVar;
  } main (function)

}
```



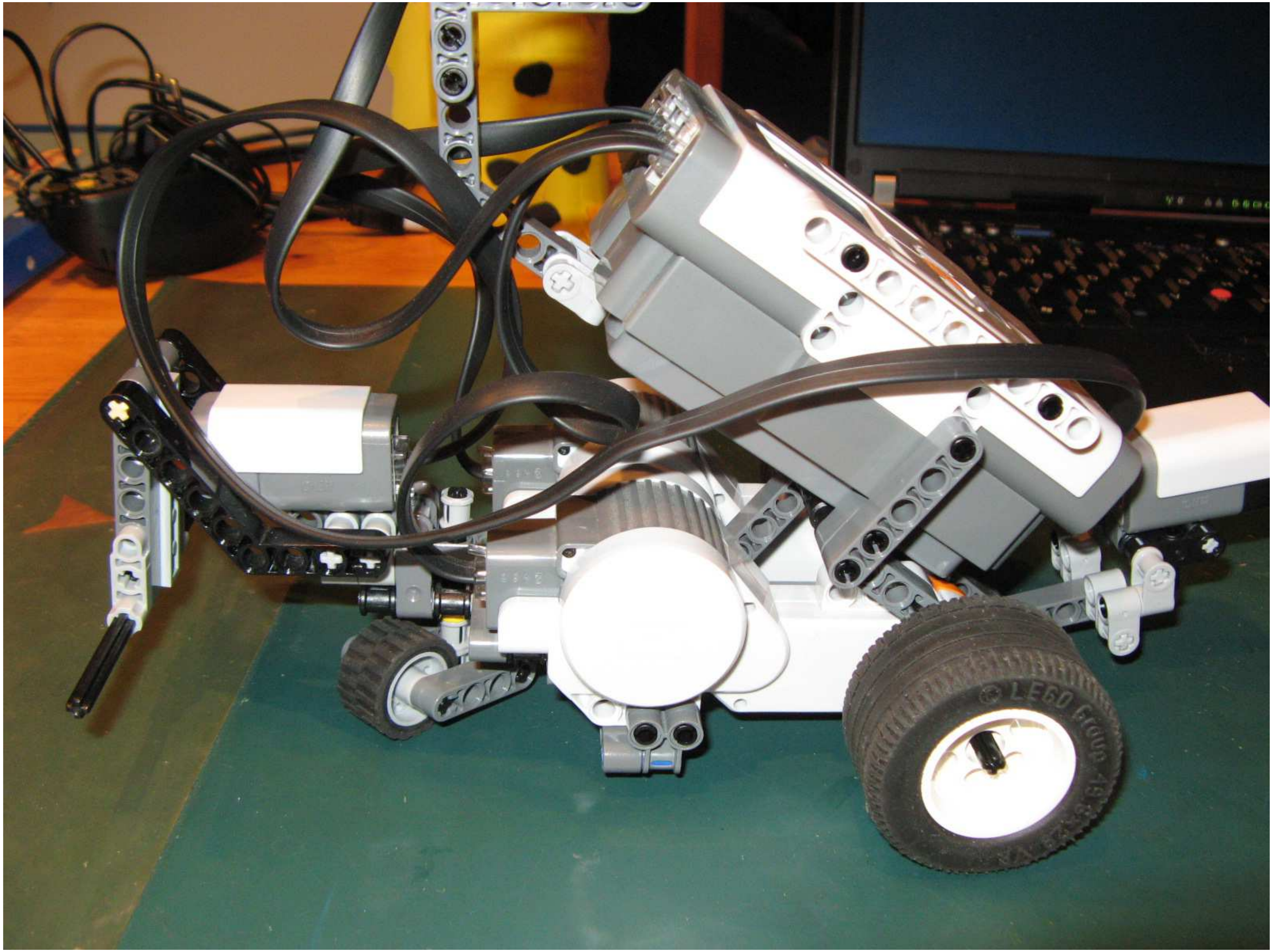
```
feature model DeploymentConfiguration
  root ? {
    logging
    test
    valueTest [int8_t value]
  }

configuration model Debug configures DeploymentConfiguration
  root {
    logging
    test
    valueTest [value = 42]
  }

configuration model Production configures DeploymentConfiguration
  root {
    << ... >>
  }
```

Production FM

Status and Availability

http://mbeddr.com

It runs on Lego/NXT with OSEK RTOS


© LEGO Group

and on our first sensors…

C is not dead !

U don´t

C it

But it's still there...

Thank you !