NordiaSoft

Why SCA for Heterogeneous Embedded Distributed Systems (HEDS)?

A Technical Perspective using a Simple Example

What is the SCA?

- The SCA is a Component-based Development architecture
  - **Operating System Independent** compared to Microsoft's CBD ".NET" framework
  - **Programming language Independent** compared to Sun Microsystem's CBD "EJB" framework
  - **Device-oriented** compared to OMG's CBD "CCM" framework
- SCA applications are portable across various hardware platforms
  - SCA enables platform abstraction
  - Supports wide range of **operating environments (OE)**
    - processors, operating systems, libraries, etc.

Why the SCA?

- Decreases application development time for Heterogeneous Embedded Distributed Systems (HEDS)
  - **Component framework allowing 3rd party involvement**
    - Defines a standard Life-Cycle for all components
    - Standardizes how components get connected to other components
    - Standardizes how the behavior of components can be influenced
    - Defines how applications are installed, deployed, and controlled
  - **Support for distributed interactions between components**
    - Handles requests/replies interactions as well as publish/subscribe interactions
    - Supports concurrent synchronous and asynchronous interactions
  - **Support for heterogenous environments**
    - Supports standalone executables, shared libraries, tasks, threads, etc.
    - Supports core assignment, priority settings, host-colocation, address-space co-location
    - Supports several transports (ethernet, shared memory, PCIe, etc.)
    - Supports several programming languages (C/C++, Java, Python, etc.)

Why the SCA?

- To allow applications and components to be reused across a wide range of operating environments
  - Reusability decreases development time and time to market
  - Open architecture for plug-and-play of software components allows easy integration of 3rd party components
- Can you avoid using the SCA for HEDS?
  - Do you know what is required to write software for HEDS?
  - Does your team have the required expertise?
  - Do you have the time to implement infrastructure equivalent to SCA?
  - Are you going to maintain your own infrastructure?
  - Will you develop tools to support your developers?
- Let's look at what the SCA addresses…

Simple Example

A single-threaded program that produces a tone

```
#include <iostream>
#include "AudioDeviceNativeImpl.hh"
#include "FMModulator_IQ.hh"
#include "FMDemodulator_IQ.hh"
#include "GeneratorAnalog.hh"
#include "Resampler.hh"

const double SINE_TONE           = 500.0;   //Hz
const double AUDIO_SAMPLING_RATE = 8000.0;  //Hz
const double IF_SAMPLING_RATE    = 48000.0; //Hz

using namespace std;

int main(int argc, char *arv[])
{
  GeneratorAnalog sineWave;
  FMModulator_IQ fmModulator_IQ;
  FMDemodulator_IQ fmDemodulator_IQ;
  Resampler resamplerMod;
  Resampler resamplerDemod;
  AudioDeviceNativeImpl audioDevice;

  // CONFIGURE
  sineWave.setF(SINE_TONE);
  sineWave.setFs(AUDIO_SAMPLING_RATE);
  sineWave.setAmplitude(1.0);
  sineWave.setMethod(DIRECT_METHOD);
  sineWave.setStartingPhase(0.0);

  resamplerMod.setDataFormat(REAL_DATA);
  resamplerMod.setInterpolationMode(INTERPOLATION_MODE_LINEAR);
  resamplerMod.setPendingAdjustment(0.0);
  resamplerMod.setInputSampleRate(AUDIO_SAMPLING_RATE);
  resamplerMod.setOutputSampleRate(IF_SAMPLING_RATE);

  fmModulator_IQ.setFs(IF_SAMPLING_RATE);
  fmModulator_IQ.setIndex(5000);
  fmDemodulator_IQ.setFs(IF_SAMPLING_RATE);
  fmDemodulator_IQ.setIndex(5000);

  resamplerDemod.setDataFormat(REAL_DATA);
  resamplerDemod.setInterpolationMode(INTERPOLATION_MODE_LINEAR);
  resamplerDemod.setPendingAdjustment(0.0);
  resamplerDemod.setInputSampleRate(IF_SAMPLING_RATE);
  resamplerDemod.setOutputSampleRate(AUDIO_SAMPLING_RATE);

  audioDevice.setAgcInput(false);
  audioDevice.setSampleSpecification(PA_SAMPLE_S16LE, AUDIO_SAMPLING_RATE, 1);
  audioDevice.initialize("PA_SRC", "PA_DTS");
  audioDevice.flushAudio();

  //Ask user for the desired tone frequency to be heard on the AudioDevice
  double prompt = 0.0;
  cout << endl << "*****************************************" << endl;
  cout << "Enter the desired tone frequency (40 - " << (AUDIO_SAMPLING_RATE/2) << "): ";
  cout << endl << "*****************************************" << endl;
  cin >> prompt;
  cout << endl;
  if((prompt<40) || (prompt>(AUDIO_SAMPLING_RATE/2)))
  {
    cout << "Invalid frequency." << endl;
    return 0;
  }
  sineWave.setF(prompt);

  //SIGNAL PROCESSING
  cout << "Transmitting data" << endl;
  int counter = 0;
  while(counter < 100)
  {
    /***************/
    // Tx Path
    /***************/

    //Generate SineTone
    long sineWaveOutputDataSize = 2000;
    double* sineWaveOutputData = NULL;
    sineWave.generateReal(sineWaveOutputDataSize, sineWaveOutputData);

    //Resample data, AUDIO_SAMPLING_RATE -> IF_SAMPLING_RATE
    int resamplerModDataSize = resamplerMod.getOutputSize(sineWaveOutputDataSize);
    double* resamplerModData = new double[resamplerModDataSize];
    resamplerMod.processReal(sineWaveOutputData,
                             sineWaveOutputDataSize,
                             resamplerModData,
                             &resamplerModDataSize);

    //Perform FM Modulation
    unsigned int fmModIqBufferSize = resamplerModDataSize * 2;
    double* fmModOutputDataIQ = new double[fmModIqBufferSize]();
    double* fmModOutputDataI = fmModOutputDataIQ;
    double* fmModOutputDataQ = fmModOutputDataIQ + resamplerModDataSize;
    fmModulator_IQ.modulateFM(resamplerModData,
                              resamplerModDataSize,
                              fmModOutputDataI,
                              fmModOutputDataQ);
    delete [] resamplerModData;

    //////////////////////
    /////  USRP //////////
    //////////////////////
```

Simple Example

A single-threaded program that produces a tone (cont)

```
102
103     /***************/
104     // Rx Path
105     /***************/
106
107     //Perform FM Demodulation
108     unsigned int fmDemodOutputDataSize = 0;
109     double* fmDemodOutputData = NULL;
110     fmDemodulator_IQ.demodulateFM(fmModOutputDataI,
111                                   fmModOutputDataQ,
112                                   fmModIqBufferSize/2,
113                                   fmDemodOutputData,
114                                   fmDemodOutputDataSize);
115     delete [] fmModOutputDataIQ;
116
117     //Resample data, IF_SAMPLING_RATE -> AUDIO_SAMPLING_RATE
118     int resamplerDemodDataSize = resamplerDemod.getOutputSize(fmDemodOutputDataSize);
119     double* resamplerDemodData = new double[resamplerDemodDataSize];
120     resamplerDemod.processReal(fmDemodOutputData,
121                                fmDemodOutputDataSize,
122                                resamplerDemodData,
123                                &resamplerDemodDataSize);
124
125     //Write Data to AudioDevice
126     audioDevice.write(resamplerDemodData, resamplerDemodDataSize);
127     delete [] resamplerDemodData;
128
129     cout << "." << std::flush;
130     if((counter>0) && ((counter%10) == 0))
131     {
132       cout << "(" << counter << "%)" << std::flush;
133     }
134     counter++;
135   } // while(counter < 100)
136
137   cerr << "(100%)" << endl << "End of program!" << endl;
138
139   return 0;
140 }  // main()
```

Simple Example

Single threaded program

- Characteristics: a single program space invoking several data processing functions sequentially
  1. Single pipeline of data, the DSP functions are called one after the other
  2. Only one processor can be used
  3. If the processor is multi-core symmetric-multi-processing (SMP), you can still get suboptimum performances because of cache misses when the program instructions are moved across cores

What can we do to increase performance and portability ?

Simple Example

Single threaded program

Time	DSP function 1	DSP function 2	DSP function 3	DSP function 4
t0	Packet #1			
t1		Packet #1		
t2			Packet #1	
t3				Packet #1
t4	Packet #2			
t5		Packet #2		
t6			Packet #2	
t7				Packet #2

- Needs 4 units of time to process Packet #1.
- Would process **4 packets** in **16 units of time**.

Simple Example

Use of a multi-core processor

- **Benefit:** some level of parallel processing
  1. Possibility to have several threads working at the same time and ability to control cache misses
  2. Possibility to have a multi-stage pipeline of data
     - i.e. at any one time, 'n' packets are being processed at the same time
  3. Can decide which threads run on which core (big.LITTLE architecture)
  4. Can turn off some cores (or reduce clock speeds)
- **Impact**
  1. Modify source code to add multi-cores related business logic
     - Syntax for core control/assignment/affinity is different with almost every operating system

Simple Example

Use of a multi-core processor

Time	DSP function 1	DSP function 2	DSP function 3	DSP function 4
t0	Packet #1	IDLE	IDLE	IDLE
t1	Packet #2	Packet #1	IDLE	IDLE
t2	Packet #3	Packet #2	Packet #1	IDLE
t3	Packet #4	Packet #3	Packet #2	Packet #1
t4	Packet #5	Packet #4	Packet #3	Packet #2
t5	Packet #6	Packet #5	Packet #4	Packet #3
t6	Packet #7	Packet #6	Packet #5	Packet #4

- Could run each DSP function on a different core
- Could process 4 packets in only 6 units of time, and 1 more packet each next unit of time.
- This means **14 packets** in **16 units of time**!

Simple Example

Homogeneous multiple processors

- **Benefit:** Even more parallel processing via multiple processors.

- **Impact**
  1. Source code must be split into several programs
  2. Cannot only rely on thread-level synchronization anymore
  3. Must choose a transport mechanism to synchronize and exchange data between different programs, all components use the same transport
  4. Must define and implement a control protocol for data exchange
     - Convert function calls into message buffers
     - Implement the message creators
     - Implement the message parsers
     - Implement a message passing infrastructure
       a. Synchronous or asynchronous?
       b. Synchronous with the middleware, the transport, the server, or the handler?
       c. Support multiple invocations at the same time or not?

Simple Example

Homogeneous multiple processors

- Impact (cont)

5. Define a way to coordinate the launch of an application across several processors
   - Write special-purpose scripts for each program to launch?
   - Implement a generic launcher?

6. Define how applications must be installed; files are across different file systems of different processors
   - Copy files into each mass storage via ftp-like service?
   - Create a federated distributed file system like NFS?

Simple Example

Heterogeneous multiple processors

- **Benefit:** Support for different kind of processors which maximizes performances of various parts of the application
- Impact
  1. Source code must be split into several programs
  2. Need to encode/decode data for endianness and different representations
  3. Define and use a control protocol for data exchange
     - Convert function calls into message buffers
     - Support different transports (shared memory between some processors, backplanes between some, Ethernet between others, etc.)
     - Implement the message creators
     - Implement the message parsers
     - Implement a message passing infrastructure that works across various operating environments
       a. Synchronous or asynchronous?
       b. Synchronous with the middleware, the transport, the server, or the handler?
       c. Support multiple invocations at the same time or not?

Simple Example

Heterogeneous multiple processors

- Impact (cont)
  4. Need to be able to launch an application made of pieces potentially developed using different tool chains for different processors
     - Write special-purpose scripts for each program to launch?
     - How are the scripts coordinated?
  5. Define how application files must be installed across different file systems of different processors
     - Copy files into each mass storage via ftp-like service?
     - Create a federated distributed file system like NFS?

Why the SCA?

- Using the SCA, applications are implemented using a single approach that supports all of the above use cases
  - The solution is the same with one single process space or with multiple process spaces running on a single processor or on multiple processors of the same kind or not
  - The level of parallelism matches the possibilities of the platform
- The way SCA applications are installed, launched and controlled is the same for any type of platform
  - The SCA offers the possibility to create an app store for embedded systems with applications made of several pieces targeting the various processors of an integrated platform
  - An SCA app can be installed remotely (i.e. from a store) if desired

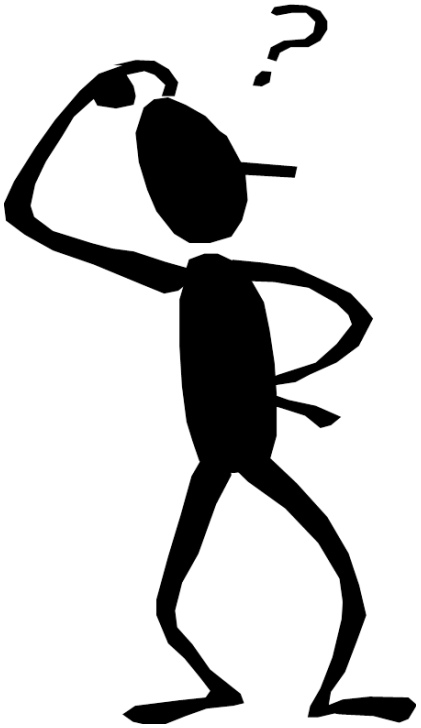
Why the SCA?

- The SCA supports Heterogeneous Embedded Distributed Systems
  - Supports systems with multiple processors with multiple buses just as well as systems made of a single processor
- The SCA offers a component-based development architecture to maximize software reuse
  - Components
    - Small units of functionality fulfilling a well-defined role that support plug-and-play of third party functionality
  - Framework
    - Infrastructure that supports deployment of applications for heterogeneous distributed embedded Systems
  - Multi-transport middleware with location transparency

Summary

- The SCA Decreases application development time for Heterogeneous Embedded Distributed Systems (HEDS)
- The SCA is a Component-based Development architecture
- It maximizes portability of applications across various platforms
- Supports systems with multiple different processors just as well as systems made of a single processor

Why do we need a solution for HEDS?



Let's take a look at the
2017 Embedded Market Survey

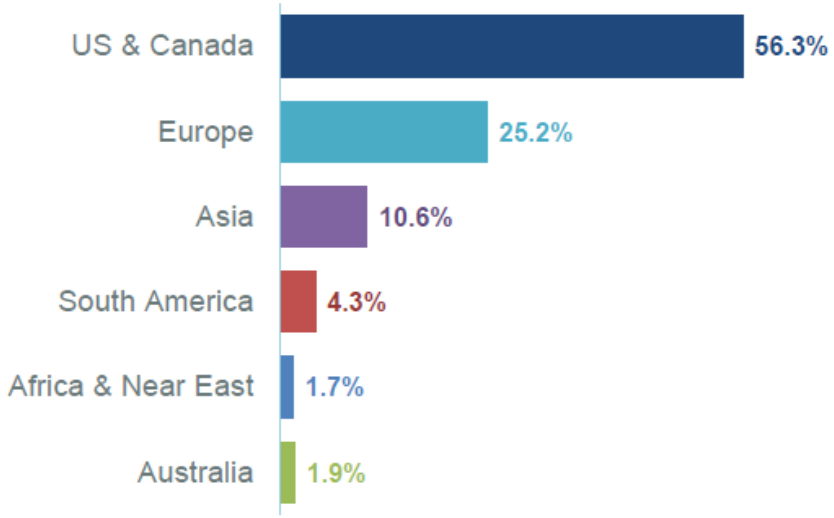
EE Times

embedded
cracking the code to systems development

Why do we need a solution for HEDS?

EETimes/embedded – April 2017

- Why do we need a solution for HEDS?*
  - Markets worldwide web-based survey



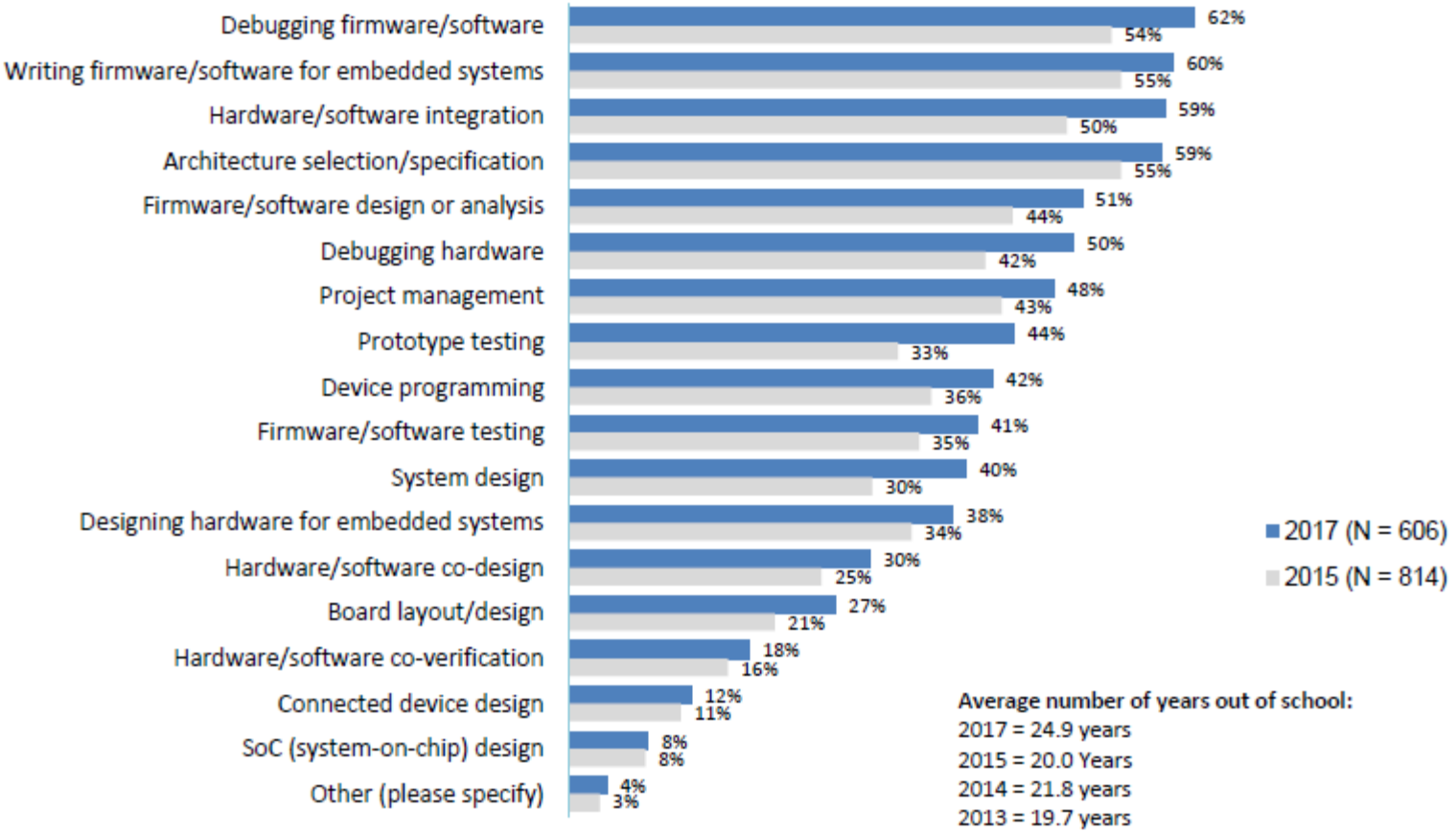
  - Developed and implemented by Wilson Research Group
  - Conducted from February 20, 2017 to April 15, 2017
  - Overall confidence of 95% ± 2.8%

*Source: www.embedded.com/electronics-blogs/embedded-market-surveys/4458724/2018-Embedded-Market-Survey

Why do we need a solution for HEDS?

EETimes/embedded – April 2017

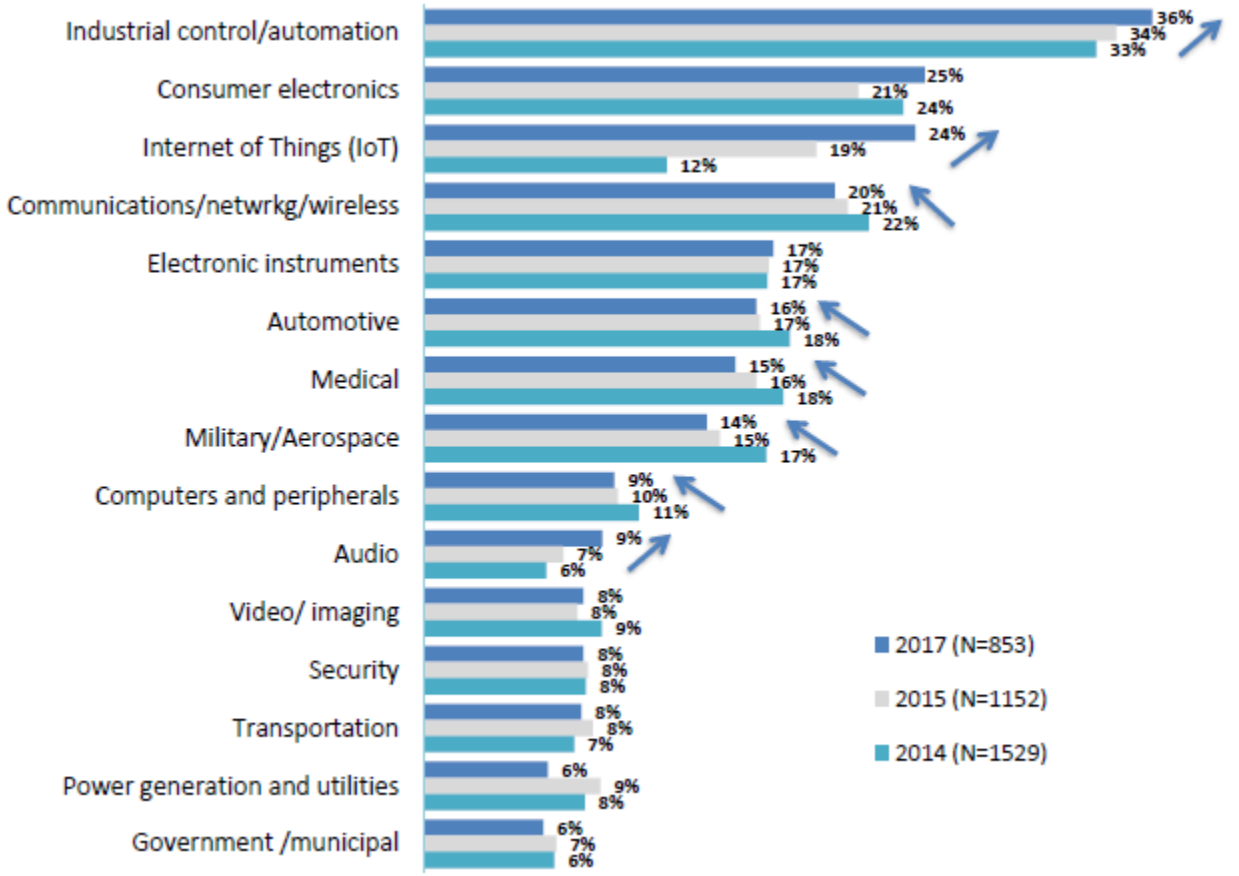
- Job functions of respondents



Why do we need a solution for HEDS?

EETimes/embedded – April 2017

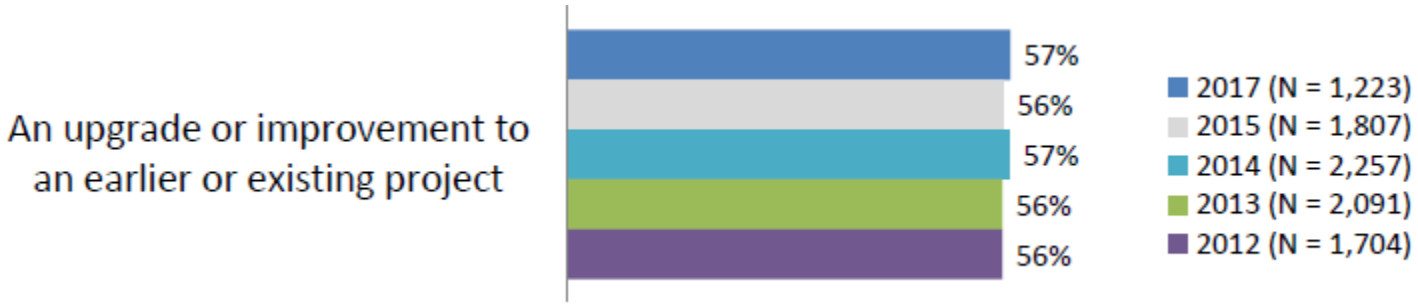
- What type of embedded systems are you developing?



Why do we need a solution for HEDS?

EETimes/embedded – April 2017

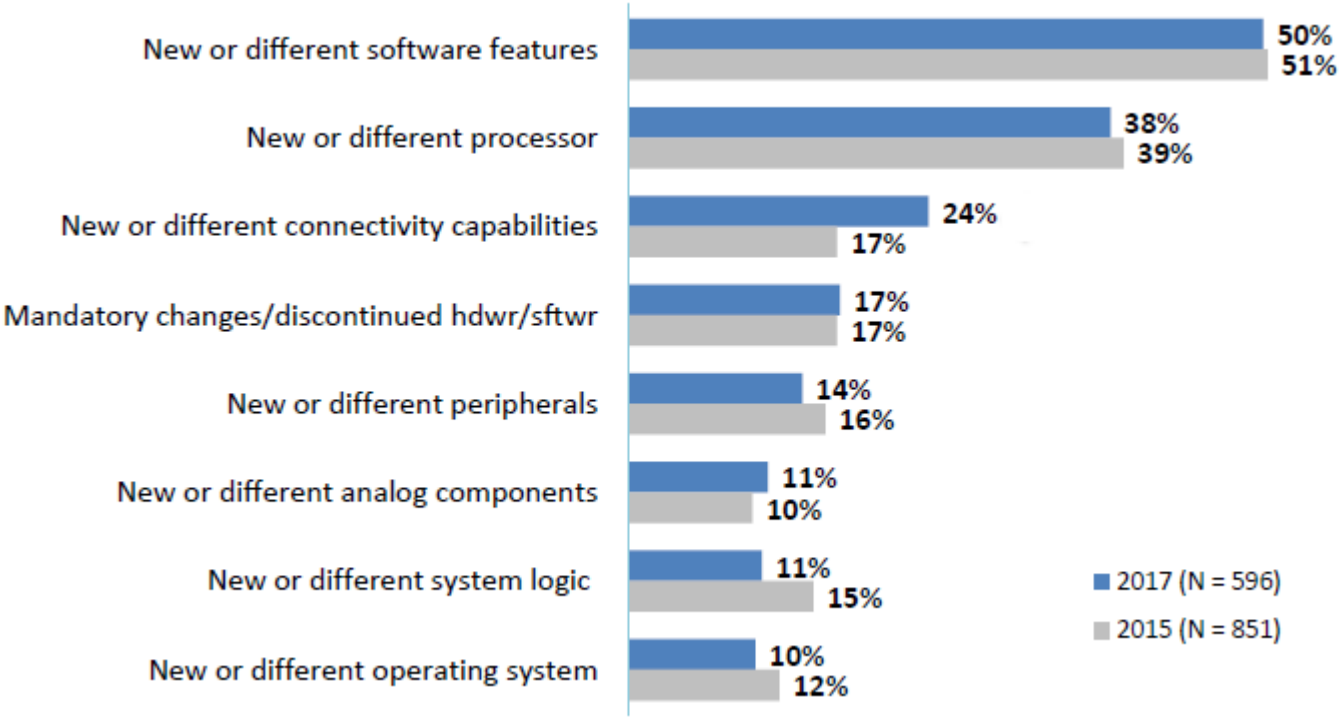
- My current embedded project is…



Why do we need a solution for HEDS?

EETimes/embedded – April 2017

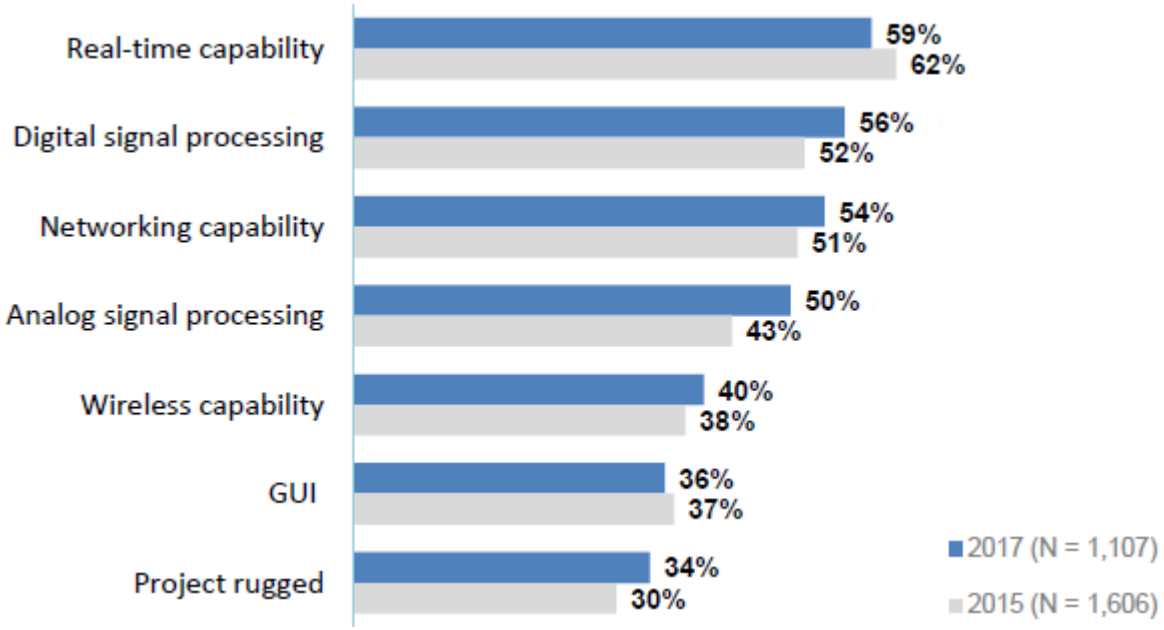
- What does the upgrade include?



Why do we need a solution for HEDS?

EETimes/embedded – April 2017

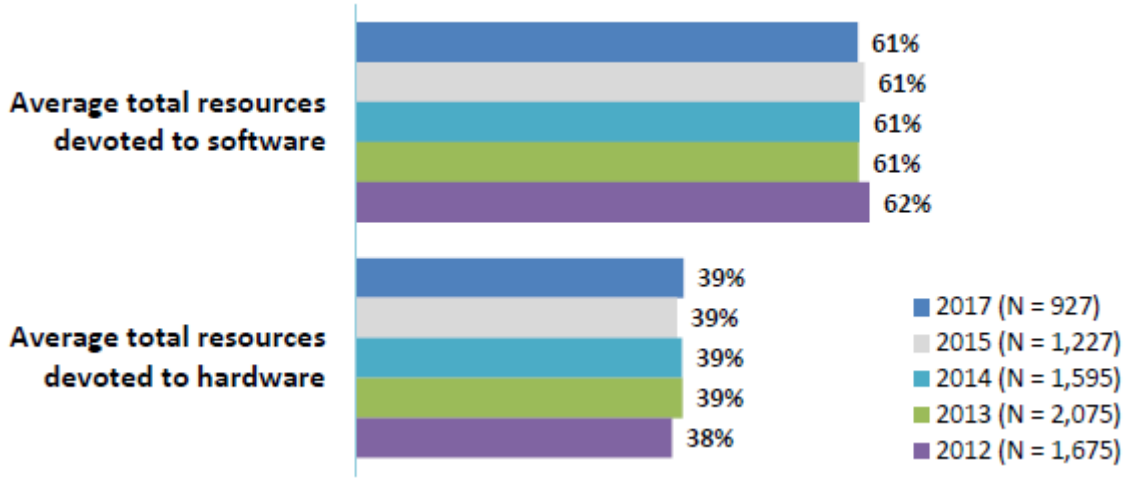
- Which of the following capabilities are included in your current embedded project ?



Why do we need a solution for HEDS?

EETimes/embedded – April 2017

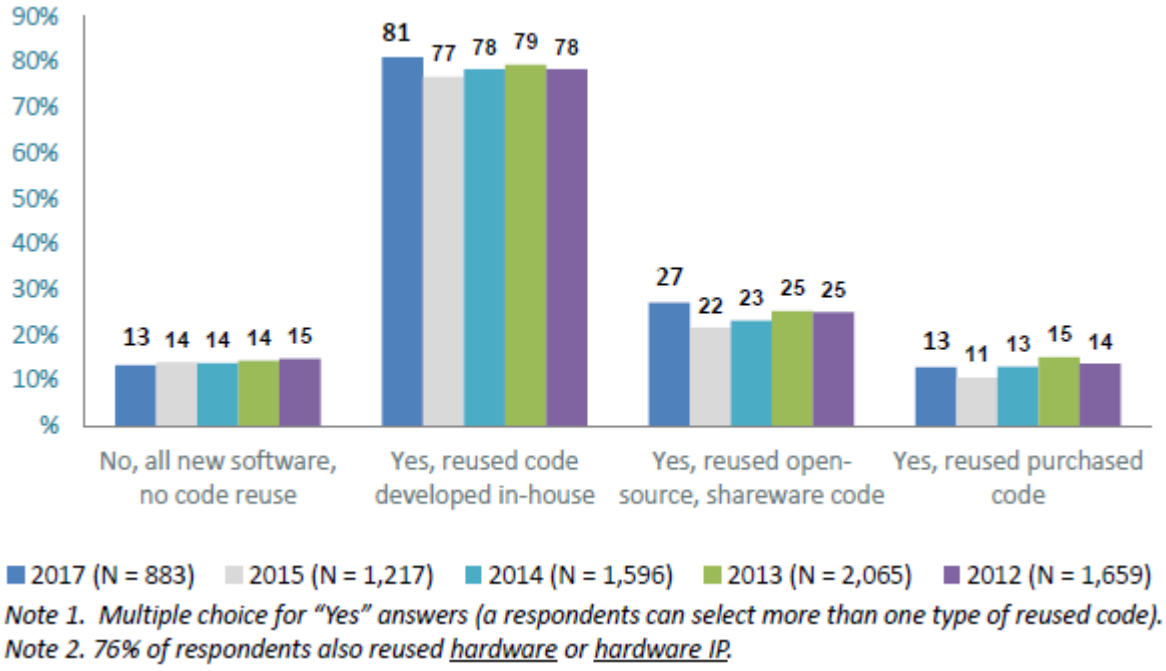
- Software vs. Hardware



Why do we need a solution for HEDS?

EETimes/embedded – April 2017

- Which boards are you currently using, and consider using?

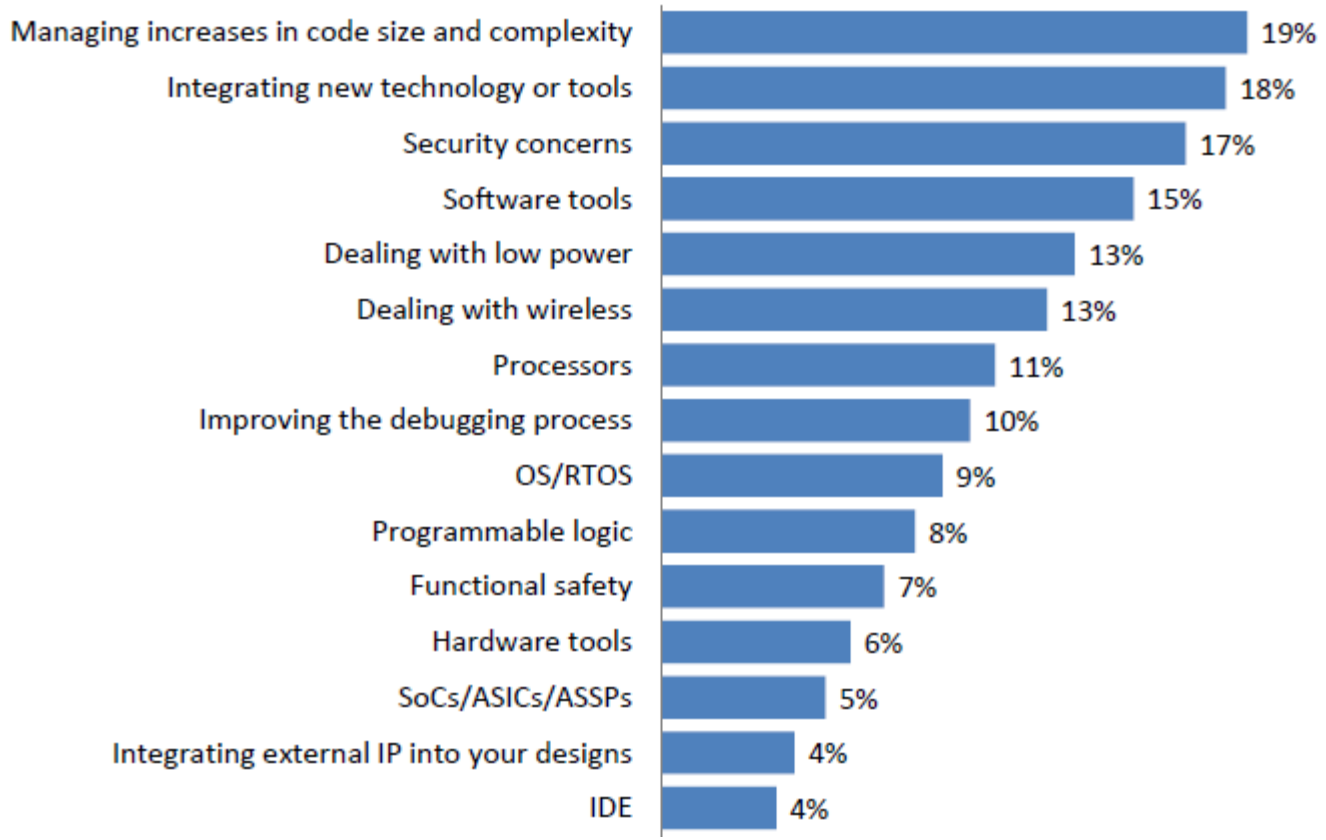


Note 1. Multiple choice for "Yes" answers (a respondents can select more than one type of reused code).
Note 2. 76% of respondents also reused hardware or hardware IP.

Why do we need a solution for HEDS?

EETimes/embedded – April 2017

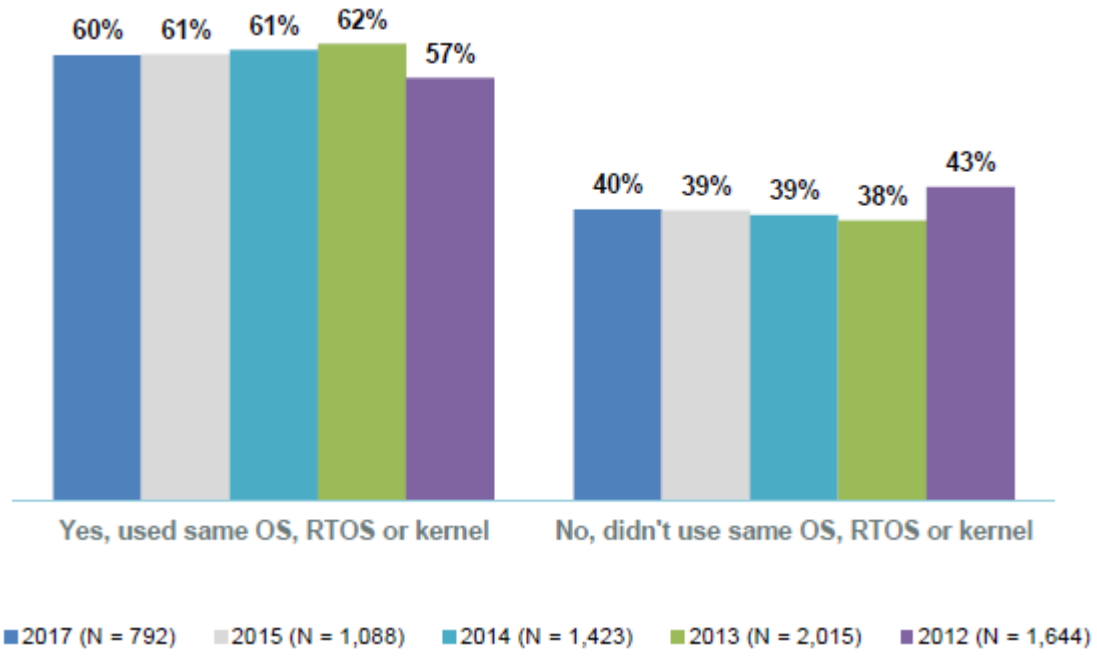
- What will be your greatest challenge next year?



Why do we need a solution for HEDS?

EETimes/embedded – April 2017

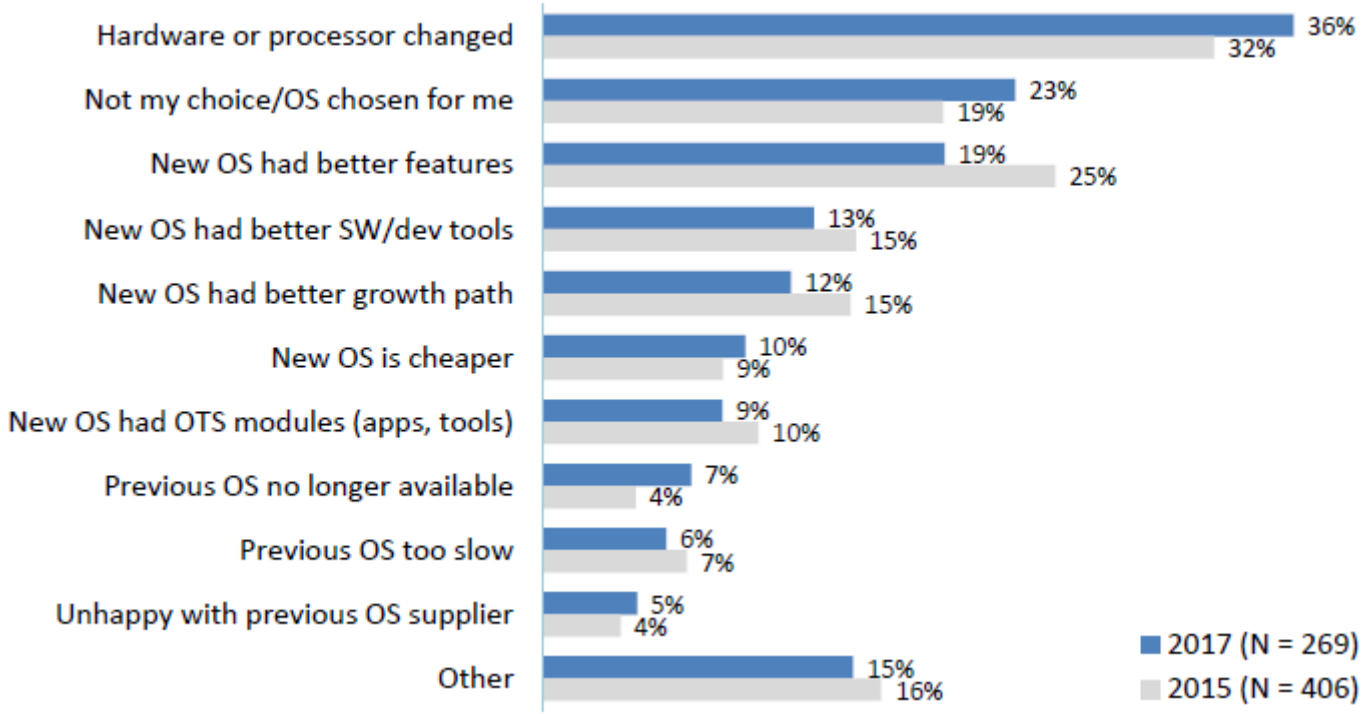
- Did you use the same RTOS as in your previous project?



Why do we need a solution for HEDS?

EETimes/embedded – April 2017

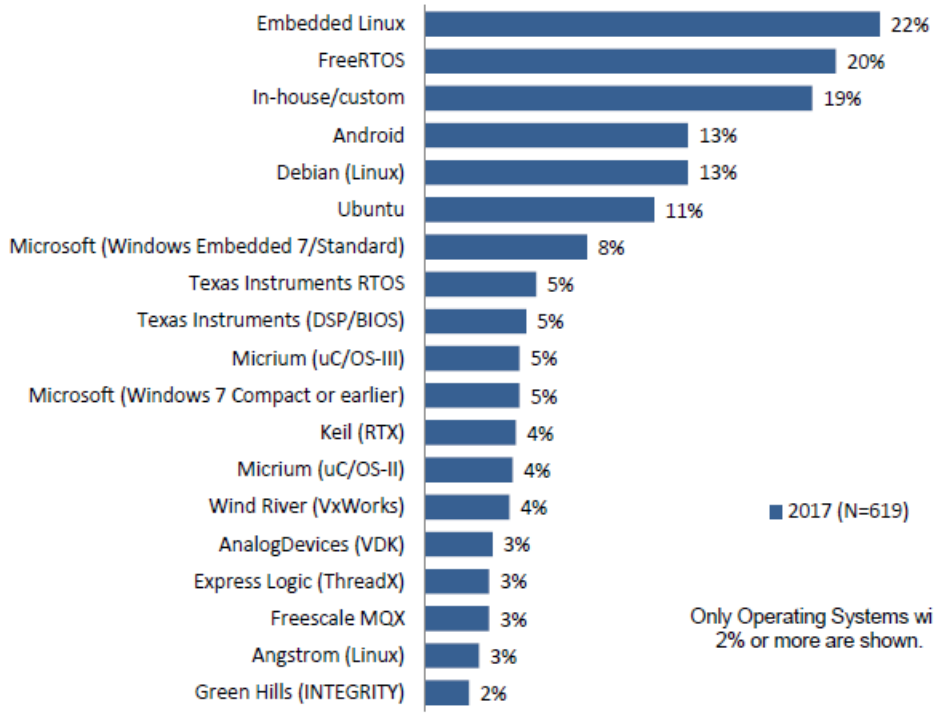
- Why did you change RTOS?



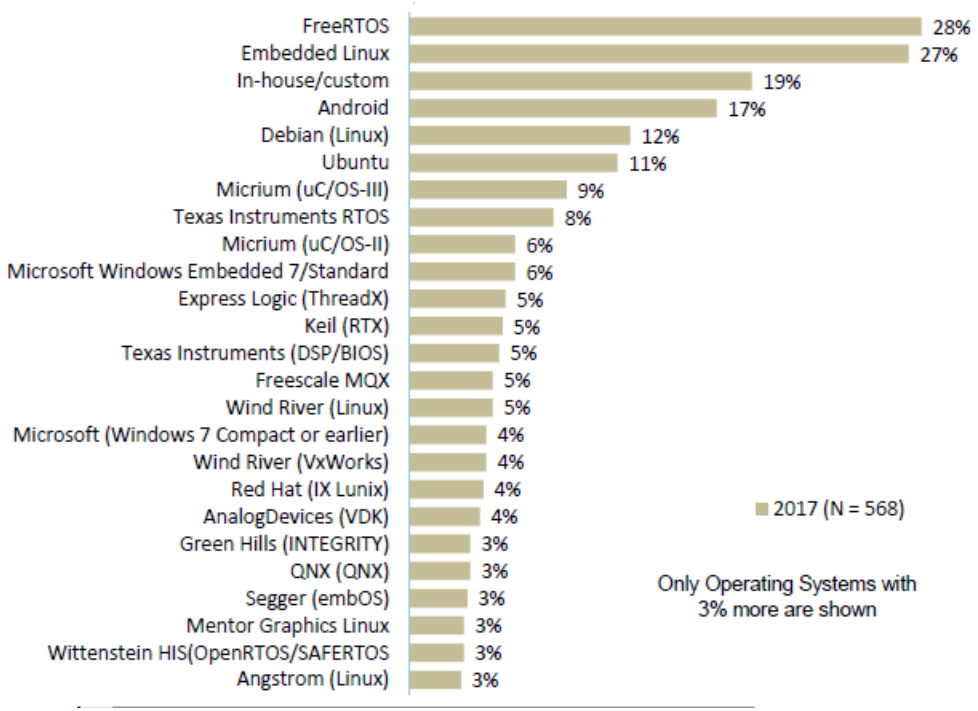
Why do we need a solution for HEDS?

EETimes/embedded – April 2017

Which RTOS are you using?



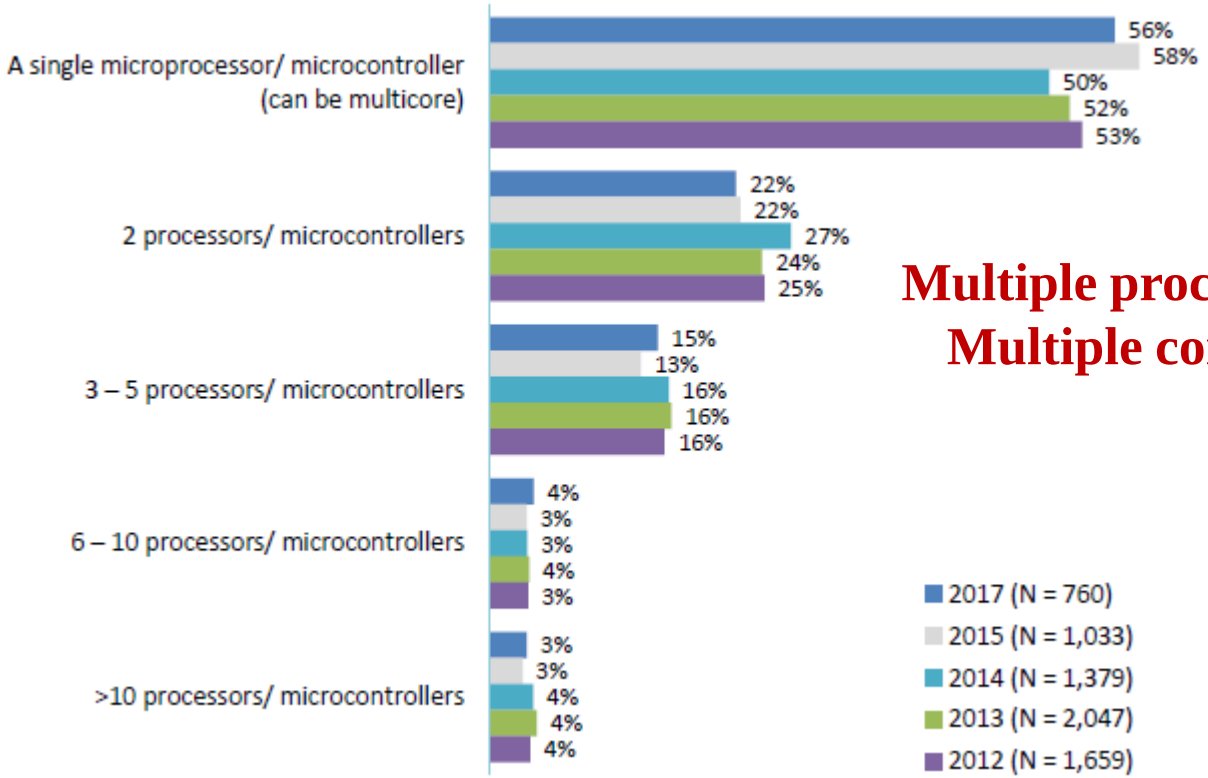
Which RTOS will you be using next?



Why do we need a solution for HEDS?

EETimes/embedded – April 2017

- My current project contains:

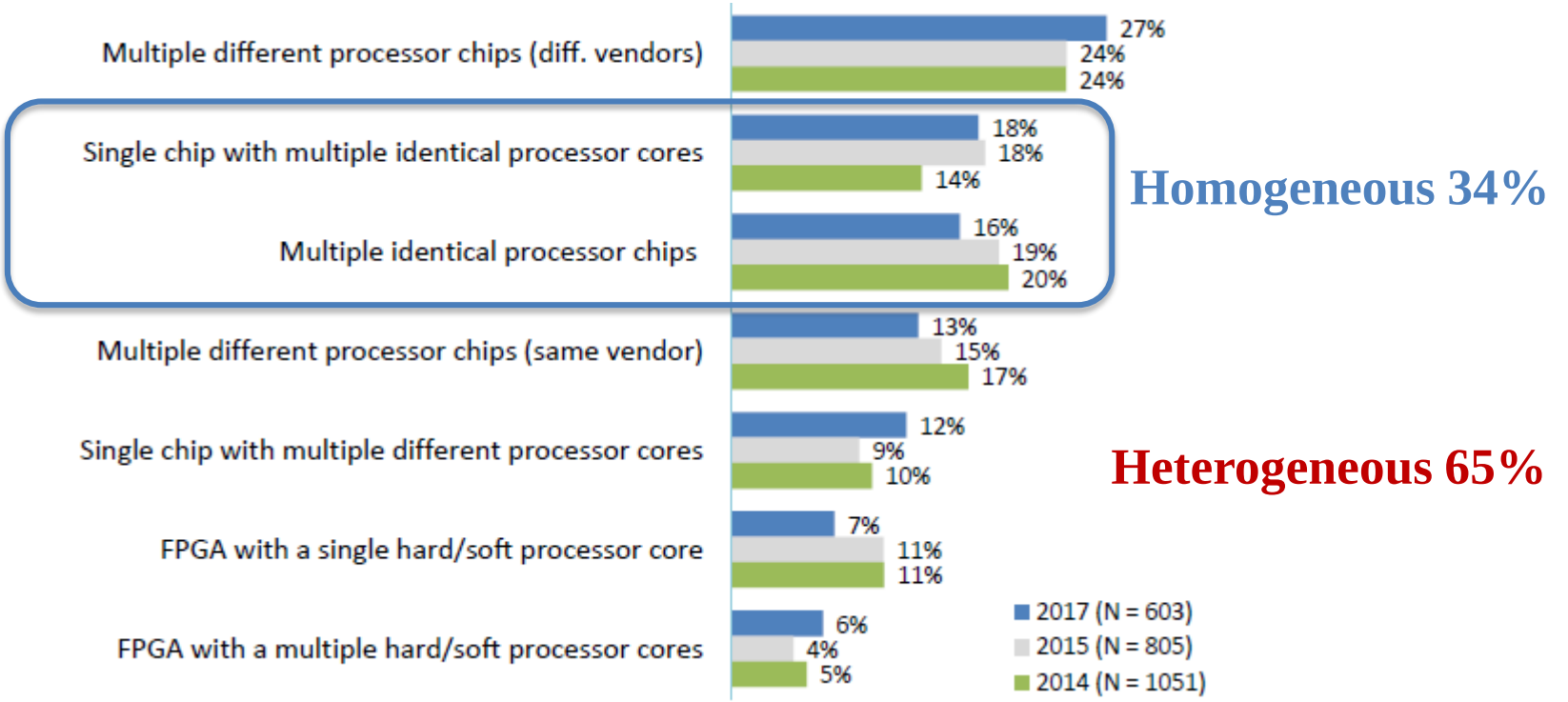


Multiple processors 44%
Multiple cores >44%

Why do we need a solution for HEDS?

EETimes/embedded – April 2017

- My current project contains:



Why do we need a solution for HEDS?

EETimes/embedded – April 2017

- Key take away?
  - **Upgrade vs. New** – 56% upgrades, 44% projects
  - **Upgrades include new:**
    - software features
    - processors
    - connectivity
  - **Capabilities:**
    - Real time 59%
    - signal processing 56%
    - networking capabilities 54%
  - **Resources used for:** Software 61%, hardware 39%
  - **Source code reuse:** +81%
  - **Changing the RTOS:** 40% of the projects and it is mostly due to new processors and corporate decisions

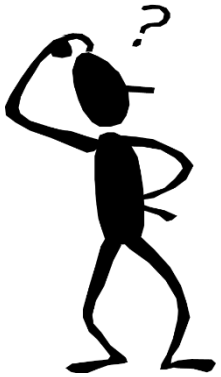
Why do we need a solution for HEDS?

EETimes/embedded – April 2017

- Key take away? (cont)
  - **Greatest challenge:** dealing with increase in code size and complexity
  - **Use Multiple processors:** 44% of systems
  - **Use Multiple cores:** over 44% of systems
  - **Create heterogeneous systems:** 65% of the systems

Why the SCA?

- In short, most projects use multiple processors/cores, integrate new processors and new operating systems, require new forms of connectivity, need real-time performances, involve an increased amount of software, and spend most of their resources on software which explains why software reuse is so important
- Can any organization afford not to use an infrastructure like the SCA to design and develop next generation embedded systems?



NordiaSoft can help you build your next Software-Defined platform

info@NordiaSoft.com