# Teaching for Evolution towards Embedded Multi-sensor Interfaces

Zeljko Zilic
McGill University
Department of ECE
Monteal, Canada
e-mail: zeljko.zilic@mcgill.ca

Boris Karajica
ST Microelectronics
Geographical Business Unit
Lexington, USA
e-mail: boris.karajica@st.com

*Abstract*—**This paper outlines experiences in bringing up a course with significant sensing and actuating capabilities, on top of the already existing infrastructure support for training embedded wireless system design. The role of the suitable development platform, design kits and the lab experiments is outlined, and the expected outcomes are highlighted. We emphasize the role of scaffolding principle, which now does not only apply to the single course, but also to our overall experience in developing such courses.**

***Keywords-embedded systems teaching, heterogeneous microsystems, sensors, wireless sensors networks***

## I. INTRODUCTION

The progress in the embedded systems platforms has been unabated in the last several decades, facilitated with the system miniaturization and integration capabilities, and limited only with the ability to imagine new classes of products that integrate computation, communication and the capabilities in sensing and actuation.

While computing and communication has been readily integrated in standard microelectronics products, integration of sensing has required breakthroughs in technologies beyond microelectronics. Progress in MEMS or microfluidics devices and their integration has facilitated versatile and integrated sensing, measurement and instrumentation subsystems. The introduction of such heterogeneous microsystems provides many opportunities in building more integrated and intelligent embedded systems.

Training of students for such integrated microsystems requires careful adjustment to current curricula. Even if there are the existing courses dealing with networked embedded systems, most notably, wireless sensor networks, they have not, to a large degree, employed capable modern sensors. The challenges of introducing this last component cannot be overestimated. In system-level design area, the arithmetic imprecision inherent in complex mechanical sensors requires careful offsetting in software. Then, there is the real-time design and, increasingly, real-time debug challenge, not to mention a need for sophisticated signal processing or control system schemes.

## II. BACKGROUND

Computer Engineering curriculum is a fast moving target. After the initial IEEE Task Force on Computer Engineering Curricula produced report in Dec. of 2004, many new networked devices and applications have captured imagination of consumers, investors, developers and society as a whole, especially in the social networking sphere. That progress by itself needs to be taken into account when upgrading the curriculum. There is much focus now of extending the networking capabilities to the physical environment, i.e., towards "cyber-physical" world. In embedded systems education, the focus shifts to the incorporation of interfaces to the physical world, i.e., modern sensors and the actuators that let us control much more of the environment in much more integrated, networked way.

The key challenge then is in teaching the system design with a wide range of sensing and actuating interfaces, while still covering ground in "classical" embedded systems topics from the computer engineering point of view, and sometimes even the general-purpose computing system education. This goal would not be possible without an effective multi-sensor platform, equipped with sufficient development and debug tools. Furthermore, the newly introduced and relatively alien (to the much of Computer Engineering) technical issues in introducing sensors and actuators now need to be covered well, but not at the expense of the rest of the embedded system design education, all within the already limited credit counts. Hence, new progress in curriculum has to be made in an extremely measured way. The following section reviews how we reached this point.

## III. ROAD TO NETWORKED, DEEPLY PHYSICALLY-EMBEDDED MICROPROCESSOR SYSTEMS

### A. Phase I: Microprocessor Systems

Not so long ago, the courses on microprocessor systems were added to EE (and sometimes Mech. Eng.) curricula, to cover in a single course the basics of using microprocessors as well provide for the extensive laboratory and project work. Those courses were intensive, requiring considerable self-learning (as they were both first and last courses on the topic and include difficult lab practice), but have also gained popularity among the students, in spite of heavy workload. The calculation on the part of the students having this course as an elective course was based on the fact that the payoff in exciting and well-paid jobs was just around the corner. Students have realized that experiences in projects such as music players, bar-code readers or sonar-equipped gadgets and the development of skills to realize such ambitious

embedded systems has opened many doors straight upon their graduation.

The problem, unlike majority of courses in EE, was that the half-period for knowledge "dissolution" of 3 years meant that once the laboratory setup was introduced, it became quickly obsolete and out of touch with the topics dealing with processor architecture, which moved quickly ahead. Additionally, the obvious limitations of earlier kits, in terms of not having non-volatile memory within a processor, bulky equipment (students needed to carry large cardboxes with microprocessor kits with them) and rudimentary tools were taking tall in being able to deliver ever-more impressive projects, which was a trend taken for granted.

### *B. Phase II: Modernizing Microprocessor Kits*

The establishment of Computer Engineering programs has placed more expectations, but not much more room in curriculum, for teaching microprocessor systems. Since computer engineering advances rapidly across the board, the space had been taken by numerous courses required for CE upbringing. At that stage, making the most of the laboratory equipment has become even more of a necessity.

Since the course might have stayed the first and the last course on microprocessor systems, the kit had to rely on the processors that are easy enough for a first contact with microprocessors and expandable enough to facilitate a set of lab exercises on microprocessor hardware and software interfacing, and to guard against obsolescence, in the light of the rapid introduction of standardized interfaces and various peripherals. Furthermore, non-volatile memory inside processor has become a must for productivity in the lab.

Extensive searches for and the original designs [1] of suitable microprocessor kits led to the selection of embedded microcontrollers, such as those based on TI's MSP430 series. They were simple enough, yet they utilized most modern concepts, except for the microarchitectural features, such as pipelining.

### *C. Phase III: Expanding on (Wireless) Connectivity*

Finally, the important trend of networking the embedded systems, especially by wireless means, has created need for introducing this component to the already demanding course [2]. Creative restructuring, such that the networking interfaces are introduced as early as possible as instances of IO interfacing, to replace earlier IO that was inconsequential to the networking. Test interfaces such as JTAG and their role, as well as the wireless debug and monitoring [3] were of especial importance in covering the topics without much time being wasted. A variety of wireless interfaces were deployed in the lab [4], with the aim of exposing interfacing at various levels of abstraction. Similar developments were reported elsewhere [5].

## IV. MICROSYSTEM KIT REQUIREMENTS AND SPECIFICATION - iNEMO

In keeping with the overall goal of training for Microsystems design, the required laboratory aid should have:

- Modern and sufficiently capable processor to perform the required signal processing.
- Significant number of advanced sensors integrated on board, using a variety of interfacing standards.
- State-of-the-art tools: IDEs including debug interfaces that not only allow the start-and-stop examination of program state, but also real-time tracing and debug capabilities.

### *A. iNEMO Kit*

The iNEMO™ is an ARM Cortex M3-based module that combines accelerometers, gyroscopes and magnetometers with pressure and temperature sensors to provide 3-axis sensing of linear, angular and magnetic motion, complemented with temperature and barometer/altitude readings, to make up a 10 degrees of freedom (DOF) platform.

The iNEMO integrates five STMicroelectronics sensors: a 2-axis roll-and-pitch gyroscope, a 1-axis yaw gyroscope, a 6-axis geomagnetic module, a pressure sensor, and a temperature sensor.

This 10-DOF inertial system is a complete hardware platform that can be used in many applications, such as robotics, gaming, location-based service and human machine interfaces. A versatile set of communication interfaces with various power supply options in a tiny board (~4 x 4 cm) makes the iNEMO a flexible and open demonstration platform. The block diagram of the iNEMO board is shown in Figure 1. The processor used in the board is the top model in a very capable family of ARM Cortex M3 processors, running at 72MHz, and executing the Thumb2 ISA in a 3-stage pipelined manner. In addition to a variety of embedded system interfaces, this processor contains 512kB of Flash memory, 64kB of SRAM and a variety of debug support, which we find very useful for education of future embedded systems engineers.

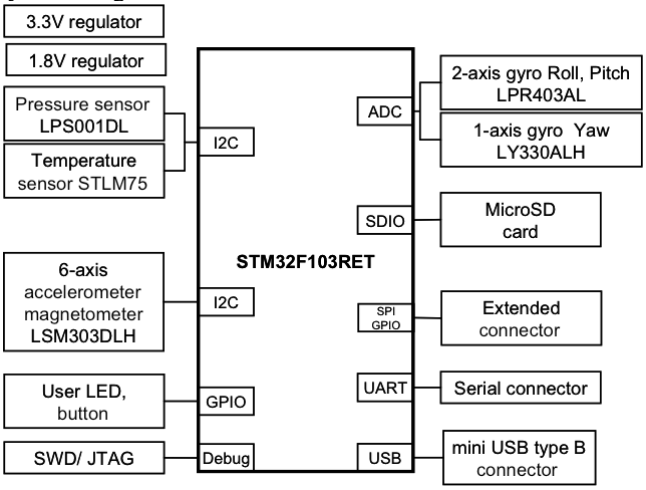


**Figure 1: iNEMO Block Diagram**

The board, measuring 42x 38 mm, and is actually as wide as a JTAG connector, so instead the programming and debugging is done via the newer SWD interface with much smaller pin pitch and less signals taken out, but the full

JTAG functionality can be achieved if needed even at that pin count. The top side of the board is shown in Figure 2.

### B. Availability of Sensors

The variety and advanced functionality of sensors is critical to providing suitable training, as some low-quality temperature sensors and other poor offerings of the recent past do not allow good exposure to the important issues and complexity in utilizing modern sensors. The iNEMO board is an extremely good choice in that sense, as it includes state-of-the-art, high-precision, temperature-stable and compensated sensors for a number of physical quantities that are challenging to capture.

First, there is the LMS303DLH sensor that includes in a single package two advanced devices: a 3D accelerometer and a 3D magnetometer, controlled independently (including powering down) and equipped with sophisticated digital interfaces that perform A/D conversion, significant processing on-chip and overally simplify software interfacing. The sensor communicates via $I^2C$ interface, requiring a minimal number of pins. Two gyroscopes complement the motion sensors set by providing readouts of angular rates of rotations in all three dimensions. They are indispensable in control of moving objects, such as those used in robotic applications.

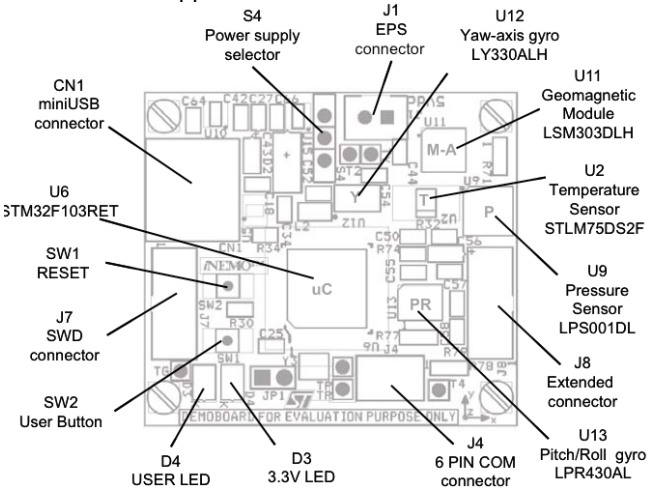


**Figure 2: Layout of iNEMO Board: Top Side**

## V. Lab Experiment Structure

The laboratory exercises that use iNEMO need to be devised carefully to achieve embedded systems teaching goals within a single course.

### A. Embedded Software Including Assembly Code

The introductory lab experiment is best performed on the simulation model of an ISA. We have maintained the emphasis in this experiment on realizing a key part of the experiment in assembly code. Even through the modern assemblers have made great progress, by requiring a non-trivial assembly code exposes students to the ISA of modern processor in a way that cannot be replicated from high-level language entries. In the best case, one can try disassembling the compiled code, but our aim here is to let students pass through the thought process of finding the best mapping of the problem into the stream of processor instructions.

For example, giving an assignment such as the packed BCD arithmetic implementation, together with handling the sign and overflow bits forces the thought process of finding efficient code. This is especially true for architectures, such as ARM, where there is no native BCD arithmetic. Further, by insisting on efficient ARM assembly programming, we expose the useful concepts such as conditional execution, instruction reordering, and a number of others, depending on how much time one can afford.

Further, by programming in assembler, students obtain the unique chance to identify numerous techniques that they already used in C language, this time closer to hardware, and to appreciate how low level C language actually is and how much hardware control one can in the end obtain from C.

The second part of the same lab exercise consist of writing the main program (e.g., a Babbage machine implementation using packed BCD arithmetic), and compiling and linking C and assembly components. Students need to understand the calling conventions, linker and its nuances. Processes of arriving from the source code to the executable program is now exposed, and the students can concentrate on the rest of the process of realizing networked embedded systems armed with multiple sensors.

### B. Using the Sensors - Basics

The second lab experiment serves the role of enabling students to use productively the hardware kit, to dispel any fears and also to protect their work and the equipment through proper handling.

In this experiment, the first peripheral needs to be attached and used in software, and for this purpose, an experiment based on SPI, $I^2C$ or similar interfaces conducted. Basic polled IO processing needs to be employed as, at this stage, interrupt-driven processing is too demanding for students, but with the proliferation of multithread way of reasoning, and the asynchronous execution, we anticipate that interrupts will become more natural to absorb. Even now, writing interrupt handlers purely in C language is straightforward.

For instance, connecting a 3D accelerometer via $I^2C$ is useful for understanding and managing well sensor interfacing, but also getting a feeling for the required signal processing that will be needed for such sensor interfaces. At this stage, we do not require complex processing, such as adaptive filtering, and will rely on the very basic control theory background and the intuitive notions of achieving stability of the response by tuning the gain, for instance, or adjusting the decision thresholds in applications such as free fall detection, where the accelerometer in all three dimension should be close to zero, or in tilt measurements, where simple geometric processing suffices to detect the tilt of the object. Simple games could be built even at this stage by these components to get students more motivated to fine-tune the system using the applied concepts.

Of course, interfacing the accelerometer requires solid, yet concise explanation of how an accelerometer, such as

ST's LSM303DLH, operates and is interfaced. We believe that a short "inline" description embedded in the lab handout suffices, but some lecture or tutorial time spent on the device definitely speeds up the execution.

Since processing the accelerometer data for purposes of deducing orientation (tilt) requires trigonometric functions and their inverses, nominally defined as floating-point values, while the sensor provides signed integer data, this lab provides further possibility to explore the relevant arithmetic processing techniques, and we guide them in developing specific techniques (e.g., CORDIC, function approximation) that are a hallmark of embedded microsystems, where the data collected requires significant arithmetic operations.

### C. Processing and Fusing Sensor Data

The third laboratory exercise, imagined as a mini-project, continues in advancing both computer interfacing and sensor data processing skills. At this stage, interrupt-driven processing arrives naturally, as there might be two sets of independent sensing, plus some processing that need to be multiplexed in time via interrupt-driven processing.

Regarding sensor processing, fusing data from two sets of sensors, such as accelerometers and magnetometers, requires also more sophisticated processing, especially with noise taken into account. At this stage, two independent streams of sensor data get merged in the enhanced signal useful for higher-level processing of physical data. Lab experiment then prescribes to design solutions such as a) tilt-compensated compass that adjusts the magnetic field reading by taking into account the orientation of the device or b) a combination of system movement control by merging accelerometer and gyroscope data via sufficiently robust filtering, such as Kalman filter, from a library of signal processing blocks, such as the one available already by ST Microelectronics. Again, there is a lot of possibility to practicing suitable arithmetic computations in this setup, including CORDIC, function approximations, and we again use scaffolding principle to achieve more in the end.

### D. Project: Multi-sensory System Design

The final project relies on the scaffolding principle of reusing the developments from previous labs, similar to our earlier developments in microprocessor [1] and embedded wireless system courses [4]. The focus now shifts in creating conditions for teamwork contributing to achieving an ambitious integrated microsystems design project, which will finally cement the buildup of required skills and the efficient team project execution. All members of the team must contribute fully in order to achieve the results, and the close monitoring by instructor is a must here.

One new element that is introduced is that of networking, mostly by wireless means, by adding interfaces such as Zigbee or Bluetooth. At this stage, as in the past, speed of the development is impacted by the ability to debug systems employing wireless networking channels, and we rely on our previous experiences, enhanced by in-house aids, such as the beacon stations [4], JTAG over wireless [3] or "packet sniffers" that are instrumental in passing the hurdle of achieving the first communication. Relative to previous courses, more emphasis is now required on filtering and adaptive signal processing for packet networks [7].

TABLE I. TOPICS COVERED

| Lab | Skills Developed | | |
|---|---|---|---|
| | ***Core Embedded Systems*** | ***Interfaces, Networking*** | ***Microsystems*** |
| #1 | ISA,interfacing Assembly-C, ISA simulator, arithmetics, debugging basics | - | - |
| #2 | I2C interfacing, devices, floating point, real-time debugging (SWD) | I2C, JTAG, SWD | Accelerometer , calibration |
| #3 | Fast interrupt processing, concurent/asynchronous debug techniques | Processing in threads, dual I2C | Fusing accel. magnetometer |
| Proj. | System-level microsystem design, RTOS, teamwork | SPI, Zigbee, Bluetooth | Fusion, Kalman filtering |

## VI. CONCLUSIONS

This paper presents an evolution in teaching courses on practical microprocessor system design. Outlined is the planning needed to achieve ever-growing teaching goals, together with the suitable teaching aids. In a significant way, the current inclusion of multi-sensory data processing is showing the full potential of the applied computer engineering to the systems deeply embedded in physical environments. In mixed teams of computer, electrical and software engineering students, the course achieves impressive crowning of all of their gathered skills. Students have reacted very positively, but the official evaluation results are not available at the time of writing of this paper.


## ACKNOWLEDGMENTS

The authors are grateful to Bob Boys and Angela Williams from ARM for numerous discussions, and for the licenses of Keil MDK IDE.


## REFERENCES


[1] J-S. Chenard, A. U. Khalid, M. Prokic, R. Zhang, A. Chattopadyay and Z. Zilic, "Expandable and Robust Laboratory for Microprocessor Systems," *Proc. Microelectronics Systems Education Conf.*, 2005.

[2] Z. Zilic, J-S. Chenard and M. Prokic, " A Laboratory for Wireless and Mobile Embedded Systems", *Proc. Conference on Microelectronics Systems Education*, MSE'07, pp. 103-104, May 2007.

[3] M. W. Chiang, Z. Zilic, J-S. Chenard and K. Radecka, "Architectures of Increased Availability Wireless Sensor Network Nodes", *Proc. IEEE Intl. Test Conference*, ITC 04, pp. 1232-1241, Oct. 2004.

[4] J-S. Chenard, Z. Zilic and M. Prokic, "Laboratory for Wireless and Mobile Embedded Systems", *IEEE Transactions on Education*, Vol. 51, No. 3, Aug. 2008, pp. 378-384.

[5] P. Frantz, C. Garnier, E. Welsh, and A. Valenzuela, "*Microcontroller and embedded systems laboratory,*" 2005 [Online]. Available: http://cnx.rice.edu/content/col10215/latest

[6] ST Microelectronics, iNEMO, www.st/inemo.

[7] J. Radecki, Z. Zilic and K. Radecka, "Echo Cancellation in IP Networks", *Proceedings of IEEE International Midwest Symposium on Circuits and Systems*, pp. 219-222, Aug. 2002.