AC 2009-774: A COST-EFFECTIVE, MODULAR-HARDWARE PLATFORM FOR EMBEDDED SYSTEMS DESIGN AND DEVELOPMENT

Eduardo Montanez, Freescale Semiconductor

Eduardo Montanez received a BS degree in Electrical Engineering with a technical concentration in Computer Engineering and Integrated Electronics from The University of Texas at Austin. Eduardo works for Freescale Semiconductor as a Systems Engineer in the Microcontroller Solutions Group where he defines microcontrollers and complimentary software and hardware ecosystem solutions to meet customer requirements for various industrial and consumer markets.

Michael Norman, Freescale Semiconductor

Michael Norman earned his BS degree in Electrical and Computer Engineering from The University of Texas at Austin. He currently works for Freescale Semiconductor as a Systems Engineer where he concentrates on microcontroller and microprocessor systems solutions, hardware and software development tools, and reference designs.

A Cost Effective, Modular Hardware Platform for Embedded Systems Design and Development

Abstract

Embedded systems such as automobiles, mobile phones, appliances, and other consumer products continue to advance along a steep curve of complexity and sophistication. As this trend continues, the learning curve for tomorrows' engineers grows steeper and the gap between designing embedded systems in industry and teaching embedded systems development at a university widens. Educators are therefore challenged to adapt to advances in embedded systems while maintaining courseware that is broken into simple building blocks capable of maintaining continuity along the growth path. This requires a rich hands-on curriculum that encapsulates modular hardware, software, and courseware that can scale from fundamental concepts to more advanced topics.

This paper introduces a modular demonstration, development and learning hardware platform and an example set of progressive laboratory exercises that help to meet this challenge. The platform includes system building blocks that can be used individually or grouped to build a complete embedded system. The kit includes function-specific hardware modules featuring microcontrollers, serial interfaces, external memories, sensors, wireless communications, and graphical displays with touch interfaces. These hardware modules are complemented with software that abstracts the hardware and provides real-time operating systems, protocol stacks and low-level drivers. The key features of the platform are:

- Modular – Function-specific hardware and software modules
- Affordable – $20-50 hardware modules
- Portable – Easy development at home or the lab
- Scalable – Basic building blocks to complex embedded systems
- Expandable – Standardized hardware interconnect encourages future growth
- Easy to use – Perfect for students of all levels

Introduction

In most universities, curriculum for embedded systems is already broken up into various courses that build on each other and grow in complexity. The main objective for this type of curriculum is for the student to build a wide knowledge base for various types of embedded systems and to learn the fundamental building blocks that make up a complete embedded system. This structured education enables the student to be a well-rounded engineer and more attractive asset in industry.

In theory this type of structured curriculum sounds attractive, but the execution proves to be more difficult. The common problem with achieving a consistent embedded systems curriculum from undergraduate to a graduate level program has to do with the many variables. The variables

include the lack of standards across courses, the little flexibility to keep up with industry changes, and the cost sensitivity for development tools.

The variables begin with the lack of standards due to differences between educators that teach the various courses. This includes differences in pedagogy and their preferences for hardware, software, and textbook material. For example, the selection of microcontroller core makes a substantial difference in classroom material and learning kits. In many instances the student will have to adapt to a different development tool when moving from course to course even though the instructional concepts are a continuation from the previous course. This type of disconnect introduces unnecessary preparation time for the student to familiarize themselves with the new equipment when the time could be used to maximize the learning. This fundamental problem is also found within many high-tech companies. The solution for most companies is to develop standards.

The next variable has to do with a lack of flexibility with the hardware and software development tools put in place in order to keep up with the rapidly changing industry standards. Our research shows that many educators select a microcontroller board with standard components and have little refresh to the course for 5-10 years due to the complexity and cost of having to update tools and courseware. One solution for this common problem is a modular teaching system that can quickly adapt to changes in industry standards.

The last variable is the cost sensitivity of having to outfit and maintain labs with new tools to replace non-functional boards or to refresh a course. The typical model is for universities to purchase tools for these courses and issue them out to students for the duration of the course. This tends to be problematic when different tools need to be purchased for all the courses. The solution for this common problem is a tool that is affordable for the student to purchase and enjoy the benefits of using it in the lab or at home for multiple courses. This helps expand their learning boundaries.

The common solution for these problems is in a modular development and demonstration platform coined the “Tower”. The Tower platform developed by Freescale Semiconductor[1] is the ideal learning kit to simplify embedded systems curriculum because it offers a standard, flexibility, and is affordable. This paper will introduce the Tower and its benefits for structured embedded systems curriculum from course to course in a breakdown of an HVAC application.

Tower Overview

The Tower is a cost effective development and demonstration platform well suited for teaching embedded systems curriculum. The platform consists of modular hardware containing simple building blocks that when used in conjunction models an embedded application. The concept behind the Tower was developed by Freescale engineers to consolidate common circuitry found in traditional boards developed for each new microcontroller or microprocessor. By breaking apart the common circuitry into individual, standardized boards the controller could change quickly and the common peripheral boards could be reused. This concept enables our customers to more quickly evaluate our embedded controllers and prototype their embedded application. This same approach benefits an engineering program by providing a structure of building blocks

that can be introduced individually from course to course—helping to keep hardware cost as low as possible. It also has the flexibility to grow with advances in industry and not become obsolete. Additionally, research shows that laboratories structured around student owned microcontroller boards and complementary hardware components add value to an embedded engineering education[2]. The Tower platform enables this model.

Structurally, the Tower consists of 3 fundamental types of boards:

1. Elevator Boards
2. Microcontroller unit (MCU) or Microprocessor unit (MPU) Story Boards
3. Peripheral Story Boards

The modular boards are named "Elevator" or "Story" boards to define their functionality and placement in the Tower. Each type of board will be described in more detail and matched to a potential structured courseware in the following sections. See Figure 1 for a breakdown of the various components that make up the Tower.

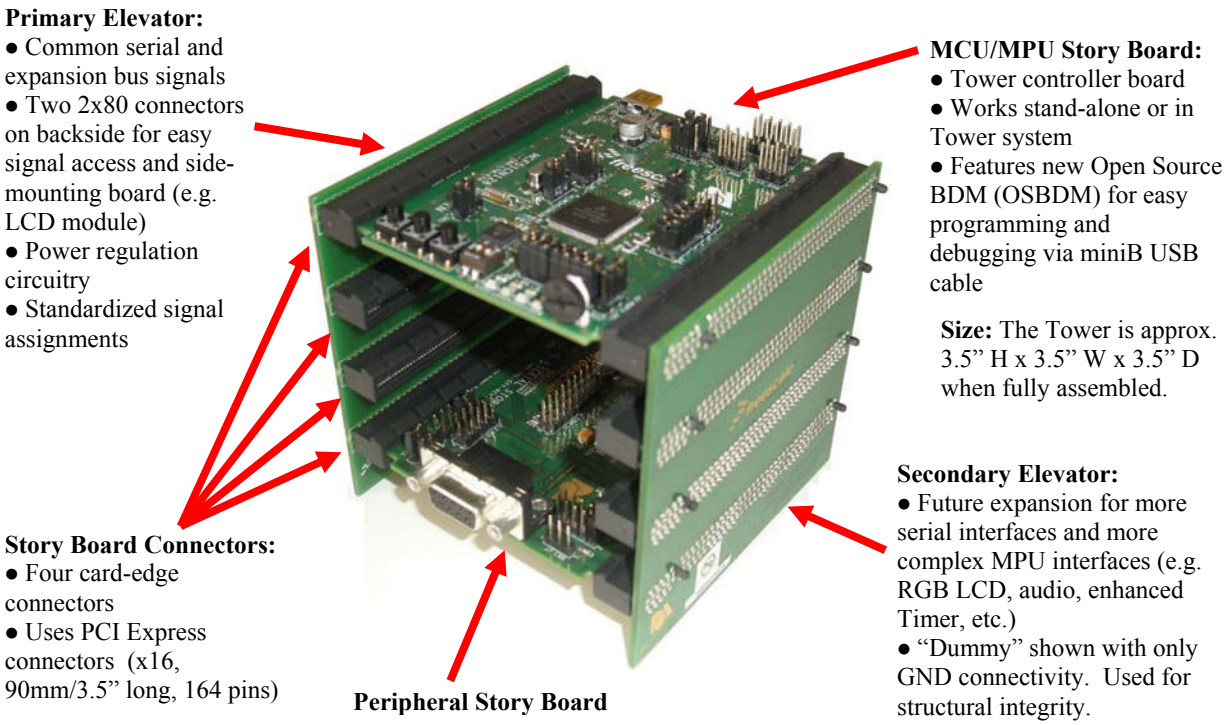


Figure 1. Tower platform

The platform is built around standards that include predefined signals and power assignments that run up and down Elevator boards to four PCI Express connectors. These connectors accept various combinations of Peripheral Story boards with one central MCU/MPU Story board. Some Peripheral Story boards are already developed (e.g. Serial, Memory), others are planned on a roadmap, and more can be custom built by the engineering program or student to fit the Tower form factor and standards depending on the courseware or embedded application for a final design project.

The Tower also includes multiple power options with built-in isolation. The first option is to power the complete platform through a miniB USB cable plugged into the primary elevator and flipping the ON/OFF switch. This power supply outputs 5V and regulated 3.3V. It also enables a power sense signal that overrides other power supply options to prevent damage. The second option is to power the complete platform while enabling debug to the controller through a miniB USB plugged into the MCU/MPU Story board. This power option allows the MCU/MPU Story board to operate standalone from the Tower platform. This suits an introductory to microcontroller's course that focuses on assembly or C/C++ programming and might use some basic components (e.g. push buttons, switches, LEDs, potentiometer, sensors). The last powering option is from the individual Peripheral Story boards. For example, our Serial Story board includes USB device, host, and OTG functionality. If it is used in USB device mode, power to the complete platform can be derived from VBUS on the dedicated USB device port provided by the USB host.

From our research, we have learned that a key educational goal in engineering programs is to expand the learning boundaries beyond the classroom and lab. The Tower system makes this goal more possible with its convenient packaging. Individual Story boards are packaged in convenient DVD carrying cases that include anti-static sleeves to hold the board and associated cables. This rugged packaging design is ideal for students allowing them to pack up their Tower and conveniently throw into their backpack to take home. Along with the hardware all MCU/MPU Story boards include a DVD filled with the following types of tools and collateral.

1. Complimentary development tools
2. Quick Start Guide
3. Lab Document
4. Software examples
5. Schematics
6. MCU/MPU reference material
7. Various cables (USB, Ethernet, etc.)

Elevator Boards

The Elevator boards are the interconnect hardware that brings together the embedded application. The boards map out a standardized bus that includes power rails, digital and analog signals. This bus connects the MCU/MPU Story board controller functionality like SPIs, UARTs, I2C, Ethernet, USB, interrupts, external bus, and GPIO to custom designed Peripheral Story boards. In the development of the Tower platform we researched multiple connector strategies to make up the Elevator boards. We chose to use commodity x16 PCI Express card-edge connectors because of their high density, high performance, and very cost effective. To maximize the types of applications and have room for future growth the Tower contains a primary and secondary Elevator. The primary Elevator contains typical signals found on most MCUs. The idea for the secondary Elevator is to maintain more advanced features more commonly found on MPUs. The current secondary Elevator provided with the platform is a "dummy" Elevator that provides mechanical stability for Story boards plugged in and for additional ground shielding. On the backside of the primary Elevator there are two female berg

connectors with larger pitch known as expansion connectors. These connectors have all the Elevator signals brought out for easy probing using available lab equipment like an oscilloscope, digital multi-meter, or function generator. They also allow convenient hand wiring from Tower to student-built circuits on a protoboard area. These expansion connectors also allow for outward facing Peripheral Story boards like an LCD Story board. See Figure 2 for a breakdown of the different components that make up the Primary Elevator board.

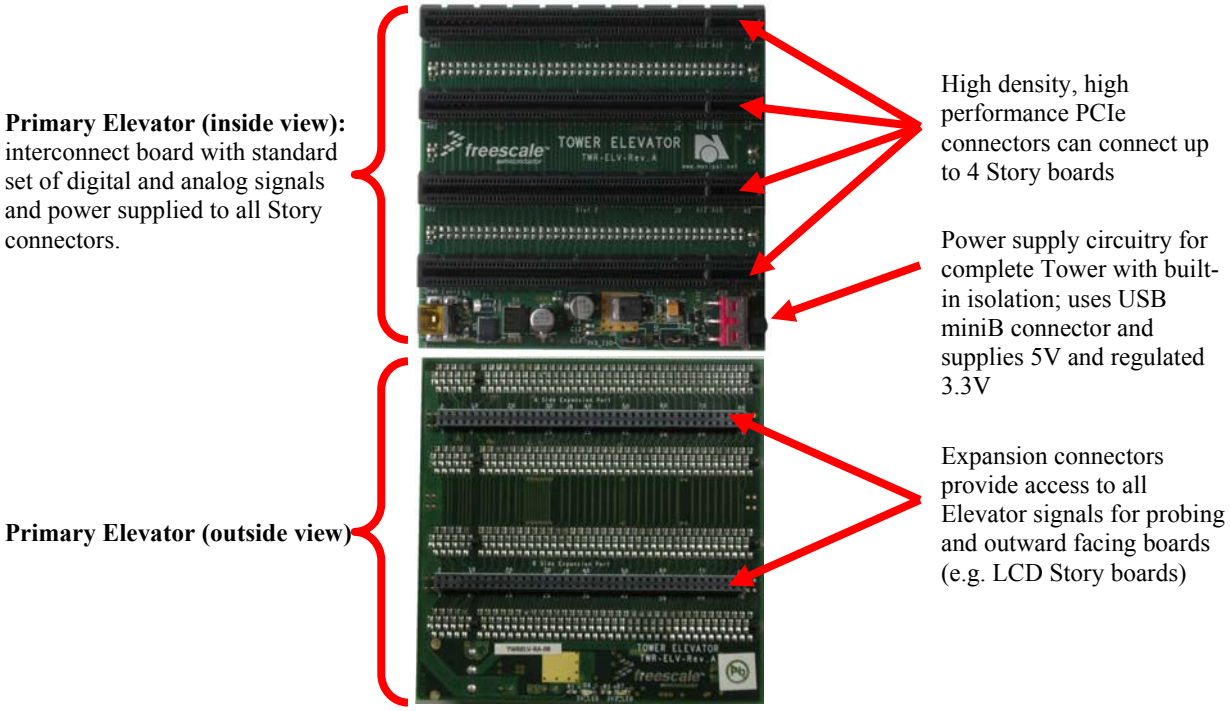


Figure 2. Primary Elevator Board

MCU / MPU Story Boards

The MCU/MPU Story boards are the brains in the Tower platform. The platform plans to support future devices in these Freescale architectures:

- 8-bit RS08 and HCS08 microcontrollers
- 16-bit digital signal controllers (DSC)
- 32-bit ColdFire® microcontrollers/microprocessors
- Power Architecture® technology microprocessors
- i.MX ARM® microprocessors

This wide range of architecture coverage is suitable for introducing students from entry-level controllers found in toys to high-end controllers in an automobile engine control module. The various core options give the professor the flexibility to choose based on personal preference. The MCU/MPU Story boards are designed to work stand-alone as cost effective, entry level tools or in combination with the Tower platform giving curriculum the access to additional Peripheral Story boards. These boards contain the basic connections needed to operate the controller along with simple interface components suitable for a course that focuses on assembly or C/C++

programming using real-time hardware instead of a simulator. These boards are also useful for a mid-level course that focuses on simple controller interfaces.

The MCU/MPU external interface signals are connected to the Tower Elevators using a card edge connector that matches the form factor of standard x16 PCI Express connectors. To develop and debug the application software the board has debug circuitry called Open Source Background Debug Module (OSBDM). This circuitry interfaces the controller to development tools running on the lab or student's computer. OSBDM gives student's access to on-chip CPU, SRAM, Flash and memory mapped registers. With this interface the student can erase and reprogram the controller with their code and send run-time commands for debugging. All the student needs is to connect the board with a provided miniB USB cable to the computer. On top of the debugging features, this same interface powers the board and allows serial transfer of UART data from target controller to a terminal window running on the computer. Last, when the student reaches their final design they can repurpose their MCU/MPU Story board as a standalone programmer to program and debug their application prototype. See Figure 3 for a breakdown of the different components that make up the MCU/MPU Story board.

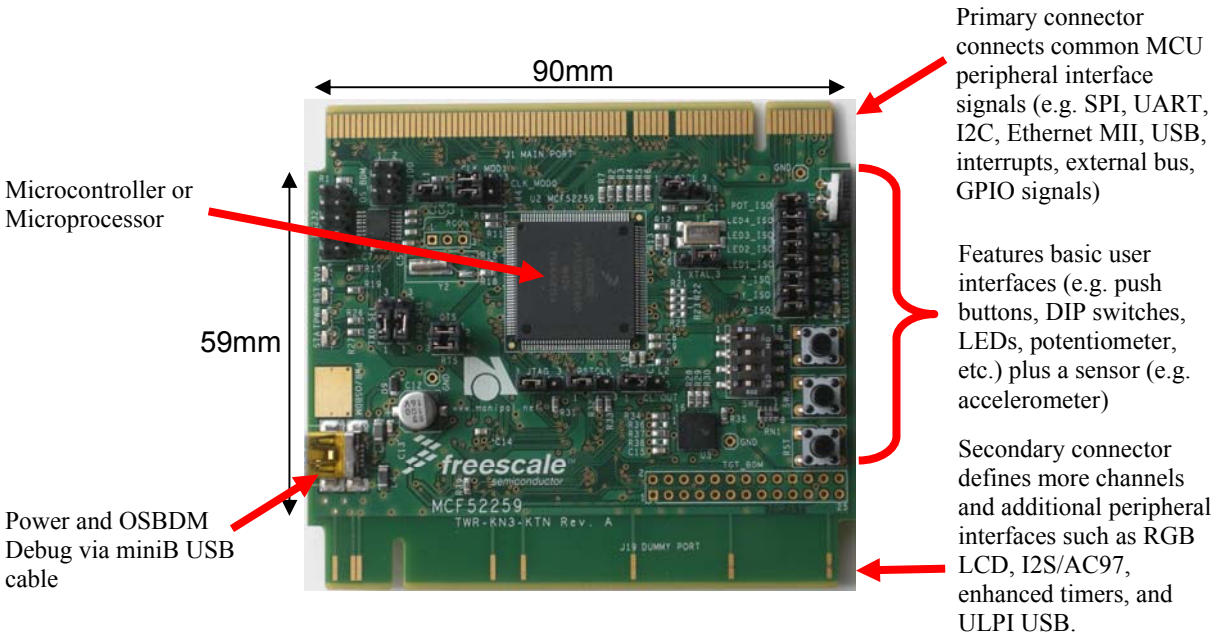


Figure 3. MCU/MPU Story Board

Now the question is "how do you structure curriculum around a standalone MCU/MPU Story board?" Table 1 below lists a structure for example lectures and labs complimented by the features provided on the standalone board.

Lesson No. & Estimated Duration	Lecture & Lab Title	Description
#1 (1 weeks)	Programming with Real-time Hardware	Learn build tools to code simple algorithms using the CPU in assembly or C/C++.
#2 (1 week)	Debugging Techniques	Exercise debugger with OSBDM to troubleshoot code. Take advantage of hardware run-time control, breakpoints, and trace features.
#3 (1 week)	General Purpose Input / Output (GPIO)	Learn GPIO controls to toggle board LEDs and read DIP switch state. Introduce polling procedure.
#4 (2 weeks)	Interrupts	Learn pin and on-chip interrupt functionality and how to service event. Use board push buttons to force interrupt event. For bonus, apply debounce techniques.
#5 (2 weeks)	Timers	Exercise timer and real-time counter (RTC) modules for periodic events, output compare, input capture, and pulse-width modulation. Use board push buttons for force input capture event and drive LEDs with varying pulse-width modulation waveforms. Configure RTC to perform simple scheduling of events.
#6 (2 weeks)	Analog to Digital Converter (ADC)	Configure ADC to continuously convert analog output of board potentiometer (POT). Control LED brightness by varying pulse-width modulation from POT input value.
#7 (2 weeks)	Interfacing to Sensors	Learn a 3-axis accelerometer. Configure ADC to sample X, Y, and Z axes from on-board accelerometer to vary the blinking rate of all board LEDs.
#8 (2 weeks)	Serial Communication	Learn various serial communication protocols and exercise UART to generate a simple “Hello World!” print statement from target to terminal

		running on PC. Use on-board OSBDM serial communication to PC. For bonus, apply protocol to send target commands over serial communication to turn on/off on-board LEDs.

Table 1. Example Structure for Lecture & Labs using MCU/MPU Story Board

This simple course curriculum fits an introductory embedded systems course. These building blocks are later applied to a full embedded system example later in this paper.

Peripheral Story Boards

The Peripheral Story boards are what give the Tower platform longevity not only as a professional demonstration and development board, but also as an educational platform. These boards allow for infinite features to tie embedded systems curriculum to various types of applications. The purpose of these boards is to consolidate common circuitry or embedded system interfaces that are controller agnostic. This way the controller of choice can be selected by the professor and the peripherals are unchanged and re-used. Table 2 lists the existing Peripheral Story boards in the Tower portfolio along with their feature sets.

Peripheral Story Board	Features
Serial Story	• Half/Full-duplex RS232 and RS485 • Control Area Network (CAN) • Low/Full-speed USB 2.0 host, device, and on-the-go (OTG) • 10/100 BASE-T Ethernet through MII or RMII interface • I2C to USB OTG device
Memory Story	• Secure Digital (SD) card interface for memory or SD input/output devices • Compact Flash card interface • Complex Programmable Logic Device (CPLD) • Magnetoresistive RAM (MRAM) • Serial Flash
Graphical LCD Story	• QVGA display • Resistive touch screen • Serial, Parallel, or RGB interface • Bi-directional Joystick • Audio Buzzer

Table 2. Existing Peripheral Story Boards and their Features

The Serial Story is shown in Figure 4 as an example of an existing peripheral Story board. This Story utilizes signals from the primary connector and only connects to GND on the secondary connector. In addition to the serial interface ICs and connectors, the board features many pin headers that are used to select options such as interrupt signal assignments, Ethernet boot strapping, USB mode selection, and RS232 or RS485 transceiver selection.

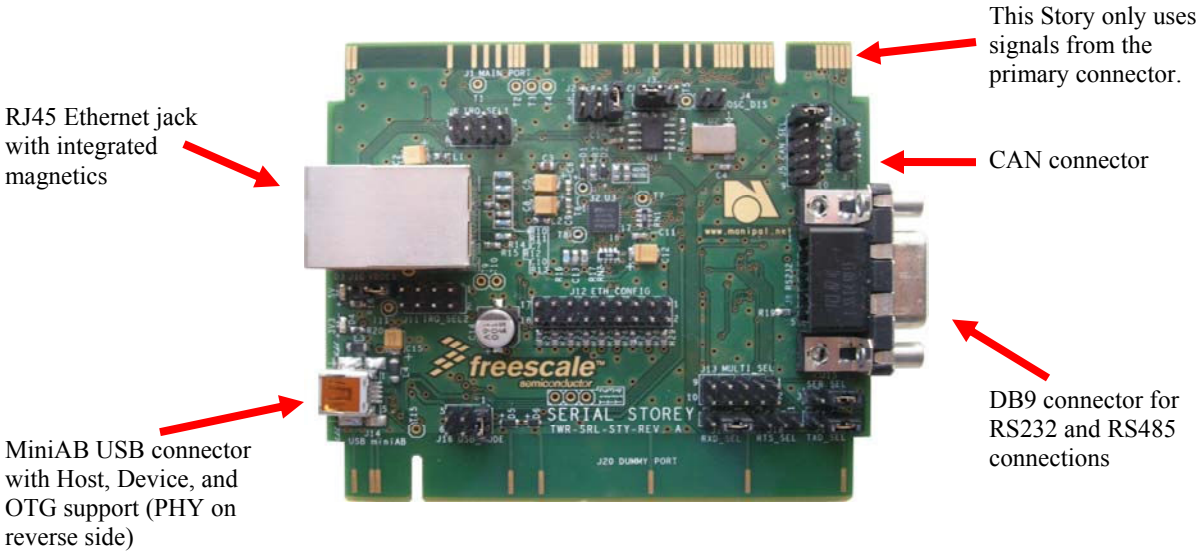


Figure 4. Serial Story

This Story alone opens up several possibilities for laboratory exercises. Several examples are provided in Table 3 below. Now the question is "how do you structure curriculum around the Peripheral Story boards?" Table 3 below lists a structure for example lectures and labs complimented by the features provided on the Serial Story peripheral boards.

Lesson No. & Estimated Duration	Lecture & Lab Title	Topics Covered

#9 (3 weeks)	USB Device Communications	• USB Fundamentals
		• USB Device circuit design and layout
		• Human Interface Device
		  ○ Accelerometer controlled mouse
		• Communications Device Class
		  ○ Serial communications over USB
		• Mass-Storage Class
		  ○ Emulate a USB mass-storage device
		  ○ Create a USB bootloader to virtualize the system flash and allow for firmware upgrades

#10 (3 weeks)	USB Host Communications	• USB host layout • MSC ○ Read/write a USB flash stick
#11 (4 weeks)	Ethernet	• Fundamentals of networking

		• Protocol stacks (e.g. uIP, lwIP, Niche-lite, RTCS) • Applications (Telnet, email, web server, etc.) o Internet controlled gadget (e.g. toggle LEDs and read sensor data remotely via web)
#12 (3 weeks)	CAN	• Industrial and automotive communications o Communicate across two or more tower systems using CAN communication protocol

Table 3. Example Lecture & Labs using Serial Story Board

To add to this Peripheral Story board portfolio, Freescale has a roadmap and concept for more story boards that can be used to develop higher level embedded systems education. See Table 4 for the peripheral boards that are planned at this time.

Peripheral Story Board	Description
802.11b WiFi Story	Low power, serial interface WiFi radio board
802.15.4 Story	Serial interface 802.15.4 radio board
Segment LCD Story	Outward facing board with segment LCD display
Audio Interface Story	AC97 or I2S external codec board
Motor Control Story	Circuits and connectors for driving motors
Capacitive Touch Story	General purpose capacitive touch development board
Battery Pack Story	Rechargeable Li-ion battery pack
Photovoltaic Energy-Harvesting Story	Solar panel to harvest ambient sun light. Energy used to run Tower and charge a reserve.
Telephony Story	Telephony interface (FXO, FXS) for embedded voice applications
IrDA Story	Simple UART interface to IrDA
Prototyping Story	Protoboard, switches, pin headers, etc. needed for quick prototyping
FPGA Story	High-end FPGA board with connection to all signals in Tower. USB interface for external control and programming. Will enable hardware prototyping and data acquisition capabilities.

Table 4. Planned Peripheral Story Boards

Custom Story Boards

Peripheral Story boards give curriculum scalability. The open source hardware model of the Tower system (schematics and layouts are freely available for Tower boards produced by

Freescale Semiconductor), and the published mechanical and electrical standard allow for quick creation of custom Story boards. Freely available CAD packages such as PCB123[3], ExpressPCB[4], and EAGLE Light Edition[5] provide the tools necessary for cost effective PCB board design. PCB board fabrication can also be realized in a cost-effective manner using services such as ExpressPCB[4] and PCBexpress[6]. These services and similar can be used to add production of real, modern hardware to a curriculum. The ability to add student-created custom hardware to an existing modular platform provides many options for a satisfying undergraduate final project or graduate level lab or research project.

Software Development Tools

There are several options when it comes to choosing software development tools to support development on a Tower platform. The basic need is a compiler and debugger or an integrated development environment (IDE). CodeWarrior® Development Studio[7] is a good option for an IDE. It supports most of the Freescale MCU and MPU architectures and offers a Special Edition version that has some restrictions on object code size but is complimentary. Having free professional tools available to outfit all labs or for students to install on their own computer to take their development anywhere is a key benefit. A Professional Edition is also available that removes all restrictions. In addition to providing traditional IDE tools, CodeWarrior Development Studio also features an advanced tool called Processor Expert[8] that offers advanced features for device initialization, device drivers, code generation and basic elements of embedded system applications. A tool like Processor Expert could be used in an introductory embedded programming course. By configuring high level variables for a given controller peripheral the graphical tool will automatically generate the necessary code to initialize that peripheral. This can be used as a learning tool to match the code to peripheral functionality.

Another compiler and debugger option to consider is the GNU toolchain maintained by CodeSourcery[9]. Regular releases are available for a wide range of computer architectures and an upgraded version is available that offers an Eclipse-based IDE and enhanced hardware debug support.

Freescale MQX™ [10] software is a logical choice for a real-time operating system (RTOS) that supports Freescale MCU and MPU Story boards. The MQX software solution is a full-featured, professionally developed, production quality software suite that is complimentary with recent Freescale MCU and MPUs[11].

Another viable OS option is FreeRTOS[12]—a portable, open-source, mini real time kernel with a strong user community and support for a wide variety of architectures.

These and other development tools add to a rich set of tools that greatly benefit engineers in the industry. Experience with these tools in an academic setting will better prepare engineering students for careers in industry.

Design Example: An HVAC Controller

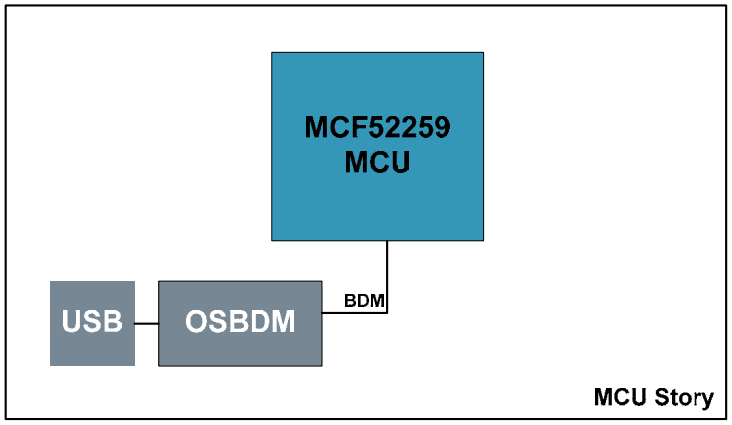
The Tower's rich set of MCU/MPU and peripheral boards along with the ability to rapidly prototype custom expansion boards provide an excellent hardware platform for embedded system course work. This section provides an example of how this platform might be used to create a series of courses or labs that build progressively toward a complete application: a digital, web-enabled heating, ventilation, and air-conditioning (HVAC) controller. This example makes use of an MCU Story featuring a V2 ColdFire MCU, the MCF52259[13], and a Serial Story.

Section 1: Using the Tools / Software Programming Refresher

Required Hardware:

- TWR-MCF52259 – MCU Story

In this section students are able to get familiar with the hardware and software tools as well as refresh their programming skills. It covers getting started activities such as how to write, compile and debug simple programs using the toolchain and student-owned MCU Story (TWR-MCF5225X).



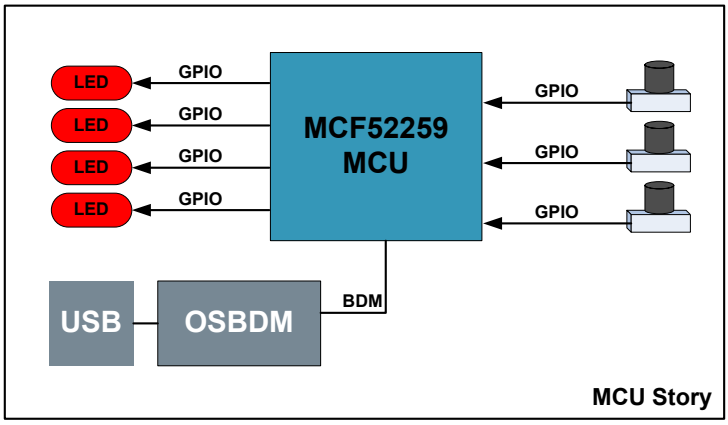
Section 2: Embedded System-on-a-chip (SoC) Fundamentals, Part 1

Required Hardware:

- TWR-MCF52259 – MCU Story

This section explores the basic features of the MCU such as general purpose input/output (GPIO) pins, interrupts, and timers. The exercises include:

1. Understanding pin multiplexing; selecting the GPIO and alternate functions of the available pins on the MCF52259
2. Program control over the output state of GPIO; toggling an LED using a GPIO signal
3. Read the state of the GPIO pins connected to the pushbutton switches.
4. Configuring a timer to timeout periodically; poll the timer status and toggle an LED on a periodic basis
5. Using the timer interrupt capability; toggle LED inside interrupt routine



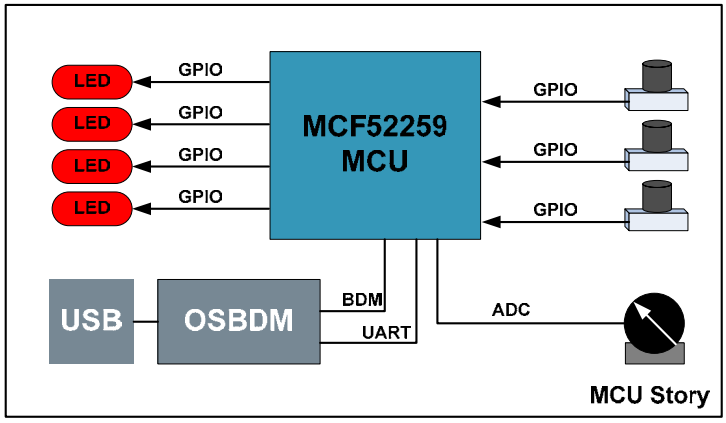
Section 3: Embedded System-on-a-chip (SoC) Fundamentals, Part 2

Required Hardware:

- TWR-MCF52259 – MCU Story
- TWR-ELEV – 2 Elevator Boards
- TWR-SER – Peripheral Serial Story

Building on the last section, students continue to explore the on-chip features of the MCU using just the MCU Story.

1. Configure an analog-to-digital converter (ADC) channel to read values from the potentiometer
2. Program additional ADC channels to read up to three axes of acceleration data from the accelerometer.
3. Initialize the UART port that is connected to the OSBDM circuit. Use a terminal application on a PC to send and receive characters over the serial port.



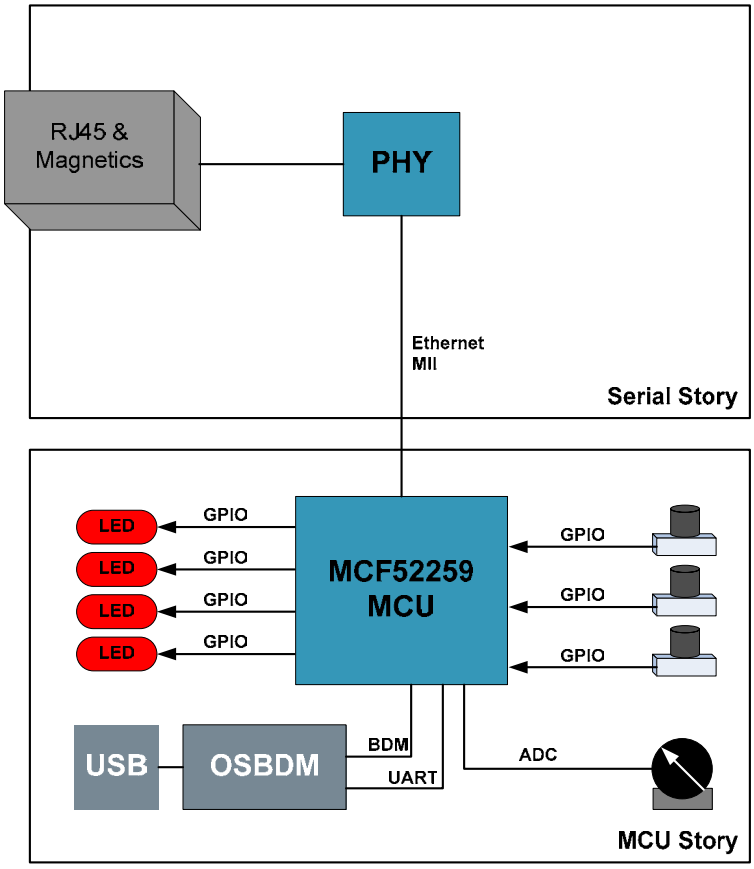
Section 4: Ethernet and Networking

Required Hardware:

- TWR-MCF52259 – MCU Story
- TWR-ELEV – 2 Elevator Boards
- TWR-SER – Peripheral Serial Story

This section requires the use of the Tower Elevators and the Serial Story board. The MCF52259 features a Fast Ethernet Controller that is connected to the Ethernet physical layer IC (PHY) on the Serial Story using an MII interface. Prerequisites for this section might include a networking course, or the fundamentals could be taught here.

1. Fundamentals of networking
2. Understanding network protocol layers and software stacks; explore the capabilities and use of one of the many stacks freely available for this platform such as lwIP[14], NicheLite[15], or RTCS[16] (part of the MQX suite).
3. Configure an embedded web server and serve pages that show the status of the LEDs and the potentiometer.



Section 5: Emulating the HVAC Controller

Required Hardware:

- TWR-MCF52259 – MCU Story
- TWR-ELEV – 2 Elevator Boards
- TWR-SER – Peripheral Serial Story

The end application starts to come together in this section. The hardware provided on the MCF52259 Tower is enough to emulate an HVAC controller. The push-button switches can be used to set the desired temperature of the system. The potentiometer can be used to

emulate the data readings from a temperature sensor. The application code must determine when to turn on/off the heating and air-conditioning unit and control the ventilation fan. A typically digital HVAC controller would use digital outputs connected to relays that are used to start and stop the equipment. Here students use the GPIO signals and LEDs to emulate the relay control. The serial interface can be used as an alternative user input method, and since this is a web-enabled HVAC controller, the students must implement a web page that provides real-time status of the system and allows remote control.

Section 6: Rapid Hardware Prototyping

Required Hardware:

- TWR-MCF52259 – MCU Story
- TWR-ELEV – 2 Elevator Boards
- TWR-SER – Peripheral Serial Story

Many labs would end with the previous section. However, given the ease of adding custom boards to the Tower platform, students can go a step further and create a Story board that realizes the hardware required to control a real HVAC system. In this section students use existing templates to design and layout a Story board that implements the additional hardware required to realize an operational HVAC controller. With the minimal addition of some MOSFETs, relays, and a temperature sensor the design would be functional.

This simple design example demonstrates how the Tower platform aides in an embedded systems design track. It could be enhanced to include the use of the graphical LCD Story with touch screen to implement a human machine interface (HMI) and the USB port for data logging or firmware updates. Furthermore, an RTOS such as MQX[16] RTOS could be utilized to add operating system experience as well.

Conclusion

Tower is a modular, cost effective demonstration and development hardware platform. The modular hardware provides building blocks that scale from entry-level to advanced embedded systems coursework. The platform is standardized with open source hardware that enables reuse of existing circuits and rapid prototyping of custom hardware. The interchangeable expansion boards enable and promote reuse of hardware and software across multiple MCU and MPU architectures giving professors flexibility in course development.

This powerful professional development tool lends itself well to an academic setting. The low cost of entry allows for student-owned hardware that will enhance and accelerate the learning experience. Embedded systems courses can leverage the modular nature of the platform to keep courseware state-of-the-art while maintaining continuity from entry-level to advanced courses.

Acknowledgements

The Tower system was a joint development project between Freescale Semiconductor and Manipal Dot Net (MDN)[17]. The success of the Tower platform is due in large part to the creative and professional skills of MDN.

References

1. Freescale Semiconductor, Inc., http://www.freescale.com
2. Jack, H. and Barakat, N "A Student Owned Microcontroller Board", ASEE Annual Conference, 2006
3. PCB123, http://www.pcb123.com/
4. ExpressPCB, http://www.expresspcb.com/
5. EAGLE Light Edition, http://www.cadsoftusa.com/freeware.htm
6. PCBexpress – A Division of Sunstone Circuits, http://www.pcbexpress.com/
7. CodeWarrior Development Studio, http://www.freescale.com/codewarrior
8. Processor Expert, http://www.freescale.com/webapp/sps/site/prod_summary.jsp?code=PROCESSOR-EXPERT
9. CodeSourcery, http://www.codesourcery.com/
10. Freescale MQX, http://www.freescale.com/mqx
11. Johnson, R. C., "Freescale offers free software license with microcontroller purchase", EE Times Online, Jan 22, 2009
12. The FreeRTOS.org Project, http://www.freertos.org/
13. MCF5225x, http://www.freescale.com/mcf5225x
14. lwIP – A Lightweight TCP/IP stack, http://savannah.nongnu.org/projects/lwip/
15. NicheLite TCP/IP – Compact Networking Protocols, http://www.iniche.com/nichelite.php
16. Freescale MQX Real-Time TCP/IP Communications Suite (RTCS), http://www.freescale.com/mqx
17. Manipal Dot Net, http://www.manipal.net/