SELECTING MICROCONTROLLERS AND DEVELOPMENT TOOLS FOR UNDERGRADUATE ENGINEERING CAPSTONE PROJECTS

Jeevan F. D'Souza[1], Andrew D. Reed[1], and C. Kelly Adams[2]
College of Engineering and Information Sciences
[1]DeVry College of New York, NY
[2] DeVry University Online, Chicago, IL

Abstract

Most undergraduate engineering programs require completion of a capstone design project. For many projects in disciplines such as electrical engineering and mechanical engineering, the chief processing component of these projects is a microcontroller unit (MCU). There are hundreds of MCUs and development tools (DT) available in the market today. Identifying a suitable MCU and DT at the detail necessary for project integration could be a challenging, tedious, and time consuming task for those involved in the capstone process. Selecting an inappropriate MCU and DT could lead to incomplete, less challenging, or poorly implemented projects. This paper strives to alleviate the problem of MCU/DT selection by researching capstone project designs, constraints, and requirements across a range of undergraduate engineering programs. It then outlines important MCU and DT selection criteria for undergraduate capstone projects. Finally, several MCUs and DTs that are both popular and widely available in today's market and suitable for undergraduate engineering capstone projects are recommended.

Introduction

Undergraduate engineering programs strive to provide a solid foundation of both theoretical and practical knowledge in order to prepare students to successfully join the workforce after graduation. Course lectures are the main vehicle for imparting theoretical knowledge to students. Students gain practical knowledge and design competency by working on hands-on lab experiments and projects throughout their course of study [1, 2]. Most engineering program curricula require completion of a capstone design project. As part of the capstone process, students use the entire scope of their past classroom and lab experiences to design and implement a real world project [3]. The capstone project is usually conducted as part of a multi-course sequence spanning one to three semesters and is typically completed during the senior year [4].

Capstone projects in electrical engineering, mechanical engineering and related disciplines generally require students to build systems that have mechanical, electrical and control components. In many cases, capstone projects use a microcontroller unit (MCU) mounted on a development board (DB) as the central processing/control unit. The DB is capable of sensing and controlling various devices with the help of a program written using an integrated development environment (IDE) software. In this paper, the DB/IDE integration is collectively called a development tool (DT).

There are hundreds of DTs and MCUs available in the market that can be chosen to satisfy the requirements of a capstone project [5]. Selecting the ideal MCU and DT for a particular project could be a time-consuming and tedious task for an undergraduate student or faculty mentor [6]. This is exacerbated by the fact that a student is usually only familiar with the MCU, DB and IDE they have used during their undergraduate classroom and lab experience. In addition, if unsuitable MCUs and DTs are selected, the project could be physically, technically and monetarily inefficient.

This paper addresses the MCU/DT selection dilemma by researching a wide array of popular MCUs and DTs available in the market today

and recommends the most popular, cost-effective, easy-to-use, robust, and highly-functional tools that are best suited for inclusion in undergraduate electrical and mechanical capstone projects. Having a smaller pool of ideal MCUs to choose from can help students move past the selection process and focus their valuable time and effort in designing and building a complex, quality project. Using popular MCUs that are widely available and commonly used in the industry builds real world engineering design skills as students prepare for engineering practice.

The Undergraduate Engineering Capstone Project

Most undergraduate engineering programs require students to design and implement a capstone project near the end of their course of study [7]. This project is usually conducted as part of a multi-course sequence completed during their senior year [8]. Students are then able to use the entire scope of their classroom and lab experience in the design and implementation of their project. Students typically spend between six months to a year working on their capstone project. During this time, they are also completing other courses needed to meet graduation requirements [7]. Students typically use personal funds to bankroll their project, although there have been cases where grants have been used to fund large projects [8]. Time and money become two central factors considered in capstone project development and implementation.

Examples of current electrical and mechanical capstone projects include intelligent traffic lights, refreshable braille readers, sensor networks, vending machines, remote controls, unmanned airplanes [9], and disparate robots. The underlying commonality between these systems is that the designs share a general three-block architecture as outlined in Figure 1.

Figure 1. A general framework of electromechanical project design.

Most capstone projects are designed around this general three-block architecture. Sensors are used in the first block of the architecture to sense physical quantities like sound, light, distance, humidity, etc. The output quantities from the sensors are sent to the second block, the electronic processor/controller, where they are processed and stored. Finally, the desired output of the processing block is passed to the mechanical or electronic devices of the output/display block. Common mechanical or electronic devices include solenoids, actuators, relays, motors, computers, light emitting diodes (LED) and liquid crystal displays (LCD).

Sensing, processing, storage and control can be implemented in many ways. Today the most common device used to integrate these functions is the MCU. MCUs are small computers that integrate a central processing unit (CPU) core, different banks and types of memory, and other programmable input/output peripherals on a single chip. Common MCU peripherals include serial communication interfaces (SCI), timers and counters, wave generators, clock generators, interrupts, analog to digital converters (ADC), and digital to analog converters (DAC). MCUs are very desirable for robotics and embedded design projects because they include an entire system on a single chip. MCUs integrate all of the functional blocks onto a single chip.

An MCU is typically mounted on a DB. Sensors and other output devices used in a project can easily be interfaced with the MCU via the DB pin connectors. The integrated MCU/DB combination makes it easy to interface the MCU with a personal computer (PC) via an SCI. The MCU typically has a boot loader program loaded on to its EEPROM

allowing easy uploading of user-written programs from a PC to the MCU via an IDE.

Figure 2 shows an Atmega328P MCU mounted on an Arduino Uno DB. The board has pin connectors that allow the MCU to be connected to input and output devices. The board has a reset pushbutton and an USB port in order to allow easy interfacing to a PC.

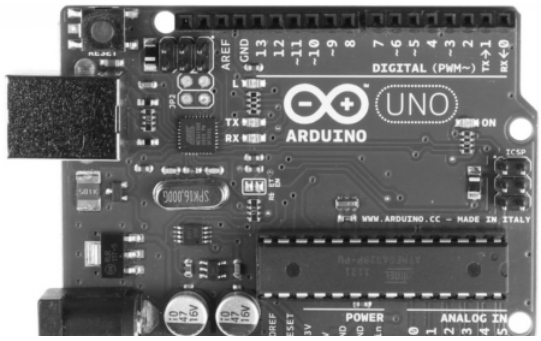

Figure 2. An MCU mounted on a DB [26].

Figure 3 shows a typical screenshot of the Arduino IDE with a sample program written in the C programming language. The program interfaces a button and LED to the Arduino board. If the button is not pressed, the LED turns on, if it is pressed, the LED blinks as many times as the user has pressed the button so far. Some IDEs are capable of program development in several languages like BASIC, C++ and assembly.

In addition to the DB and MCU, the project might often need mechanical housing and auxiliary parts to perform certain functions. Figure 4 shows a MCU and DB embedded into a completed project designed and implemented by students at DeVry University. This system is capable of reading radio frequency identification (RFID) cards, determining the value of money left on the card, and then turning on cell phone charging devices depending on the value left on the card.

This section presented examples of typical undergraduate capstone project design requirements in order to shed some light on the monetary and time constraints students might face during project design and implementation.

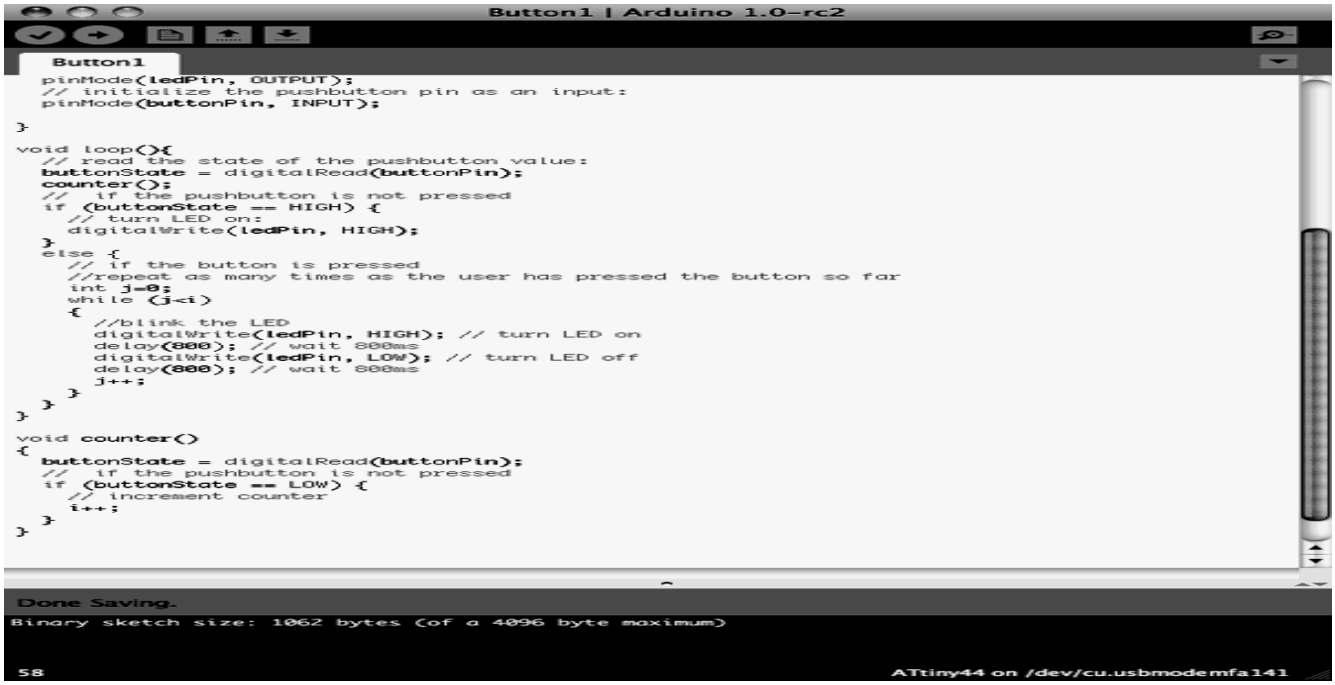

Figure 3. A typical MCU IDE [31].

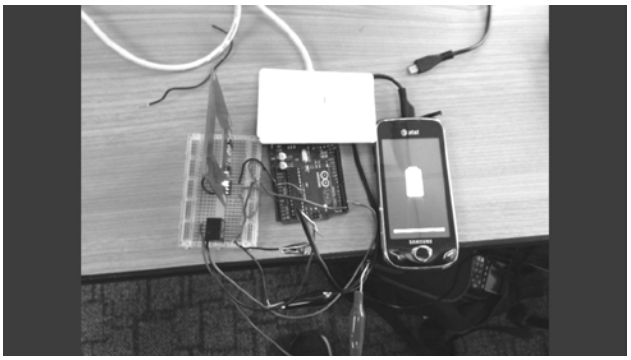

Figure 4. An undergraduate capstone project using an MCU mounted on a DB interfaced with other devices.

Important Selection Factors

Attempts have been made to select ideal microcontrollers based on certain criteria [5, 14, 15], but these articles are not specifically aimed at undergraduate capstone projects. This section lists important considerations in MCU, DB, and IDE selection for an undergraduate capstone project. A brief explanation of each factor is paired with suggestions designed to help guide students and their mentors through the MCU/DT selection process.

MCU Selection Factors

1) Pin Count

More pins suggest more ports available for interfacing devices to the MCU. Whereas some pins have specific functionality, most pins are general purpose input/output (GPIO) pins. The MCU chosen for the design must have enough pins to connect all IO devices used in the project, but minimize the number of unused pins.

2) CPU Speed

The speed of a MCU is measured in megahertz (MHz) and million instructions per second (MIPS). Typically, MCUs with larger bus widths tend to have higher speeds. The amount of processing required for the project determines the necessary MCU speed. The amount of processing needed should match the type of processor selected. For example, selecting a very powerful processor for a simple processing need would be a waste of power and resources. Powerful MCUs have more complex architectures. This could increase the learning curve for a student unfamiliar with the device chosen and increase the amount of time needed for project design and implementation. Becoming familiar with a new MCU at a level necessary for project integration is a time-consuming task for anyone. For most undergraduate capstone projects, an 8-bit MCU running at 8-16 MHz proves sufficient speed for the processing needs of a typical project.

3) Power Consumption

Power consumption depends mainly on the current being drawn by both the MCU and any external devices connected to it. More powerful and complex MCUs draw more current. In addition, external devices such as motors and sensors could require more current to operate. It is important to take the current capacity per pin of the MCU into consideration during the selection process. The current requirements of the devices used in the project should weigh heavily in the selection process. The MCU selected should have appropriate current ratings to cover the current drawn by both the MCU and the external devices.

4) Sleep Mode

Some MCUs have low power sleep modes to help conserve power. The MCU awakens during the occurrence of an event, but spends the rest of the time “sleeping” [11]. If a project is using batteries and needs to remain functional for extended periods of time, this could be a very important factor in the MCU selection process.

5) Instruction Set

The larger the data bus width, the more complex the instruction set of the MCU [10]. For example, a 32 –bit MCU has more complex instructions compared to an 8-bit MCU. For a simple project requiring little processing, using a complex 32-bit MCU would be a waste of resources. Selecting a 32-bit MCU, with its inherently more complex design, would require more time for a student to become familiar with

its programming and operation at the detail necessary for successful project integration. In most cases an 8-bit or 16-bit MCU should meet the processing requirement of the project.

6) Program memory

MCU program memory is generally present in the form of non-volatile FLASH or EEPROM. The MCU boot loader and user programs are typically stored in the program memory. Size of the user-written code determines the ideal size of the program memory needed for a specific project. For example, projects using large programs should use MCUs with larger program memory. For most capstone projects, using an MCU with 8 – 16 KB of program memory would accommodate the program code.

7) Data memory

Data memory is generally used to store data acquired from the sensors. MCU data memory, generally present in the form of volatile RAM, can also be present in the form of FLASH or EEPROM**.** Size of the data stored in the capstone project determines the ideal size of the data memory necessary for a specific project. For most general projects 8-16 KB of data memory should be sufficient to store logged data.

8) Memory Expansion

Memory expansion capability allows the MCU to interface with additional external memory of much larger sizes. This allows for the addition of memory capacity beyond the built in capacity of a particular MCU. In projects where storing large amounts of data is critical, this could be an important selection criteria.

9) Analog-to-Digital Converter (ADC)

An ADC takes analog physical quantities, samples them, and converts them to digital values that can be processed by the program stored in the MCU program memory [12]. Since most capstone projects use analog input devices to sense physical quantities, most projects will require using an MCU with an ADC.

10) ***Digital-to-Analog Converter (DAC)***

Many output devices typically used in capstone projects need analog input signals to function. MCUs are digital devices. Therefore, digital values need to be converted into analog form before they are sent to any output device that needs an input signal in analog form. The DAC enables the digital to analog conversion to take place. In projects where output devices need an analog signal to function, this is an essential feature of the MCU selected.

11) Interrupts

Interrupts allow the MCU CPU to temporarily stop its regular processing cycle and focus on a real-time event. Most MCUs contain both hardware and software interrupts. Interrupts can also be used to wake an MCU from sleep mode. If real-time events and power saving are an important need of the project, a MCU with interrupts must be chosen.

12) Pulse Width Modulator (PWM)

Pulses of different frequencies and duty cycles can be generated by using a PWM. These pulses can then be used to independently drive motors and other loads without using CPU processing power. Since the PWM is an independent unit in the MCU, choosing a MCU with a PWM is important to any capstone project that needs to control the speed or power of motors or other loads.

13) Timers

Timers are used to count down from a specific given value to zero. Flags or other events are generally used to trigger a timer. When a timer finishes counting, flags are modified and interrupts are sometimes generated depending on the MCU design. Timers are ideal for counting pulses, measuring frequencies and generating waveforms. Since the timer is an independent unit in the MCU, choosing a MCU with a built in timer is an important option to any capstone project that needs this type of function.

14) Serial Communication

Serial communication allows communication between digital devices over a serial link. MCUs generally use serial communications to download programs from a PC to a MCU. For this reason, most MCUs have serial communication capabilities by default. Common MCU serial communication protocols are RS-232 serial communication interface (SCI), serial peripheral interface (SPI), inter-integrated circuit (I2C) and Universal Serial Bus (USB). Projects that require establishing connections to several devices should consider using a MCU with multiple serial communications options.

15) Vehicle Bus

The Controller area network (CAN) standard was designed to enhance communication between a MCU and another device located inside a vehicle. For example, the target device could be located inside an automobile or airplane [13]. Projects that include creating a specific vehicle that needs to communicate with the MCU and other devices should consider selecting a MCU with a CAN.

16) Real Time Clock (RTC)

An RTC is a clock that keeps time independent of the MCU CPU. An RTC typically works in conjunction with an interrupt and is designed to interrupt the CPU after a specific amount of time. Projects that need to keep track of time independent of the CPU should consider selecting a MCU with a RTC.

DB Selection Factors

1) MCU Support

The MCU is the most important part of the DB. The MCU features will strongly impact the DB selection process. The primary consideration in selecting a DB for any project is then the specific set of MCU features needed for the applications at hand. The previous section outlines important factors to consider in selecting a MCU. In addition, it could prove beneficial if the DB selected supports a family or series of MCUs. Having the option to replace the MCU with another in the same series gives added flexibility to the project developer.

2) Cost

Generally, costs associated with the IDE and MCU are built into the cost of the DB that is purchased. Costs of a DB can range from $20 to $2000 depending on which MCU, IDE and DB feature sets are included [16]-[24]. Often, the IDE that ships with a DB is a limited version IDE. An additional cost would be incurred to access the full-featured IDE. Capstone projects are often limited enough in scale to be satisfied with a low end DB coupled with a simple 8-bit MCU and the default, limited-functionality IDE.

3) Size

Size can be a very important factor when selecting a DB. For example, a remote control project would require a very small DB so that the entire device could be handheld. For other projects, a larger DB might be suitable. DB size depends on the number of embedded external peripherals. DBs that include many peripherals tend to be larger. For projects that require small size and maximum power saving, DBs with minimum peripherals are recommended.

4) Power Supply

MCUs typically operate using a 3-6 volt supply voltage [20]-[25]. DBs tend to operate with a wider range of acceptable supply voltages, typically in the 5 – 15V range [16]-[19]. As a result, powering the DB with different batteries and power sources is easier. Many DBs also have the ability to supply power to external peripherals. DBs offering several different supply voltages can power a wider range of external IO devices without the need for multiple power sources. This is an important feature for a project powering multiple external peripheral devices.

5) Power Regulation

Devices can be damaged inadvertently if too much current is drawn while the device is in use. Most DBs come with a current and voltage regulating circuit designed to prevent current overflow damage to the MCU and attached

devices. It is recommended that a DB selected for any capstone project come with a regulation circuit.

6) Embedded IO Devices

Projects often make use of displays, motor drivers, switches, relays and other external peripheral devices. In some cases, a project can be implemented faster if these key devices are embedded on the DB. If embedded devices are not used for a project, their inclusion is unnecessary and could increase the size of the DB. In projects where size is an important factor, efficiency dictates using a DB with minimum embedded IO devices and interfacing only the specific devices required.

7) PC Interface

The two common interfaces between a PC and an MCU are USB and RS-232 [20]-[25]. If both devices use the same protocol and physical interface, interfacing is easy. In many cases, the MCU uses the RS-232 communication protocol and the PC uses the USB communication protocol with a USB physical interface. In these cases, a USB-to-RS-232 driver will be needed for the PC. It is important to make sure the PC has the appropriate port and driver for interfacing purposes. An adapter and driver may have to be purchased in some cases. Using a plug-n-play (PnP) board has advantages in saving time and integration time. PnP boards and adapters are automatically recognized by the operating system (OS) and the required drivers are automatically installed.

8) Program Upload

The DB should enable fast and easy user-program uploads onto the MCU memory. It should also have the capability of allowing the program to be uploaded to RAM, EEPROM, or Flash memory. In addition, the DB should have the ability to upload a new boot loader program. A boot loader is loaded onto the MCU program memory and allows the user to upload and run user-written programs. A program using custom code will benefit from a more user friendly and easy to use development environment.

IDE Selection Factors

1) DB Compatibility

The most important factor in selecting an IDE is DB compatibility. The IDE must be compatible with both the DB and MCU being used in the project. In many cases, IDEs are developed by the DB manufacturer, but alternate IDEs are available for internet download. For capstone projects, students are constrained by time and experience. In this case it is recommended that a manufacturer-developed IDE is used for compatibility reasons. In addition, if a DB is capable of using a family of MCUs, the IDE must be capable of supporting all of the MCUs in the series.

2) Time

A large portion of a student's time during their capstone project is spent learning the IDE and using the IDE to develop the necessary program. Project development time depends heavily on the IDE capabilities which in turn affect the program development and uploading process. It takes time for a student to become familiar enough with an IDE to complete a project. Since capstone projects are already time constrained, it is important to select an IDE that can be mastered in a relatively short timeframe.

3) Libraries

Writing a program at the bit-level to control each MCU peripheral would be tedious. Project development time can be reduced by having access to pre-written libraries [14]. Most IDEs available today come with pre-written libraries and it is recommended that students take advantage of them in order to save time and increase efficiency.

4) Online Support

Students might run into situations where they need help in IDE programming, especially if this is a new environment for them. Some available IDEs have good online resources often in the form of both text and videos. Selecting an IDE with extensive internet resources could significantly reduce project development time.

5) Ease of Use

The IDE program will have to be edited and uploaded several times onto the MCU throughout the course of the capstone project. Choosing an IDE that is fast and easy to use is paramount. Having a fast and easy process for IDE use can significantly save project development time.

6) Languages

Assembly, C and C++ are the most common languages used for MCU and embedded system programming [19], [21]-[24], [27]-[30]. Any IDE selected should have the capacity to compile and assemble at least one of these three languages.

7) Simulation

Having the ability to simulate the MCU environment on a PC to run and test code independent of the MCU can help speed up project development time and should be considered.

8) Debugging

Debugging is the process of finding bugs in a program**.** The IDE should be able to display all registers, memory and port contents for efficient debugging. The IDE should also have the capability to use breakpoints and single-step programs in multiple languages and display and edit these values in real-time. Project Development time can be reduced by using an IDE with advanced debugging capabilities.

Results

This section describes the research methodology used to select MCUs and DTs based on the previous section. It also enlists some ideal candidates in tabular form.

Research Methodology

This subsection describes the research methodology used for selection of MCUs and DTs to be use in undergraduate capstone projects. In addition to their project criteria, students working on a capstone project should also consider the currently available microcontroller technology popular in industry. Therefore, the most important criteria used in the final selection of MCU and DT for this paper was industry popularity. The second most important criteria used in the selection of MCU and DT for this paper was overall cost and size. Finally, IDE development/learning time and other factors mentioned in the previous section were considered.

Epstein [15] lists the top ten MCU manufacturers by market share from 2012. These manufacturers are Renesas Electronics, Freescale Semiconductor, Atmel, Microchip Technology, Infineon Technologies, Texas Instruments (TI), Fujitsu, NXP Semiconductors, STMicroelectronics (STM), and Samsung. A thorough internet search was performed to find DBs that use MCUs manufactured by these popular corporations. First, DBs were shortlisted based on their low cost and small size. Next, DBs were shortlisted based on IDE development/learning time and other factors mentioned in the previous section.

Findings

Primarily based on industry popularity, cost and size, the DBs listed in Table I have been selected as ideal candidates for undergraduate capstone projects. Students can then use their specific project criteria to select an ideal one from the list.

The specifications of the MCUs supported by the DBs mentioned in Table I are mentioned in Table III. For the MSP430 Launchpad DB, the MSP430G2553 MCU was chosen from the series of MCUs that are compatible with the DB.

Based on compatibility with the DBs mentioned in Table I, IDE development/learning time and other factors mentioned in the previous section, the IDEs listed in Table II have been selected as ideal candidates for undergraduate capstone projects. Students can then use their

specific project criteria to select an ideal one from the list.

Discussion

The MCUs and DTs identified in Tables I - III are suggestions to help students and faculty mentors jumpstart the selection process. Students can use their specific project criteria to select a possible candidate from the list. Most student capstone project requirements can be satisfied by using one of the DBs, MCUs, and IDEs mentioned in Tables I – III, although all MCUs and DTs in the list may not be suitable for all projects. Microcontroller technology changes over time. Tables I-III are designed to be guidelines that refer back to the selection criteria guidelines in the previous section.

As each DB and MCU has its own pros and cons, this is designed as a guide. Students can use this guide to help narrow the various available choices to find a MCU/DT that meet their own particular project requirements. MCUs and DTs that were investigated but did not make the top selection list might still be good candidates for some capstone projects. They are listed in Table IV.

Renesas Electronics, Infineon Technologies, Fujitsu and Samsung were identified as top MCU manufacturers. Their MCUs were not selected as ideal candidates for undergraduate capstone projects mainly due to their high cost, complex microarchitecture, application specific processors, and large DB size. [21-24].

Table I. DB Specifications.

Manufacturer →	**Uno** Arduino	**StartUSB** Mikro-Elektronica	**Launchpad** TI	**Thundebird12** EVBplus	**Mbed** Mbed
MCUs supported	Atmel ATmega328	Microchip PIC18F2550	Any TI MSP430 MCU	Freescale MC9S12DG256	NXP LCP11U24
Power Supply	7-12V (USB or battery)	5V (USB or battery)	5V (USB only)	5V (USB or battery)	5-9V (USB or battery)
Power Regulation	Max per pin 40mA	Max per pin 25 mA	Max per pin 12mA	Max per pin 25 mA	Max per pin 20mA
Size	2.7” x 2.1”	1.5” x 1”	2.6” x 2.0”	3.05” x 0.88”	2.1” x 1”
Embedded devices	1 LED	None	2 Buttons, 2 LEDs	4 LEDs	4 LEDs
PC interface	USB Virtual Serial Port	USB	USB Virtual Serial Port	USB via board	USB
Program upload	To Flash Memory via USB and boot loader	Boot loader Program	To Flash Memory via USB	To Flash Memory via USB and boot loader	Drag and Drop like a Flash drive
Cost	$29.96	$19.00	$4.30	$78.00	$45.00
Source	[26]	[17]	[18]	[19]	[20]

Table II. IDE Specifications.

	Arduino	MicroC PRO for PIC	Code Composer Studio	Code warrior	Mbed Online Compiler
Developer	Arduino	Mikro-Elektronika	TI	Freescale	mbed
Time	Small	Large	Large	Large	Medium
Ease of use	Very Easy	Medium	Medium	Medium	Easy (Online)
Languages	C/C++	Basic/Pascal/C	Assembly/C	Assembly/ C/C++	Assembly/ C/C++
Simulation	No	Yes	No	Yes	No
Debugging	No	Yes	Yes	Yes	No
Library Support	Large	Large	Large	Small	Medium
Online support	Excellent	Good	Excellent	Good	Very Good
Free Version Limitations	None	Max 2K Words	Max16KB code size	None	None
Cost of full version	None	$250.00	$795.00	None	None
Source	[27]	[28]	[29]	[36]	[30]

Table III: MCU Specifications.

	ATmega 328	PIC 18F2550	MSP 430G2553	MC 9S12DG256	LPC 11U24
Manufacturer	Atmel	Microchip	TI	Freescale	NXP
Pin count	32	28	20	80	64
CPU speed	20 MHz	48 MHz	16 MHz	50 MHz	50 MHz
Power consumption	1.8V to 5.5V Active Mode: 0.2mA @ 1.8V/1MHz	2 to 5.5V Active Mode: 15 mA @ 2.0V/32kHz	1.8V to 3.6V Active Mode: 0.23mA @ 2.2V/1MHz	2.5 to 5V Active Mode: 65mA @ 5Vea	1.8V to 3.6V Active Mode: 2mA @ 2.2V/12Mhz
Sleep modes	6	3	5	3	4
Instruction set	8-bit	8-bit	16-bit	16-bit	32-bit
Program memory	32Kb	32 Kb	16Kb	256Kb	32Kb
Data memory	2Kb	2Kb	512 bytes	12Kb	8Kb
Memory expansion	No	No	No	Yes	No
ADC	10-bit 8 channels	10-bit 10 channels	10-bit 8 channels	10-bit 16 channels	10-bit 8 channels
DAC	No	No	No	No	No
PWM	6 channels	2 channels	2 channels	8 channel	6 channels
Timers	3 8-bit	4 8-bit	2 16-bit	8 16-bit	2 16-bit 2 32-bit
Interrupts	24	19	19	50	24
Serial Communication	UART SPI(2) I^2C	USB Device UART SPI I^2C	UART(2) SPI(2) I^2C IrDA	UART(2) SPI(3) I^2C	UART SPI(2) I^2C USB Device
Vehicle bus	None	None	None	CAN 2.0(3)	None
RTC	Yes (32kHz)	External Support	External Support	Yes	Yes
Source	[35]	[32]	[25]	[33]	[34]

Table IV. Other MCUs and DTs Suitable for Capstone Projects.

DB	MCU	IDE	Source
Arduino Due	Atmel SAM3X8E	Arduino	[37]
Leaflabs Maple	STM32F103RB	Maple IDE	[38]
Arduino Mega	ATmega1280	Arduino	[16]
STM Discovery	STM32F100	IAR	[39]
Parallax Basic Stamp 2	Microchip PIC16C57c	PBasic	[40]
Parallax Propeller QS	Parallax P8X32A	Propeller tool	[41]

Conclusion and Future Work

The goal of this paper was to assist undergraduate electrical engineering students, mechanical engineering students, and faculty mentors in selecting MCUs and DTs suitable for designing and implementing undergraduate capstone projects effectively. Typical undergraduate engineering capstone project requirements were analyzed. Important factors in selecting MCUs and DTs were then identified, described and cataloged. Available MCUs and DTs in the market were investigated based on industry popularity, cost, time constraints on the learning curve, and technical features. Finally, five MCUs, DBs and IDEs suitable for use in undergraduate capstone projects were suggested and described.

Modern MCUs are complex, with industry technology advancing at a rapid pace. New MCUs and DTs are being released every year. The MCUs and DTs suggested as candidates for use in undergraduate projects were selected based on today's MCU technology and the standard complexity of current level undergraduate engineering projects. Whereas MCU technology can be expected to change over time, the important selection factors described can be used as a general guide to help students best prioritize their specific project needs compared to what MCUs/DTs can offer. This study focuses on undergraduate education. Similar work could be done focusing on the requirements of graduate education.

References

1. A. J. Dutson, R. H. Todd, S.P Magleby, and C. D. Sorensen, "A review of literature on teaching engineering design through project-oriented capstone courses," *Journal of Engineering Education*, vol. 86, no. 1, pp. 17-28, Jan. 1997.

2. L. J. McKenzie, M. S. Trevisan, D. C. Davis, and S.W. Beyerlein, "Capstone design courses and assessment: A national study," in *Proceedings 2004 ASEE Annual Conference and Expo.,* Salt Lake City, Utah, *USA, 2004*, pp. 1-14.

3. J. A. Marin, J. E. Armstrong, and J. L. Kays, "Elements of an optimal capstone design experience," *Journal of Engineering Education*, vol. 88, no. 1, pp. 19-22, Jan. 1999.

4. S. Howe and J. Wilbarger, "2005 National Survey of Engineering Capstone Design Courses," In *ASEE Annual Meeting*, Chicago, IL., 2006, pp. 6-7.

5. M. Slade, M. H. Jones, and J. B. Scott, "Choosing the right microcontroller: A comparison of 8-bit Atmel, Microchip and Freescale MCUs," Faculty of Engineering, The University of Waikato, Hamilton, New Zealand, Tech Rep. http://hdl.handle. net/10289/5938 , Nov. 2011.

6. D. Vyas, "Microcontrollers: options and trends in today's market," in *ACM Proceedings International Conference and Workshop on Emerging Trends in Tech*nology, Mumbai, India, 2010, pp. 1019-1019.

7. S. Howe, “Where are we now? Statistics on capstone courses nationwide,” *Advances in Engineering Education*, vol. *2*, no. 1, pp. 1-27, spring, 2010.

8. R. H. Todd, S. P. Magleby, C. D. Sorensen, B. R. Swan, and D. K. Anthony, “A survey of capstone engineering courses in North America,” *Journal of Engineering Education*, vol. 84, no. 2, pp. 165-174, April, 1995.

9. R. H. Klenke, “A UAV-based computer engineering capstone senior design project,” in *Proceedings 2005 IEEE International Conference on Microelectronic Systems Education*, Anaheim, CA, 2005, pp. 111-112.

10. A. V. Deshmukh, *Microcontrollers: theory and applications*, New Delhi, India: Tata McGraw-Hill Education, 2005, ch 1, sec1.6, pp.8.

11. M. K. Parai, B. Das, and G. Das, “An Overview of Microcontroller Unit: From Proper Selection to Specific Application,” *International Journal of Soft Computing*, vol. 2, no. 6, pp. 228-231, Jan., 2013.

12. J. J. Vaglica and P.S. Gilmour, “How to select a microcontroller,” *IEEE Spectrum*, vol. 27 no. 11, pp. 106-109, Nov., 1990.

13. H. W. Huang, *The HCS12/9S12: An Introduction to Software and Hardware Interfacing.* Clifton Park, NY: Delmar, 2009, ch. 13, sec. 13.2, pp. 634.

14. G. S. Gupta and C. Moi-Tin, “New frontiers of microcontroller education: Introducing SiLabs ToolStick University daughter card,” in *IEEE International Conference Sensor Networks, Ubiquitous and Trustworthy Computing*, Taichung, Taiwan, 2008, pp. 439-444.

15. B. Esplin, “Microcontroller Market Share: In 3 Dimensions,” *The Databeans Monthly*, vol. *10*, no. 3, pp. 1-3. March, 2012.

16. (2013, Feb 12) *Arduino Board Mega* [Online]. Available: http://arduino.cc/ en/ Main/arduinoBoardMega

17. (2013, Feb 12) *StartUSB for PIC* [Online]. Available: http://www.mikroe. com/startusb /pic/

18. (2013, Feb. 12) *MCU Launchpad Evaluation Platform* [Online]. Available: http://www.ti.com/ww/en/launchpad/msp430_head.html

19. (2013, Feb. 12) *Wytec HCS12 Development Boards* [Online]. Available: http://www. evbplus.com/ ThunderBird12_9s12/ThunderBird12_9s12.html

20. (2013, Feb. 12) *Mbed microcontrollers* [Online]. Available: http://mbed.org/ handbook/mbed-Microcontrollers

21. (2013, Feb. 12) *Microprocessors and Microcontrollers* [Online]. Available: http://am.renesas.com/products/mpumcu/index.jsp?campaign=gn_prod

22. (2013, Feb. 12) *Infineon Micro controllers* [Online]. Available: http:// www. infineon.com/cms/en/product/microcontrollers/channel.html?channel=ff80808112ab681d0112ab6b2dfc0756

23. (2013, Feb. 12) *Microcontrollers - United States* [Online]. Available: http://www. fujitsu.com/us/semiconductors/microcontrollers/

24. (2013, Feb. 12) *Samsung Product Catalog*. [Online]. Available: http:/ /www.samsung .com/global/business/semiconductor/product/application/catalogue

25. (2013, Feb. 12) *MSP 430 16-bit ultra-low power MCU* [Online]. Available: February 12, 2013, from http://www.ti. com/product/msp430g2553

26. (2013, Feb, 12) *Arduino Board Uno [Online.* Available: http://arduino.cc/ en/Main/arduinoBoardUno

27. (2013, Feb. 12) *Arduino – Environment* [Online]. Available: http://arduino.cc/ en/Guide/Environment

28. (2013, Feb. 12) *MicroC PRO for PIC – C Compiler* [Online]. Available: http://www.mikroe.com/mikroc/pic/ide/

29. (2013, Feb. 12) *Code composer studio IDE v5* [Online]. Available: http:// www.ti.com/tool/ccstudio#descriptionArea

30. (2013, Feb. 12) Mbed Compiler [Online]. Available: http://mbed.org/ handbook/mbed-Compiler

31. (2013, Feb.) *Embedded Programming* [Online]. Available: http://fab.cba.mit.edu/classes/4.140/people/theodora.vardouli/06_embeddedProgramming/06.html

32. (2013, Feb. 12) *PIC18F2550* [Online]. Available: http://www.microchip.com/ wwwproducts/Devices.aspx?dDocName=en010280

33. (2013, Feb. 12) *MC9S12DG256FS* [Online]. Available: http://www.free scale.com/files/microcontrollers/doc/prod_brief/MC9S12DG256FS.pdf

34. (2013, Feb. 12) *Microcontrollers – LPC11u2x Mbed Module* [Online]. Available: http://ics.nxp.com/support/development.hardware/mbed.lpc11u2x/

35. (2013, Feb. 12) *Atmega328* [Online]. Available: http://www.atmel.com/ devices /atmega328.aspx?tab=parameters

36. (2013, Feb. 12) CodeWarrior Development Tools [Online]. Available: http://www.freescale.com/webapp/sps/site/homepage.jsp?code=CW_HOME

37. (2013, Feb. 12) Arduino Board Due [Online]. Available: http://arduino.cc/en/Main/ArduinoBoardDue

38. (2013, Feb 12) *The Maple* [Online]. Available: http://leaflabs.com/devices/maple/

39. (2013, Feb. 12) *STM32VLDISCOVERY* [Online]. Available: http://www.st.com/internet/evalboard/product/250863.jsp

40. (2013, Feb. 12) *Basic Stamp 2 Microcontroller Module* [Online]. Available: http://www.parallax.com/ Store/Microcontrollers/BASICStampModules/tabid/134/ProductID/1/List/1/Default.aspx?SortField=UnitCost,ProductName

41. (2013, Feb. 12) P8X32A QuickStart [Online]. Available: http://www. parallax.com/Store/Microcontrollers/Propeller DevelopmentBoards/tabid/514/ProductID/748/List/0/Default.aspx?SortField=ProductName,ProductName

Biographical Information

Jeevan F. D'Souza is an associate professor of engineering and information sciences at DeVry College of New York. He holds a Ph.D. in computer science from Nova Southeastern University. His research interests include engineering education and pattern recognition. He is a senior member of IEEE and a member of ASEE.

Andrew D. Reed holds a B.A. in linguistics from Western Washington University and a B.T. in Electronics Engineering from DeVry College of New York. His research interests include engineering education, natural language processing and machine learning.

Kelly Adams is an associate professor of engineering and information sciences at Devry University Online. She holds an MSc in Electrical Engineering from Georgia Institute of Technology. She previously worked for Cisco Systems and Lucent Technologies. Her research interests include engineering education and pattern recognition. She is a member of IEEE, ASEE, and SWE.