COMPARISON OF DIFFERENT DEVELOPMENT KITS AND ITS SUITABILITY IN SIGNAL PROCESSING EDUCATION

DongYuan Shi and *Woon-Seng Gan*

Digital Signal Processing Lab, School of Electrical and Electronic Engineering,
Nanyang Technological University, SINGAPORE

ABSTRACT

With the availability of many low-cost programmable development kits in the market, real-time signal processing projects can now be readily introduced into today's signal processing and embedded system course curriculum. In this paper, we group these popular development kits in terms of their cost, hardware architecture, development methodology and software resource. We further illustrate the programming efforts in implementing a real-time digital signal processing algorithm using different types of programmable development kits.

Index Terms— Real-time Signal Processing, Signal Processing Education, and Development Kits.

1. INTRODUCTION

With the advents of many low (or moderate)-cost, portable, USB based, readily available programmable development kits (PDKs) from Arduino[1], Raspberry Pi[2], Teensy[3], Intel, ST Microelectronics, Texas Instruments, National Instruments, Xilinx, and many others, it now becomes very convenient to use one of these development kits to learn how to implement real-time digital signal processing algorithms and build an embedded system readily. These PDKs are now being used in many universities to conduct hands-on classes or hackathon sessions for students to learn how to implement a real-time signal processing system and making signal processing come alive[4] [5][6][7][8]. These hands-on educational tools can play a vital role in promoting the importance of digital signal processing education and make an abstract signal processing concept become more tangible. Furthermore, through these PDKs, students can now see, hear, touch, and understand how sensors are being used and interfaced to the processor inside their smart phones, tablet, and smart watch when an app is being activated.

In this paper, we carry out a comparative study on how different popular PDKs differ from each other, and group them under different price range, and further examine the hardware and software development environment in these groups of PDKs. We further illustrate the steps taken to program a real-time signal processing algorithm in C and port to different types of PDKs to highlight the programming effort to implement a functional embedded system. Unlike the traditional DSP development boards that requires AC adapter to power up, and cost more to access to unnecessary high-speed peripheral interface and extensive development suite, recent PDKs are packed with just enough processing capabilities with add on peripheral daughter boards, and powered by USB battery pack. Notebook, which hosts the integrated development environment (IDE) software[9], provides a portable platform for students/hobbyists to program these PDKs and turns it into an interesting project outside their classroom, and thus, engaging students with anytime, anywhere learning.

However, there are not many papers that compare the different features of these PDKs, and highlight the effort in programming real-time signal processing on these PDKs. Some PDKs may be more suitable for freshmen/sophomore students or non-engineering students; while more advanced PDKs are necessary for senior students or postgraduate students. For some of the more advanced PDKs, they can even serve as useful proof-of-concept prototypes to illustrate new research ideas. However, some PDKs are more suited for a certain type of signal processing based application due to its hardware peripherals. Therefore, this paper is our attempt to compare some of the popular PDKs and provide guidelines in programming real-time signal processing algorithms onto these platforms.

This paper is organized as follows. In the next two sections, we will group the PDKs in terms of their hardware architectures, and software development environment. Section 4 outlines the steps in programming a real-time spectral analyzer onto different PDKs. In particularly, we highlight the different programming effort for different PDKs and give a conclusion in using different types of PDKs for conducting different level of courses.

2. HARDWARE ARCHITERCTURES OF DEVELOPMENT KITS

Undoubtedly, price is one of the most important aspects considered when choosing a PDK for implementing a real-time signal-processing project. This paper divides these development boards into five groups, which range from low to high price. The first group of PDK is priced at US$20 and below. It includes Teensy LC, Teensy 3.1, leaf maple[10], Arduino min, Arduino micro, Arduino UNO,

STM32F3Discovery family and NUCLEO family[11]. The second group, which include Arduino Nano, Arduino Mega 2560, Arduino Zero, 86Duino Zero[13], Raspberry pi, Beagle Bone, Pcduino[13], STM32F4 Discovery family and FIO, has a price range from US$20-US$50. The third group (US$50-US$100) comes with higher speed processors and includes Link ONE, 86Duino One, Intel Edison family, Intel Galileo, Cubieboard Family, Radxa Rock and NET Gadgetee[14]. The fourth group, which mainly consists of FPGA/ARM processors, has a price range from US$200-US$1,500. PDKs that belong to this group are Altera DE0, Altera DE1[15], Zynq-7010[16] boards and MyRIO. The final group, which costs above US$1,500 consists of Altera DE2 to DE4, Spartan-6 FPGA Industrial Video Processing Kit[17], Virtex-6 FPGA DSP Development Kit[18] and Kintex-7 FPGA DSP development kit[19] and they are mainly used for developing high-end video applications. This latter group of PDK may be more suitable for developing research prototypes for multi-channel, high resolution signal processing applications. We shall next investigate some common hardware architectural features that are common to the different groups of PDKs:

Group 1: This group mainly consists of the low-end Arduino PDKs, which are embedded with 8-bit or 16-bit AVR microcontrollers, such as the ATmega8, ATmega168 and ATmega328 chips. These microcontroller also consists of an on-chip 10-bit ADC with a maximum sampling frequency of 15 kHz and a hardware multiplier. The clock frequencies in these boards are usually less than 16 MHz. The general I/Os on boards make them possible to connect various expansion boards (shield). These expansion boards include gas sensor, 3 axes accelerometer, humidity temperature pressure sensor, ECG, EEG sensor and so on. With the help of these shields, these boards are widely used in environment or health monitoring, such as water quality testing platform, communication system (such as in Arduino Phone), and precision control device (such as in Ardu-Pilot and 3-D printer.)

Group 2: This group of PDKs is generally characterized by using 32-bit ARM cortex (M0,M3,M4) as the main processors (see Table 1). The ARM cortex M series is a range of scalable and compatible, energy efficient, and easy-to-program processors. It is generally used to fulfill the requirements of smart and high performance embedded applications. The ARM cortex M processors often have superior code density compared to 8-bit and 16-bit microcontrollers. Furthermore, the ARM cortex processors offer floating-point unit (FPU) and hardware divider. As shown in Table 1, Teensy and Arduino zero are both powered by Cortex M0, which has extremely low power consumption so that they are largely applied in IoT and wearable devices. The Cortex M3 processor in Arduino Due and Leaf maple boosts the performance of these devices to achieve performance efficiency of 1.89 DMIPS/MHz. With the help of the Cortex M4's single 32-bit multiply-accumulate computation (MAC) engine with SIMD

Table 1. The core processor, main clock frequency, memory size, ADC's and DAC's features of boards in Group 2.

PDKs	Clock (MHz)	RAM (Kbyte)	ROM (Kbyte)	ADC		DAC
				(MHz)	(Bit)	(Bit)
LC	48	8	64	1	16	12
zero	48	32	512	1	16	12
due	96	96	512	1	12	12
maple	72	20	128	1	12	12
Teensy 3.1	96	64	256	0.8	16	12
F334-R8	72	12	64	5	12	12
F401-RE	84	96	512	5	12	12
Disc-overy	168	192	1,000	2.4	12	12
LinkIt	260	4,000	16,000		12	

arithmetic, Teensy 3.1, NUCLEO-F334R8, NUCLEO-F401RE and STM32F4Discovery[20] are suitable for high performance signal processing algorithm development. The The LinkIt ONE[21] PDK successfully combines the ARM7 processor with WIFI (MT5963), GPS (MT3332) and MediaTek Aster (MT2502) into one single board, which provides on of the best choice for IoT/wearables devices.

Group 3: This third group of PDKs has common hardware architectures of high-end processors with clock speed of up to hundreds of MHz[22][23], as shown in Table 2. For example, Intel Galileo[22] is a compact microprocessor board based on the Intel Quark SoC X1000, which belongs to the 32-bit Intel Pentium-class and operates at 400 MHz system clock. Therefore, these boards have high performance and are widely used in IoT and wearable devices.

Group 4: This group of PDKs generally has excellent video and image processing performance[24][25][26], because most of these boards own graphic processor unit(GPU), as shown in Table 3, which is dedicated to accelerate image rendering through frame buffer. These boards also support lots of video accessories, such as Raspberry camera, which can produce 1080p or 720p video and image.

Group 5: PDKs, which are in the last group, are all FPGA development boards. They are inherently suitable for high speed digital signal processing[27] due to the FPGA excellent parallel processing ability.

3. SOFTWARE ARCHITECTURES OF DEVELOPMENT KITS

In this section, we outline the integrated development environments (IDE) used to implement real-time signal-processing algorithms onto the 5 groups of PDKs. The Arduino IDE is a high efficient programming platform. Majority of the PDKs in Group 1 to Group 3, which are based on Arduino platforms, can be easily programmed in the Arduino IDE.

Table 2. The main clock frequency, memory size, ADC's and DAC's features of boards in Group 3.

Intel Board	Clock (MHz)	Ram (Mbyte)	Rom (Mbyte)	ADC	
				(MHz)	(Bit)
Galileo	400	265 (DDR3)	8 (Flash)	1	12
86Duino zero	300	128 (DDR3)	8 (Flash)		
86Duino one	300	128 (DDR3)	8 (Flash)		
Intel Edison	400	1,024 (DDR3)	4096 (Flash)		
Edison Break	400	1,024 (DDR3)	4096 (Flash)		

Table 3. The Processor, GPU, clock frequency and memory size of boards in Group 4.

PDKs	Raspbery Pi	Beagle Bone	Cubie Board	RADXA ROCK	MyRIO
Multi core	1	1	2	4	2
	Arm 11	Cortex A8	Cortex A7	Cortex A9	Cortex A9
DSP core		TMS320 C4			FPGA core
GPU	Video Core IV	PowerVR SGX350	Mali400 MP2		
Clock (MHz)	700	700	1000	1600	50
Ram Mbyte	265 (RAM)	512 (DDR)	2048 (DDR3)	2048 (DDR3)	.276 (Cash)
Rom Mbyte		4098 (Flash)	8192 (Flash)	8192 (Flash)	.256 (chip)

Table 4. The Processor, embedded memory, logic element, multiplier of boards in Group 5.

PDKs	Core	Embedded Memory	Logic element	18by18 multiplier
Altera DE3	EP3SL 150	5,499 (Kbyte)	142,000	384
Altera DE4	EP4SGX 230	17,133 (Kbyte)	228,000	1288
Virtex6 kit	XC6VLX 240T	3,650 (Kbyte)	241,152	DSP48E1 768
Virtex7 kit	XC7VLX 690T	37,080 (Kbyte)	485,760	DSP48E1 2800

In Figure 1(a), the Arduino IDE follows the typical editor-compiler-link-run IDE. Signal processing algorithm can be coded in C and compiled into binary files, before downloading into the Arduino-based PDKs using the USB-serial port connection. With a comprehensive suite of high-level and user-friendly API functions for I/O interface and peripheral configuration, development time can be focused mainly on algorithm programming. The developed code for can be easily ported among the Arduino-based PDKs, which allow scalability of project from entry to advanced complexity.

Figure 1(b) shows a more traditional DSP development process. DSP C/assembly code must be written for both algorithm and system/peripheral configuration in IDE like IRA[28], Kiel[30], Eclipse[29], etc. The PDKs, which use

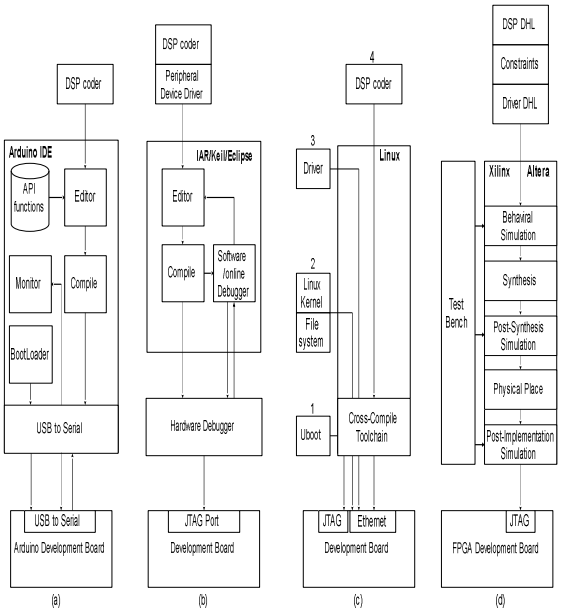

Figure 1. Software development methodologies for different PDKs

this process to program, include the STM32FxDiscoveray family. Hardware debugger is used to ensure both program logic and configuration/syntax errors are validated before proceeding to download the code onto the PDKs. This low-level programming demands the users to have a good understanding about the software development and hardware configuration on the board.

The third type of IDE supports operating systems (OS), like Window, Linux, Mac OS, etc., and it is used to program higher complexity (with multimedia, multi-function, multi-rate, and multi-channel requirements) signal processing algorithms. Some of Group 4's PDKs, like Raspberry pi, Beagle Bone, Cubie board and Radxa rock usually use this type of IDE to take advantage of the powerful processor and GPU. Figure 1 (c) demonstrates how to implement signal processing algorithms onto this group of PDKs based on the embedded Linux. An open-source bootloader program, Uboot[22] is used to load the Linux OS and driver API into the Flash of the PDKs. The programming code can be compiled either in PC (on cross compiler) or in the PDKs. In order to take advantage of the GPU processing capability in Group 4's PDKs, OpenGL[23]/OpenCV[24] can be used for their rich library functions. However, this approach is more cumbersome to program, and has a steeper learning curve compared to the previous IDEs.

The fourth type of IDE, as shown in Figure 1(d), is mainly for Group 5 PDKs. Algorithms must be written in low-level HDL language and assign hardware constraints. The popular Xilinx[35]/Altera IDE[36] compile these DSP HDL codes and constraints. Behavior of the HDL code can be measured and analyzed, before synthesizing the HDL code into gate netlists, which will be further tested. Next,

the gate netlist is placed and routed to form the digital circuit. The final verification stage is carried out to examine time compliance, before downloading onto the FPGA[37] PDKs. The FPGA development is often time consuming compared to the other IDEs, and users must have a good knowledge of digital circuit design.

The final type of IDE, is known as graphical programming, as contrast to the conventional text programming. One such IDE is the LabVIEW and it offers user the convenience of linking programming blocks together to form the system diagram. This high-level and high-visualization program development is highly suited for students without much programming knowledge, and provides an entry level into embedded system programming. However, optimized code is not feasible due to the headroom in embedding automatic coding. The MyRio[38] PDK is one such platform that uses the graphical programming.

4. IMPLEMENTATION OF REAL-TIME SIGNAL PROCESSING

In this section, we evaluate the programming effort and what students need to know before embarking on the programming. We used an example of programming a real-time spectral analyzer on four PDKs coming from different groups to illustrate. Table 5 shows the detailed setting among these boards.

In this programming exercise, students need to understand the basic operation of the analog-to-digital conversion (ADC), the concept behind FFT computation, and how digitized signal can be displayed in the LCD monitor. Under the Arduino group (Group 1), we chose the Arduino Uno and Mega boards, which have an integrated on-chip ADC, and using the ***analogRead()*** function found in the Arduino IDE to continuously acquire value from the ADC. Due to its computational limitation, 64- to 512-point float point or fixed point FFT can only be programmed onto these boards. The C code is added into the ***loop()*** function, which is a part of the Arduino software framework. The ***SPI.transfer()*** function of Arduino is inserted into the code to control the LCD display of the spectrogram. The programming effort is low due to the direct library programming approach in Arduino at the expense of limited hardware control.

In contrast, the STM32F4 Discovery board in Group 2 requires students to obtain extra information to configure the phase lock loop (PLL), ADC, and serial port interface (SPI) in a low-level programming of relevant register banks to set the clock, sampling frequencies and resolution of ADC to 168 MHz, 48 kHz and 16-Bit, respectively. The Keil IDE comes with a suite of floating-point (FPU enabled) signal processing C routines that allows students to modify for different order of radix-2 FFTs. However, accessing the LCD requires addition effort in programming a C code that control and display spectrogram onto the LCD. Unlike the Arduino boards in Group 1, which uses USB connection to download the code, a dedicated hardware debugger is used to debug and download this C code in the Discovery board.

Table 5. System clock, consumption of memory, sample rate of three different boards for implementation of spectral analyzer

Name		Uno	Mega	Discovery	Galileo
System clock (MHz)		16	16	168	400
Sample rate (kHz)		10	10	48	1,000
128 FFT	RAM(Kbyte)	0.512	3	24	62.3
	ROM(Kbyte)	1	5.5	12	
512 FFT	RAM(Kbyte)		6	27	65
	ROM(Kbyte)		6.7	14	
1024 FFT	RAM(Kbyte)			29	67.5
	ROM(Kyte)			20	

For the higher-end boards in Group 3, like the Intel Galileo, an operating system (such as uLinux) must first be loaded onto the board through the micro-SD card. The analog signal acquisition is carried out by calling an API library to bring in the sampled data and the sample rate of the ADC can go as high as 1 MHz. A C code can be programmed to realize a 512-point and 1024-point FFT. An API library for LCD display is also provided. However, a cross compile toolchain must be used to compile the code into executable file to load into the Galileo board.

For small size projects, where high resolution processing algorithm may not be required, the Group 1 PDKs provide the fastest approach to implement the real-time code and illustrate the signal processing concepts. A more advanced project, which requires OS to perform multitasking and resource management, the more expensive Group 3 PDKs provide a good memory-performance option. Group 2 PDKs, which come in between the two, provide an attractive entry level for students, who wanted to program high resolution real-time signal processing applications. A set of spectral analyzer C codes, which are used in the above implementation, can be found in https://github.com/StoneSmart223/FFT/).

5. CONCLUSIONS

In this paper, we reviewed several popular PDKs and grouped them under 5 groups in terms of their price range. Most of these PDKs can be programmed in floating point, and thus, removed the fixed-point programming issues that impede real-time implementation in the past. In addition, most of these boards support Arduino IDE, which can be easily learned by students on their own, and the code developed can be easily ported from one Arduino platform to another. In summary, the Group1 PDKs are mainly for simple I/O control applications, Groups 2 and 3 PDKs are very useful as a starting point for learning real-time signal processing of audio/speech signal, while Groups 4 and 5 provide excellent platforms for more advanced image/video processing projects, and building research prototypes.

ACKNOWLEDGE

This work is supported by the Singapore Millennium Foundation Grant, under project code: M4061449.

REFERENCES

[1] Arduino website, "Getting Started with Arduino", https://www.arduino.cc/en/Guide/HomePage.

[2] Wikipedia website, "Raspberry Pi Introduction", https://en.wikipedia.org/wiki/Raspberry_Pi.

[3] Teesy website, "Teensy Introduction", https://www.pjrc.com/teensy/teensyLC.html.

[4] C.H.G Wright, M.G. Morrow and T.B. Welch, "Comparison of DSP boards for classroom use" in *Digital Signal Processing and Signal Processing Education Meeting (DSP/SPE)*, 2013 IEEE, 2013, pp. 273–278.

[5] W.S Gan., A. Seth, and S.M. Kuo. "Versatile and portable DSP platform for learning embedded signal processing." in *Acoustics, Speech and Signal Processing (ICASSP)*, 2011 IEEE International Conference on. 2011.

[6] A. Seth and W.S.Gan. "Fixed-point square roots." in *Acoustics, Speech and Signal Processing (ICASSP*), 2012 IEEE International Conference on. 2012.

[7] W.S.Gan and Sen M. Kuo, *Embedded Signal Processing with the Micro Signal Architecture*, Wiley-IEEE press, 2007 ISBN:0471738417.

[8] Sen M. Kuo, W.S.Gan, *Digital Signal Processors: Architectures, Implementations, and Applications, Prentice Hall*, Published 03/26/2004 ISBN-10: 0130352144 • ISBN-13: 9780130352149.

[9] Arduino IDE Websit, "Arduino software - Introduction", www.arduino.cc.

[10] Maple website, "Maple Introduction", http://www.leaflabs.com/device-details/.

[11] ST website, "STM32 MCU Nucleo", http://www.st.com/web/catalog/tools/FM116/SC959/SS1532/LN1847?sc=stm32nucleo.

[12] Wikipedia website, ".NET Gadgeteer Introduction", https://en.wikipedia.org/wiki/.NET_Gadgeteer.

[13] 86Duino website, "Products 86Duino boards", http://www.86duino.com/?page_id=11.

[14] L.F.Zhao and J.F.Liu, *Introduction to pcduino*, publish on website, https://s3.amazonaws.com/pcduino/book/Introduction+to+pcDuino.pdf.

[15] TerasIC website, "DE Serial Introduction ", http://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&CategoryNo=163.

[16] Xilinx website, "Zynq-700 Silicon Device", http://www.xilinx.com/products/silicon-devices/soc/zynq-7000/silicon-devices.html.

[17] Fpga blog website, "Xilinx Spartan-6 FPGA Industrial Video Processing Kit", http://fpgablog.com/posts/avnet-aes-s6ivk-lx150t-g/.

[18] Business Wire website, "Avnet Virtex-6 FPGA DSP Development Kit Jump-Starts DSP Designs with Industry's Most Widely Adopted DSP Design Flows", http://www.businesswire.com/news/home/20100210005318/en/Avnet-Virtex-6-FPGA-DSP-Development-Kit-Jump-Starts#.VgT1R7OqpBc.

[19] AVNET website, "Xilinx Kintex FPGA DSP Development Kit with High-Speed Analog", http://www.em.avnet.com/en-us/design/drc/Pages/Xilinx-Kintex-7-FPGA-DSP-Development-Kit-with-High-Speed-Analog.aspx.

[20] ST website, "STM32F4DISCOVERY Introduction", http://www.st.com/web/catalog/tools/FM116/SC959/SS1532/PF252419.

[21] Seeed wiki, "Linkit one Introduction", http://www.seeedstudio.com/wiki/LinkIt_ONE.

[22] Arduino website, "Intel Galileo Introduction", https://www.arduino.cc/en/ArduinoCertified/IntelGalileo.

[23] Intel website, "Intel Edison-One Tiny Module, Endless Possibility", http://www.intel.com/content/www/us/en/do-it-yourself/edison.html.

[24] Wikipedia website, "BeagleBoard Introducntion", https://en.wikipedia.org/wiki/BeagleBoard.

[25] Wikipeida website, "Cubieboard Introducntion", https://en.wikipedia.org/wiki/Cubieboard.

[26] Radxa website, "radxa Introducntion", http://wiki.radxa.com/Rock.

[27] DY Shi and H. Zishu, "A new frequency source based on Sigma Delta and CORDIC," in *Consumer Electronics, Communications and Networks (CECNet),* 2012 2nd International Conference on. 2012.

[28] Wikipedia website, "IRA Introducntion", https://en.wikipedia.org/wiki/IAR.

[29] Wikipedia website, "Eclipse (software) Introducntion", https://www.wikipedia.org/.

[30] Keil website, "Keil software Introduction", http://www.keil.com/.

[31] Wikipedia website, "Linux Introducntion", https://en.wikipedia.org/wiki/Linux.

[32] Wikipedia website, "Das U-Boot Introducntion", https://en.wikipedia.org/wiki/Das_U-Boot.

[33] OpenGL website, "Welcome to OpenGL wiki!", https://www.opengl.org/wiki/

[34] OpenCV website, "Welcome to opencv documentation!", http://docs.opencv.org/.

[35] Xilinx website, "ISE Design Suit", http://www.xilinx.com/products/design-tools/ise-design-suite.html.

[36] Altera website, "QUATUS II SOTFWARE", https://www.altera.com/products/design-software/fpga-design/quartus-ii/overview.html.

[37] Wikipedia website, "Field-programmable gate arrary Introducntion", https://en.wikipedia.org/wiki/Field-programmable_gate_array.

[38] NI website, "What's New for NI myRIO", http://www.ni.com/white-paper/52419/en/.