DSP EVOLUTION FROM A TEACHING POINT OF VIEW

Dr N. Dahnoun and Dr J. Brand*

School of Electronics, University of Bristol
Merchant Venture Building, Woodland Road, Bristol, BS8 1UBR, Bristol, UK
phone: + (44) 1179545181, email: Naim.Dahnoun@bristol.ac.uk
* Texas Instruments, UK, email: jason-brand@ti.com

ABSTRACT
In this paper the authors are addressing the concerns associated with fast growing DSP chips and tools and the impact they have on teaching DSP implementation. The authors also provide solutions, advice and suggestions on how to select a DSP, set a DSP implementation course and the associated laboratory hardware and software that fit a specific application. To help differentiate between the multitude of high performance DSPs, vendors are developing market-specific DSPs as an alternative to more generic DSPs. This is putting pressure on educators to update courses and reinvest in hardware and software more often than for traditional courses. For the less experienced or new educators, it is a daunting task to cope with this fast evolving technology, to make an informed choice and select the right tools. Choosing the wrong tools not only can be very expensive and time consuming but can also have a negative impact on teaching and learning outcomes.

1. INTRODUCTION
Digital signal processor hardware and software are growing at a vertiginous rate and that is due mainly to the emerging and complex applications. If we divide DSP applications by algorithm type, we find four main categories: audio, speech, imaging and video. We have observed a noticeable trend for increasing algorithm complexity and also increasing number of features (profiles, levels, standards etc). For instance, multiple channels, higher resolutions, higher bit rates, and more complex standards are becoming the norm for many applications. These algorithmic demands have forced silicon vendors to develop higher performance DSPs. However, due to the physical limitations in transistor size and switching frequency, it is not always possible to meet these demands with shear processing power alone. If we divide the market by application type e.g. multimedia, digital communications, automotive, video surveillance, medical imaging etc., we see that two requirements are needed: one is for handling the processing of the relevant data; the other for handling the user interface. These diverse market demands have led silicon vendors to develop flexible homogenous and heterogeneous architectures including high speed DSPs, SoC DSPs with accelerators and SoC multi-core DSPs to target high performance demanding applications. To add to the confusion, many of these processors also run different operating systems on different cores. These architectures raise several pertinent questions, crudely separated into hardware and software:

Hardware Questions:
- Why use a DSP and not an ASIC or an FPGA?
- Do I need floating point or fixed point?
- Why is it better to use multiple processors, SoC DSPs with accelerators or a SoC multi-core processor?
- Which key features should we look for?
- What integrated peripherals are available?
- How many audio ports are available?
- How many video ports are available?
- What are the characteristics of these ports?
- Are the video or audio ports sampled simultaneously?
- What is the power consumption of these devices?
- What is the footprint of these devices?
- Does the DSP support JTAG?
- Does the DSP support embedded trace?
- Does the DSP platform incorporate emulation hardware or does it require an external emulator?
- What development platforms are available?
- What is the cost of the DSP platform and the DSP chip?
- Who supplies DSP platforms and do they conform to the EU regulations?

Software Questions:
- Which operation system(s) do I use and why?
- Do I need to license the OS, libraries and tools?
- Are peripheral drivers available out-of-the-box?
- Do I need to write in assembler or can I use generic C?
- If I use C, what is the efficiency of the compiler?
- Can the DSP vendor provide support and training?
- Can the DSP vendor provide an educational discount?
- For multi-core DSPs, how easy is it to write “parallel code”?
- Will the system design be different on a single core?
- What language(s) are used to program a DSP?
- Which performance benchmark method to look for?
- The list is inexhaustible and the answer to these questions is very application and time dependent as DSPs are continuously evolving at a very fast rate. For instance, in the first half of 2010, Texas Instruments launched the following embedded processors targeting a wide range of application areas: C5514/5 (audio, biometrics), DM6467T (video), TMS570 (automotive), DM368, DM8168, DMVA1 (video security), AM1705/7, AM1806/8 (industrial) and AM3705/15 ( portable) giving a total of 13 new parts based on 7 different cores. In addition to these individual parts, they also released the following application specific reference design kits: C5515 fingerprint...
kit, C2000 DC/DC LED Lighting kit and the DM368 Video Conferencing Developers Kit. Similarly, ADI and Freescale also launched a wide range of new parts.

Once these technical questions are answered there will be more questions that need to be addressed if the DSP implementation in education is to be enhanced, that is:

1. What is the technical knowledge of the educator and how can he or she be trained?
2. What is the knowledge of the students and what is the pre-requisite for the course?
3. Which material to use.
4. How do you assess progress and set exams?

The rest of this paper is divided in 5 Sections. Section 2 provides a general knowledge, Section 3 shows how to select a DSP platform, Section 4 provides an overview of different DSP types and also draws attention to some of the concerns behind programming multi-core DSPs. Section 5 discusses issues related to the setup of a real-time DSP implementation laboratory.

2. GENERAL KNOWLEDGE

Before attempting to answer any question, it is worth at this stage to set good foundations. After running 100s of technical training workshops, we have discovered that over 70 % of engineers cannot justify why to use a DSP instead of a Pentium, PowerPC, FPGA, a System on Chip (SoC) or a microcontroller. Often the decision is based on familiarity of tools (i.e. time-to-market) and not necessarily on picking the perfect architecture.

Historically four basic technologies were used for implementing an embedded application, that is: Application Specific Integrated Circuit (ASIC), Field Programmable Gate Arrays (FPGA), Digital Signal Processor (DSP) and mixed technologies. In recent years, a fifth category has also started to emerge, that of high performance, low power General Purpose Processors (GPP) with DSP like instructions e.g. ARM Cortex-A8/A9.

Development of an ASIC is usually only justified when the volume of units is in the region of millions. The non-recurring engineering cost of an ASIC is in the region of millions of dollars; if the application is fixed they can save cost in the long run but they are inflexible in terms of changes, upgrades and reuse. From all these technologies, ASICs have the longest design to market cycle but have the lowest power consumption and the lowest cost per unit shipped.

FPGAs are not generally suitable for high volume consumer products due to the relatively high power consumption cost and the flexibility in the design can be dominated by “the place-and-route” stage in large devices. FPGAs are clocked at a lower rate than DSPs or ASICs but can perform more operations simultaneously which make them suitable for parallel processing. They are flexible through programmable hardware but are not suited for task performing control, management or floating point operations. However, recently they have seen an improvement in terms of power consumption, cost and performance. FPGAs can perform in very harsh environments such as space and are mainly used in research, telecommunication and military.

DSPs are optimised for size, cost and power consumption. They operate at a high sampling rate and are very flexible in terms of software changes and field upgrades. The choice between using a DSP or an FPGA is primarily application dependent but can also depend on the experience and preference of the developer.

GPPs such as the ARM Cortex-A8 usually abstract the hardware via a high level OS such as Linux or WinCE. This is advantageous for engineers unfamiliar with the limitations of embedded processors. Although GPPs can be low power, their performance is not as optimal as that of a DSP. Choice of GPP usually comes down to the available peripherals and board-support-package.

In an ideal world, DSP and FPGA complement each other in many high performance applications that use these mixed technologies. DSPs are used to implement the flexible software components of the system and FPGA is used to provide high speed interconnect and hardware flexibility.

DSP applications are inexhaustible and range from simple MP3 players to high speed Internet browsing on mobile devices. In general these applications can be divided into six main categories, see Table 1: Voice, Speech, Audio, Imaging, Video and Vision.

<table>
<thead>
<tr>
<th>Voice</th>
<th>Hearing aids, Telephone, Toys</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speech</td>
<td>Synthesis, Recognition, Vocoder</td>
</tr>
<tr>
<td>Audio</td>
<td>Monaural, Stereo, Surround sound, Professional</td>
</tr>
<tr>
<td>Imaging</td>
<td>Digital Still Camera</td>
</tr>
<tr>
<td>Video</td>
<td>Standard definition (SD), High definition (HD)</td>
</tr>
<tr>
<td>Vision</td>
<td>Automotive Vision, Video Surveillance, Machine Vision</td>
</tr>
</tbody>
</table>

Table 1 - DSP applications in categories.

3. SELECTING A DSP PROCESSOR

Selecting a suitable device on its own is not always sufficient. The platform hosting the device is just as important, especially for educational purposes. Universities cannot afford to design platforms as this requires hardware design expertise and high development cost. It is also expensive and very time consuming. Platform selection is trivial once a processor is selected since there is only a limited number of Evaluation Modules (EVMs) for a specific DSP. Selecting a DSP processor is application dependent and factors such as: the arithmetic used, data width, performance, cost, size, on-chip memory size,
peripherals available, technical support, and so on, help in selecting the right DSP. The timely availability of a comprehensive robust Software Development Kit (SDK) for the platform is critical to enable the assessment of complex DSPs. DSP vendors offer roadmaps, benchmarks and application notes to help select a processor. Some vendors also offer board support packages and one-to-one technical support. This service can be invaluable when wading through so much information. Companies such as BDTI can provide comparative benchmarks for various vendors for both DSPs and FPGAs [1]. Processor vendors can be divided into two categories; the first category is proprietary cores (ADI, Freescale, TI etc) and the second is licensed cores (see Table 2).

<table>
<thead>
<tr>
<th>Licensor</th>
<th>Family</th>
<th>Floating, Fixed, or Both</th>
<th>Data Width</th>
<th>Core Clock Speed (MHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARM</td>
<td>ARM7</td>
<td>Fixed point</td>
<td>32 bits</td>
<td>145</td>
</tr>
<tr>
<td>ARM9E</td>
<td></td>
<td>Fixed point</td>
<td>16/32 bits</td>
<td>456</td>
</tr>
<tr>
<td>ARM1176</td>
<td></td>
<td>Fixed-point</td>
<td>16/32 bits</td>
<td>335</td>
</tr>
<tr>
<td>Cortex-A8/9</td>
<td>Fixed-point</td>
<td>8/16/32 bits</td>
<td>1500+</td>
<td></td>
</tr>
<tr>
<td>Cortex-R4</td>
<td>Fixed-point</td>
<td>16/32 bits</td>
<td>160+</td>
<td></td>
</tr>
<tr>
<td>Cortex-M3/4</td>
<td>Fixed-point</td>
<td>16/32 bits</td>
<td>150+</td>
<td></td>
</tr>
</tbody>
</table>

Table 2 – Examples of core licensors

4. DSP TYPES

Solutions for handling complex multimedia applications can use one or a combination of the following approaches: General purpose processors, hardware accelerators, or SoC multi-core DSPs. General Purpose processors have the capability of supporting various video standards in addition to baseband processing, such as audio signals and synchronization with video. Hardware accelerators can only support a limited number of standards but have the advantage of having low die size and high video processing performance. Video co-processors have a computational speed that is as fast as that of ASIC designs which are optimized for individual specific functions. Due to limitations in process technology and switching frequencies, there is an emerging trend towards multi-core DSP, although this concept is not new. In 1994, Texas instruments announced the TMS320C80 which was the first commercially available single-chip processor to combine multiple parallel DSPs and a RISC processor onto a single chip. In 2003, the TMS320C5561 incorporated five DSPs. Multi-core architectures can be split into two categories: homogeneous and heterogeneous.

1. Heterogeneous architectures can include a DSP and a GPP, a Graphics Processing Unit(s) (GPU(s)) or Micro Controller Unit(s) (MCU(s)).
2. Homogeneous architectures include multiple DSPs or GPPs (i.e. the processors are identical).

There are three main limitations for a single core:

- a. Power consumption and heat dissipation issues associated with the increase in clocks frequencies run in parallel.
- b. Limitation in the number of instructions that can run in parallel.
- c. Memory access bottle neck.

The solution to the problem is to reduce the clock speed (therefore reducing power) and, then, place more cores on a single piece of silicon as long as Moor’s law holds (that will allow true parallelism and solve some of the memory access bottle neck). However, these come at a cost as more pressure is put on to the software developers to write parallel applications. In the past, programming was in a sequential line-by-line manner to replicate the single thread behaviour of most single core processors. Saying that, there are now new tools that alleviate this problem [2, 3 and 4]. The ARM Cortex-A9 Symmetric Multiprocessing (SMP) architecture has started to address this programming paradigm by enabling the OS to choose where to execute code [11].

5. DSP LABORATORY FOR REAL-TIME IMPLEMENTATION

The purpose of these types of laboratories is to familiarise students with the tools, DSP architectures, implementation and optimisation. Due to the nature of the algorithms, it is advisable for the instructor to select an audio or a video laboratory experiment depending on the curriculum and interest. Most DSP manufacturers now provide low cost plug and play evaluation modules (EVMs) that have the necessary peripherals for capturing audio and/or video signals and, in addition to these, Texas Instruments provides free teaching material for various DSP families [5]. The cost of the EVMs is relatively low and educational discounts are also available. The teaching materials provide lecturers with out-of-the-box PowerPoint presentation material and source code for a large number of signal processing algorithms such as FIRs, IIRs, FFTs, DCTs, Edge detection etc.
The application notes are also worth consulting before embarking on such a project. Various DSP libraries such as the fixed point math library, digital filter library and the image library are available [7]. These libraries shorten the development time and improve performance via optimised code [6]. In addition to these standard libraries, TI have started to release application specific libraries, such as the Video Analytics Library, to accelerate algorithm development [7]. There are also technical workshops that run from one to five days that help instructors to quickly get hands-on experience. The material can be downloaded free of charge without the need to register for a course [8]. When looking for material, it is worth being aware of the online sites that provide free material [1, 9] and also the recent engineer-to-engineer forums [10].

If we assume the main focus of the course is to develop students’ DSP algorithm implementation skills, the instructor needs to be aware of the complexity of setting up peripherals and memory. Ideally this needs to be abstracted. Due to the vast amount of code a programmer has to write to complete an application and the small number of sessions available, the instructor needs to carefully prepare a framework/template that enables the student to focus on DSP Algorithm development alone.

Once the laboratory infrastructure is set and prior to embarking on teaching a new DSP course, it is worth dedicating a couple of laboratory sessions for testing the students’ knowledge. To ensure all students are starting from a similar level, a background knowledge test is recommended. It is fair to say that most of the students seem to struggle with the C language and numerical issues. This needs to be addressed at the beginning of the course.

Finally, in academia the students need to be examined based on the content of the module. The practical yet flexible nature of programming means multiple solutions can be correct. Therefore, assessment is not like a traditional exam. A combination of hand written and computer based exams is most appropriate. It is important that the assessment can be automated to save time in marking. Multiple choice questions and graphical outputs can facilitate this. It is also important to incorporate continuous feedback.

6. CONCLUSION

There are a number of issues to consider before selecting a DSP platform for teaching. Educators need to be aware that there is a large amount of free support material available, thanks to silicon vendors and generous online contributors. The standard approach is to create a course based on a single core. You need to start with the basic building blocks. It is important to understand the effort required to update laboratories to train students on the latest DSPs. This can be very challenging even for an experienced professor as it requires advanced knowledge to study the new architectures and peripherals to become familiarised with the new tools and the development board(s). It may even require a complete rewrite of all the content.

There are additional challenges in developing tools for programming and debugging multi-core DSP that puts further pressure on the programmers. The issue for automatically assessing a large number of students without the need of reading and debugging a large number of programmes needs further consideration.

Given the diverse complexity of DSP architectures and software tools, it is clearly a challenge to produce up-to-date, coherent teaching material that closely follows the trends of DSP. There is clear latency between the announcement of silicon and the emergence of relevant training material. Even with early knowledge, the best case latency is a 2-3 year gap. Given the speed at which silicon is continually changing, it is clear that silicon manufacturers need to start working closely with academics at a very early stage to ensure industry and education are aligned.

REFERENCES


Texas instruments, C64x+ Image Library (IMGLIB). [online]: http://focus.ti.com/docs/toolsw/folders/print/sprc264.html

Texas instruments, C67x Fast Run-Time Support (RTS) Library. [online]: http://focus.ti.com/docs/toolsw/folders/print/sprc060.html


