WHITE PAPER

TEXAS INSTRUMENTS

Reaping the Benefits of SoC Processors for Video Applications

By
Jeremiah Golston,
CTO Streaming Media,
Texas Instruments
Rishi Bhattacharya,
Systems and Software Architect,
Texas Instruments

Overview

Today's demanding consumer video applications often require the high performance of system-on-a-chip (SoC) integration, yet SoC processing engines have created new challenges for system developers. SoCs have traditionally been based on closed architectures, giving developers few options for implementation; however, video applications ranging from consumer communications to multimedia products are becoming increasingly complex, requiring design flexibility for greater customization and advanced feature updates.

Original equipment manufacturers (OEMs) often need to use the same system platform across a range of products that are tailored for specific markets; or they need to combine different applications in the same system, such as a security camera with object recognition, or an IPTV set-top box (STB) with integrated video phone or digital media adapter capabilities. As feature-rich, multiple-application products continue to grow in number, system developers increasingly need SoC processors that have been designed with open architectures to meet the versatile, rapidly changing requirements of the consumer video market.

Fortunately, a new type of SoC processor has appeared that integrates high-performance and programmable cores, together with the essential memory and peripherals for building a wide range of consumer video systems. The SoC architecture is based on a programmable digital signal processor (DSP) with video-specific hardware acceleration providing the computational performance needed for real-time compression-decompression algorithms (codecs) and other communications signal processing. Combining a RISC processor with the DSP adds control and user interface support, together with programming ease; and integrated video peripherals reduce system cost and simplify design. Since this multiprocessor hardware serves as the foundation for an open software architecture, the result is an SoC processing engine that enables the flexible, rapid development of robust consumer video products.

An Example SoC Video Processor: Hardware

An example of such an SoC video processing platform is DaVinci™ technology from Texas Instruments. The underlying DaVinci hardware has been designed specifically to support video systems, not only serving to reduce board space and component counts, but also to eliminate much of the low-level software development required to integrate a complex system. The TMS320DM644x digital media processors, which provide this hardware foundation, integrate a TMS320C64x+™ DSP and an ARM926EJ-S RISC processor as cores, with additional hardware acceleration to perform specific operations frequently used by video codecs. The fully programmable cores are supported by on-chip caches that automate program and data memory allocation to simplify design and promote flexibility. Direct Memory Access (DMA) to on-chip SRAMs from external memory is also supported to optimize heavy video data traffic.

The DM644x architecture has also absorbed many of the external components required for digital video, reducing hardware bills of materials by as much as 50 percent. A video processing subsystem includes a front end with an on-chip image pipeline for camera image capture and processing, supporting both BT.656-compliant devices and CCD/CMOS sensors. A back end with an on-screen display (OSD) driver and integrated digital-to-analog converters (DACs) provides analog and/or digital RGB/YCbCr video output. Other integrated features include

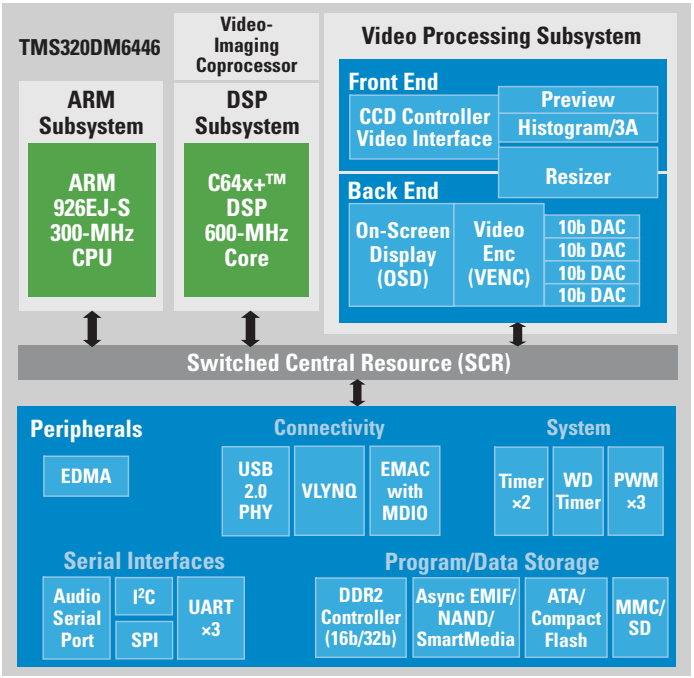

Figure 1. TMS320DM6446 Block Diagram.

networking peripherals, audio-video interfaces, and an enhanced direct memory access (EDMA) controller with support for up to 64 simultaneous transfer channels. Having so many audio-video features integrated in hardware saves programming time and also reduces the need to dedicate software cycles to interfacing with and controlling external devices. Figure 1 shows the functions integrated in one of the first DaVinci™ processors.

An Example SoC Video Processor: Software

The software platform that sits on top of the DM644x processor takes full advantage of the hardware for performance, while at the same time abstracting the complexity of programming the underlying hardware for greater versatility and ease of use. With two types of processing cores available, software can be designed to operate where its execution is most efficient: the user interface and system control on the RISC, and real-time signal processing algorithms such as codecs on the DSP, with boosts from the accelerators. Each core is programmable, so that the processor as a whole will support whatever code the developer needs to create for a video system. However, the platform makes the DSP transparent to application developers, so that only the RISC needs to be programmed, greatly simplifying the development task. Those developers who want to program the DSP with its accelerators can still do so, and the open software platform enables choosing the type of development environment best suited to the needs of the project. Much of the software required for a video system—from peripheral drivers to memory management to essential codecs—has already been created. As a result of the high level software integration and flexibility in the open DaVinci platform, video system development time can be cut by more than half.

Figure 2 on the following page shows how the DaVinci platform appears at a high level. The stacks in the center and on the right represent the RISC and DSP software, respectively. The RISC is the master, communicating directly to the DSP's real-time operating system (RTOS), DSP/BIOS™, through the DSP/BIOS Link interprocessor communication software. This low-level communication between the cores enables the developer to work at a high level on the RISC, without having to bother with the DSP or even see into what it is doing.

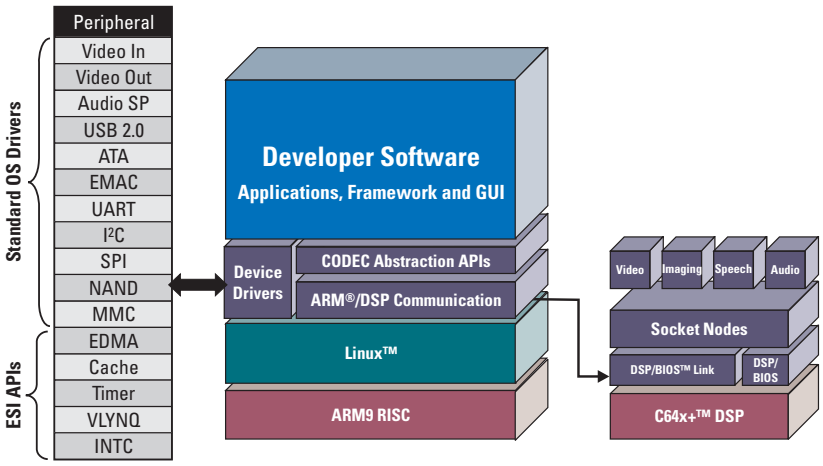

Figure 2. DaVinci™ Software Platform Overview.

The developer's software sits on top of the RISC stack, and the underlying operating system (OS), in this case, is Linux®. The Linux kernel, developed for DaVinci technology by MontaVista, includes optimized drivers for the audio-video and communications peripherals shown on the top left of the figure. One of the strengths of the DaVinci platform is that it can reside on different OSes. A set of easy peripheral software interfaces (EPSI) for drivers that are not OS-specific, as shown in the bottom left of Figure 2, allows for easy porting. EPSI takes advantage of features of the DaVinci platform that are not covered by standard driver interfaces. For example, V4L2 (Video for Linux 2), a standard interface for video capture, does not encompass the previewer, resizer, histogram and other blocks that are provided in the video processing front end of the DM644x processors. Because of the hardware abstraction inherent in the platform, EPSI-enabled drivers will overlap with almost any OS that a developer wishes to port to.

Working with the Software Platform

To the software developer writing application code in C on the RISC, the DSP appears as another resource, similar to the peripherals and memory. Since video systems rely heavily on codecs, the DaVinci platform provides a codec engine that enables straightforward application programmer interfaces (APIs) to algorithms in each of the four processing domains: video, imaging, speech and audio (VISA). For each of these four types of algorithms, the VISA codec engine abstracts the complexities of the signal-processing layer into APIs for encoding and decoding. In other words, the programmer needs to deal with only eight VISA API specifiers: `VIDENC`, `VIDDEC`, `IMGENC`, `IMGDEC`, `SPHENC`, `SPHDEC`, `AUDENC` and `AUDDEC`.

The APIs follow a set framework, consisting of four APIs for each instance or thread of the codec. `xxx_CREATE()` reserves I/O and allocates memory, while `xxx_DELETE()` surrenders I/O and frees memory. ("`xxx`" represents one of the eight specifiers above, such as `VIDDEC_CREATE()`.) `xxx_PROCESS()` either encodes or decodes a frame, while `xxx_CONTROL()` permits changing the algorithm parameters dynamically or querying the status of the codec. All of the APIs serve as wrappers for the DSP's low-level services, or socket nodes, that would be used to write C code in a DSP framework. For example, programmers familiar with the workings of the DSP would recognize the `xxx_create()` APIs equivalent to the following APIs from the DSP/BIOS™ RTOS and TMS320™ DSP Algorithm Standard: `algNumAlloc()`, `algAlloc()`, `MEM_alloc()`, and `algInit()`. Thus, the VISA engine provides low-level control of codecs along with high-level abstraction from the details. In addition, the API framework provides hooks that allow the developer to build additional functionality into codecs for product differentiation.

The algorithms all follow a standard, xDM, that builds on the general TMS320™ DSP XDAIS algorithm standard (XDAIS), with extensions for compliance with the VISA codec engine. The effect is to ensure plug-and-play capability for xDM-approved multimedia codecs, regardless of the vendor or implementation. The codec may run on the DSP to take advantage of high signal-processing performance, or it may need only the resources of the RISC and be executed there. Either way, the APIs look the same to the application programmer. As a result, changing a codec within one of the four VISA types does not require changing APIs, so a system can be designed to exchange, say, MPEG-2 and MPEG-4 decoding without any modification of the application that runs the decoder. For algorithm development, the availability of the xDM standard and the VISA engine software infrastructure means that the DSP programmer does not need to know the end application, making it easier to create codecs that are directly portable from system to system.

API Coding Example

Figure 3 shows a generalized view of the API framework. The user, through the GUI, calls an instance of a codec that proceeds through phases of creation, execution and deletion. As the program executes, the VISA APIs control the codec, which can be running on the DSP, with or without acceleration, or on the RISC, or on both processors. Standard Linux driver interfaces (FBDev/DirectFB, V4L2, OSS/ALSA, etc.) and EPSI APIs facilitate communication with the peripherals for I/O.

The following lines of pseudo-code show the API framework in more detail as the program runs a video decoder. The first two lines of code open a device for reading input and a file for writing output. Line 3 handles what would usually be several lines of control information to fully initialize each I/O device. Line 4 starts the VISA codec engine, and line 5 creates an algorithm named "`viddec`" and passes its parameters. `VIDDEC_create` also reserves I/O and memory for the thread.

```
1.  idevfd = open("/dev/xxx", O_RDONLY);
2.  ofilefd = open("./fname", O_WRONLY);
3.  ioctl(inputfd, CMD, &args);
4.  myCE = Engine_open("vcr", myCEAttrs);
5.  myVE = VIDDEC_create(myCE,"viddec",params);
```

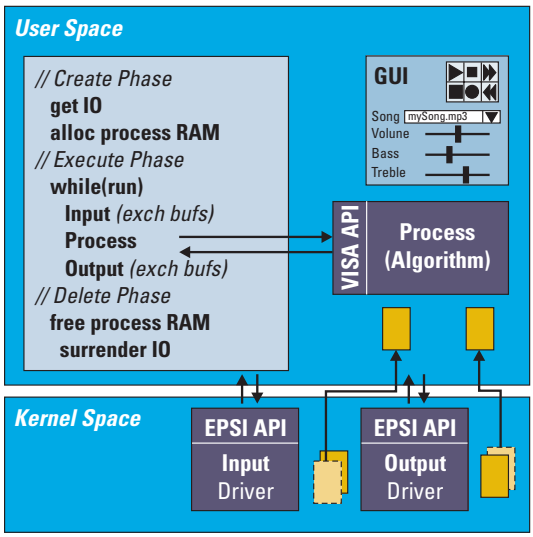

Figure 3. API Framework for DaVinci™ Technology

```
6.  while( doRecordVideo == 1 ) {
7.  read(idevfd, &rd, sizeof(rd));
8.  VIDDEC_control(myVE, …);
9.  VIDDEC_process(myVE, …);
10. write(ofilefd, &wd, sizeof(wd));
11. }

12. close(idevfd);
13. close(ofilefd);
14. VIDDEC_delete(myVE);
15. engine_close(myCE);
```

The thread has now been created, and the while loop beginning in line 6 executes it. First, a new buffer of data is read from the input device in line 7. (For simplicity, multi-threading and double- or triple-buffering of memory have been omitted, though real-world applications would use these techniques.) The `VIDDEC_control` API in line 8 controls the algorithm in a defined manner, and the `VIDDEC_process` API in line 9 runs the video decoder. In line 10, the loop ends by writing the encoded data to the output file. The last four lines delete the thread by closing the input device and output file, releasing I/O and memory with `VIDDEC_delete`, and finally closing the codec engine.

Both the `VIDDEC_control` and `VIDDEC_process` APIs are common to any algorithms implementing video decoding, so the author of this program can use any algorithm of this type without having to rewrite the code. Examples of creation time parameters that these APIs can pass (`iVIDDEC_Params`) include video height and width, maximum frame and bit rates, input and output buffer size, big- versus little-endian data, chroma format and others. Run-time parameters (`iVIDDEC_DynamicParams`) include the input data size, frame ID assignment, display width and frame skip indicator which can be useful for supporting trick play. Status parameters (`iVIDDEC_Status`) are also available.

Development Tools and Support

The complexity of the new video systems means that developers have to rely heavily on SoC suppliers in order to integrate their systems successfully. One significant shortcut comes through using reliable off-the-shelf software whenever possible. Silicon vendors have responded to this need by developing libraries of standard codecs and other algorithms. For instance, DaVinci technology supplies a range of codecs that include:

- Video: MPEG-4, H.261, H.263, H.264/AVC, AVS-M, SVC, WMV9/VC-1, MPEG-2, MJPEG
- Imaging: JPEG
- Speech and Audio: AAC, WMA, MPEG-1/2, MP3, Dolby® AC-3, QCLP, EVRC, SMV, SMVIite, GSM-AMR, G.711, G.726, G.728, G.729, G.723.1, G.722, G.722.1, G.722.1/AMR-WB, iLBC

A large selection of algorithms is also available through third parties, and xDM compliance assures developers that the codecs operate interchangeably on the DaVinci platform. The control mechanisms built into the VISA codec engine make it possible to modify the operation of many algorithms in order to create additional features for the system. For instance, a security system might obtain real-time motion feedback that it can then feed into an object analysis program to help it identify critical events. Proven off-the-shelf software, support for customization, and the ability to create codecs directly—all of these give video OEMs maximum flexibility, allowing them to differentiate their products at the system level, signal-processing level, or both levels.

Developers also have flexibility in choosing the tools they want to work with. Application developers can use MontaVista's GNU suite and other standard tools in developing code for Linux, and TI's Code Composer Studio™ Integrated Development Environment (IDE) is available for programming the DSP as well as the RISC. As the DaVinci platform is ported to other OSes, other sets of tools will become available, too.

For testing and debugging, DaVinci includes Socrates, a tool that provides non-intrusive visibility into different areas of the processor. Like the VISA codec engine, Socrates works through the RISC, providing hooks that the programmer can use to evaluate system loading on all the critical components including the ARM® and DSP. Detailed subsystem analysis can be performed such as tracking the min, max, and average frame processing time for specific functions. Socrates also facilitates analysis of interaction between the ARM and DSP and also between the DSP and video accelerators. DSP/BIOS™ Link analysis allows tracking the data rate and latency of link activities. Critical system insight can be obtained such as EMIF (DDR) bandwidth utilization and contention analysis. In addition, codec developers can employ TI's real-time data exchange (RTDX™) tool to visualize the real-time operation of the DSP. The platform also supports a variety of Linux analysis and hardware profiling tools. With its broad tool support for programming, debugging and testing, DaVinci allows developers to choose the right tools for the way they work and the job they want to do.

Staying Ahead in a Changing Market

A user of audio-video equipment does not really care whether the system is decoding MP3 or AAC or AC-3, as long as the music sounds right. Increasingly, multiple codecs are being used for multimedia systems, especially as these systems integrate multiple applications. Programmable SoC processors can provide flexibility, together with high-level integration that has been tailored for advanced video systems, helping OEMs stay on top of market developments while keeping their high-volume systems cost-efficient.

Multi-core DSP-RISC processors offer the added advantage of maximizing performance by using each core for the tasks it performs best: codecs and other real-time signal processing on the DSP, and system control and user interface on the RISC. Comprehensive technology such as DaVinci designs this hardware foundation to support software platforms that are both powerful and easy to use. The platform makes use of the RISC's programming familiarity to control the DSP's high performance, providing straightforward development and integration at any level, from high-level applications to in-depth codecs. In these SoC solutions, video OEMs have available a processing engine that can support their needs for a wide variety of products, providing the performance, flexibility and ease-of-use that can save development time and manufacturing costs while opening up a world of new application possibilities.

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Interface	interface.ti.com	Digital Control	www.ti.com/digitalcontrol
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
Low Power Wireless	www.ti.com/lpw	Telephony	www.ti.com/telephony
		Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments
Post Office Box 655303 Dallas, Texas 75265