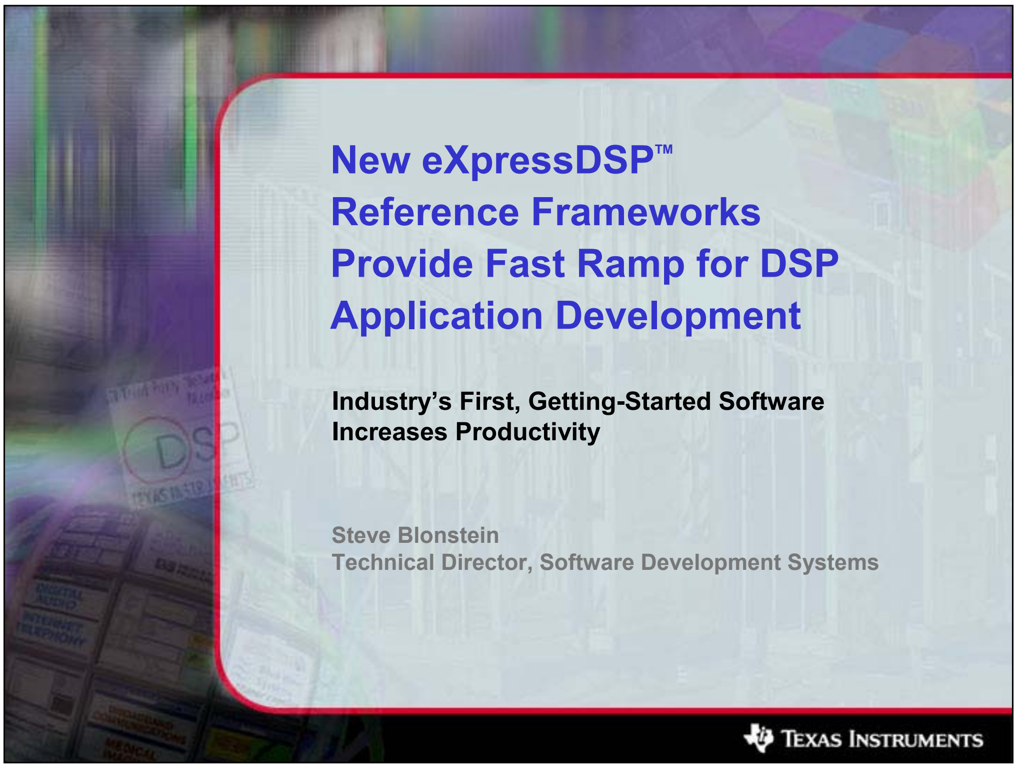


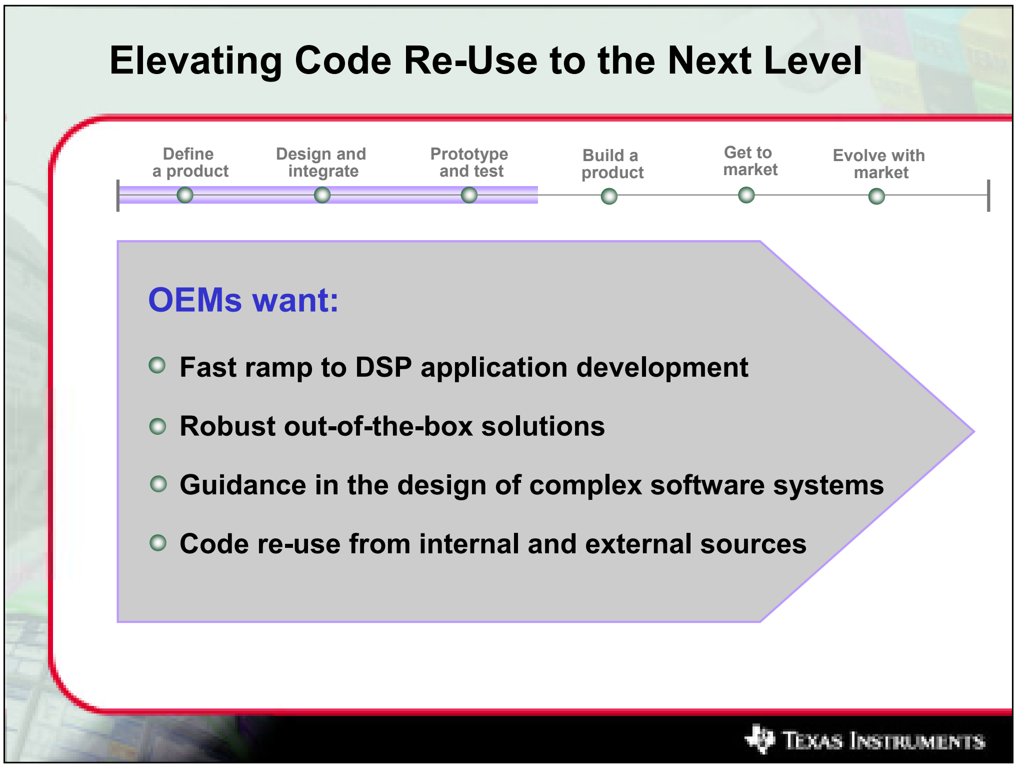
New eXpressDSP™ Reference Frameworks Provide Fast Ramp for DSP Application Development

Industry's First, Getting-Started Software Increases Productivity

Steve Blonstein
Technical Director, Software Development Systems

TEXAS INSTRUMENTS

This is Steve Blonstein, technical director for TI's Software Development Systems. Today, we're going to be talking about a brand new DSP software offering from TI. We call these new offerings eXpressDSP Reference Frameworks or RF for short. These Reference Frameworks are the industry's first, getting started application software, helping to increase overall development productivity.



Elevating Code Re-Use to the Next Level

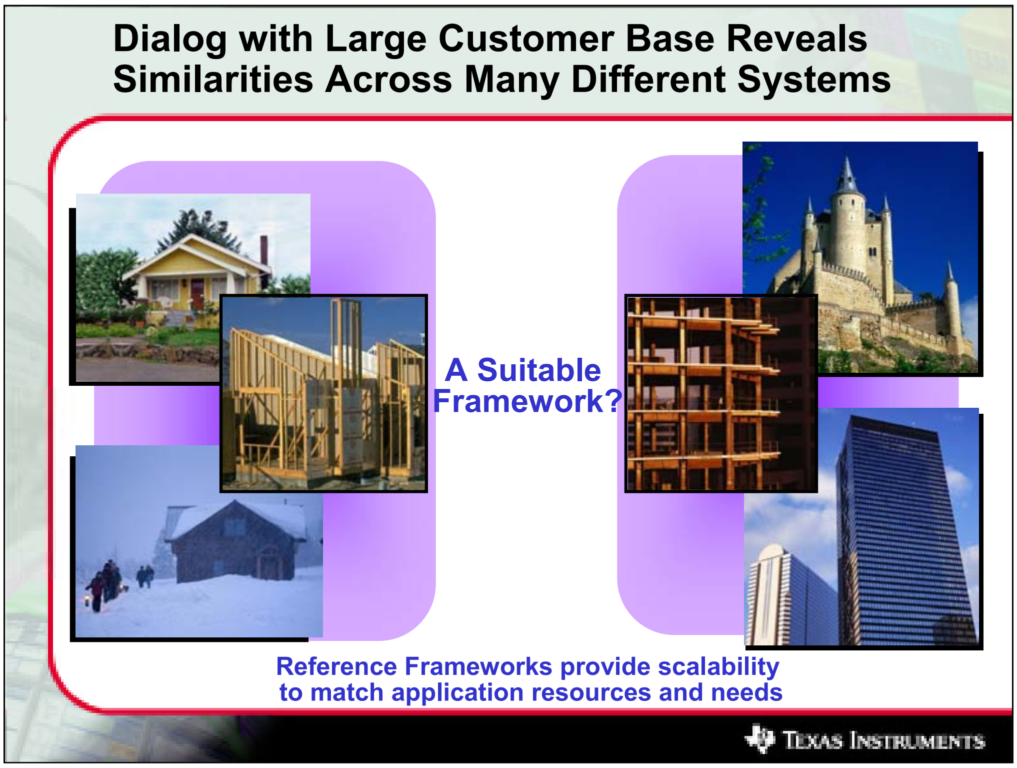
Define a product
Design and integrate
Prototype and test
Build a product
Get to market
Evolve with market
Texas Instruments

OEMs want:

- Fast ramp to DSP application development
- Robust out-of-the-box solutions
- Guidance in the design of complex software systems
- Code re-use from internal and external sources

Code re-use has become one of the biggest issues in the software development arena. As you can see from the top of this slide we're representing the typical product development cycle by this timeline. Reference Frameworks are designed to provide a significant boost to the early stages of this cycle, highlighted here with the purple shading.

In talking with many OEMs, we consistently hear a similar set of requirements when it comes to software application development on DSP platforms. Everyone wants a fast ramp. Many projects face life and death in their ability to quickly show management or investors the approximate capabilities of the proposed system. Anything that can be done to accelerate this early phase is a boon to most OEMs. We also hear the need for robust out of the box software. There are literally hundreds of "demo" type programs out there but the real issue is that they're often buggy, unsupported, not maintained, and don't map well to a final real product. In addition, as systems get more complex, designers are looking for guidance from experts as to how best use the available resources to create an efficient solution. TI's Software Development Systems consists of over 200 software development engineers with over a thousand years of DSP development experience. We're tapping this experience to give customers better guidance on how to best get things done in the TMS320 DSP world. Lastly, it's that code reuse issue. How easy will it be to pick up the code from one project and move it to the next? Much of the answer relates to the level of structure and hardware abstraction that's been employed in the original development. The Reference Frameworks we're about to describe tackle all of these issues head on.



Dialog with Large Customer Base Reveals Similarities Across Many Different Systems

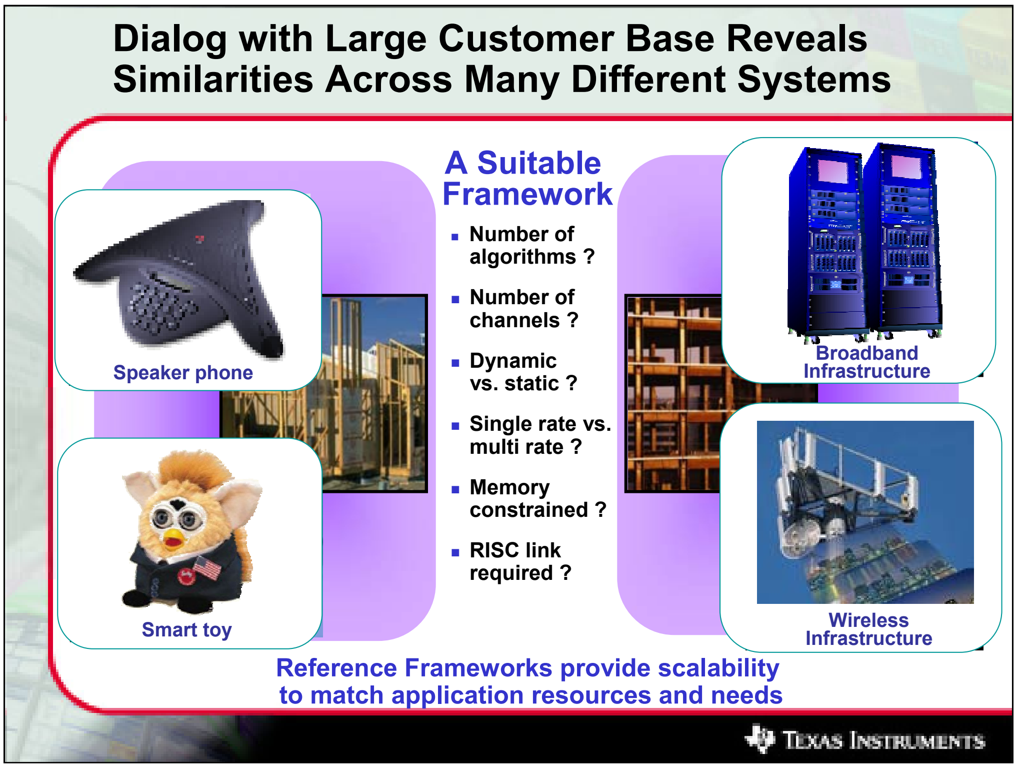
A Suitable Framework?

Reference Frameworks provide scalability to match application resources and needs

TEXAS INSTRUMENTS

Our research has shown that despite the incredible proliferation of DSP usage beyond the traditional telephony and communications application spaces, there are still only a relatively limited number of framework architectures required to get many different systems built. It is this recognition that has allowed us to define and produce the new Reference Frameworks for eXpressDSP Software.

Here we draw a simple analogy to the house building business. Most modern building practices rely on architects designing a suitable house for the customer. Clearly the architect doesn't want to go to the expense of designing a mansion for someone who only needs or can only afford a condominium. The same is true for the customer looking for a castle. They're not going to be happy ending up with a two bedroom apartment. What really enables a functional process is the creation of architectural designs that show the customer, and builder, ahead of time what the final home will look like and how it will be constructed. Clearly the architect will use very different styles and design techniques depending upon the final product and the budget available.



Dialog with Large Customer Base Reveals Similarities Across Many Different Systems

A Suitable Framework

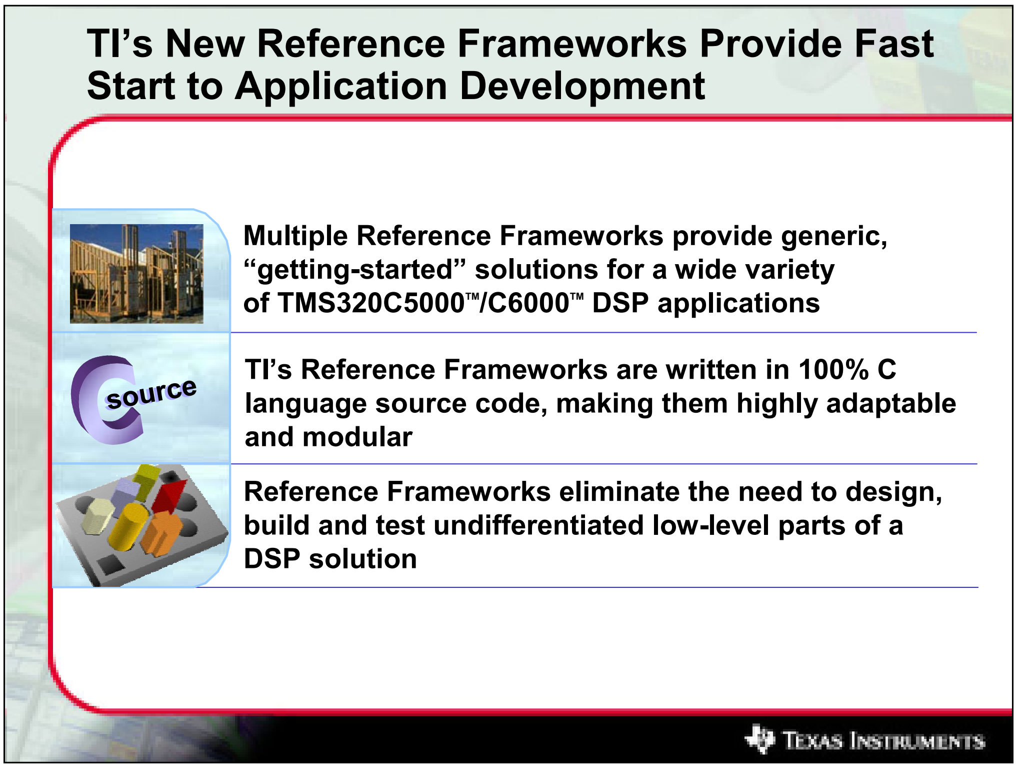
- Number of algorithms ?
- Number of channels ?
- Dynamic vs. static ?
- Single rate vs. multi rate ?
- Memory constrained ?
- RISC link required ?

Speaker phone
Smart toy
Broadband Infrastructure
Wireless Infrastructure
Texas Instruments

Reference Frameworks provide scalability to match application resources and needs

Our research has shown that despite the incredible proliferation of DSP usage beyond the traditional telephony and communications application spaces, there are still only a relatively limited number of framework architectures required to get many different systems built. It is this recognition that has allowed us to define and produce the new Reference Frameworks for eXpressDSP Software.

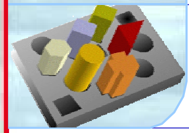
Here we draw a simple analogy to the house building business. Most modern building practices rely on architects designing a suitable house for the customer. Clearly the architect doesn't want to go to the expense of designing a mansion for someone who only needs or can only afford a condominium. The same is true for the customer looking for a castle. They're not going to be happy ending up with a two bedroom apartment. What really enables a functional process is the creation of architectural designs that show the customer, and builder, ahead of time what the final home will look like and how it will be constructed. Clearly the architect will use very different styles and design techniques depending upon the final product and the budget available.



TI's New Reference Frameworks Provide Fast Start to Application Development

Multiple Reference Frameworks provide generic, "getting-started" solutions for a wide variety of TMS320C5000™/C6000™ DSP applications

TI's Reference Frameworks are written in 100% C language source code, making them highly adaptable and modular



Reference Frameworks eliminate the need to design, build and test undifferentiated low-level parts of a DSP solution

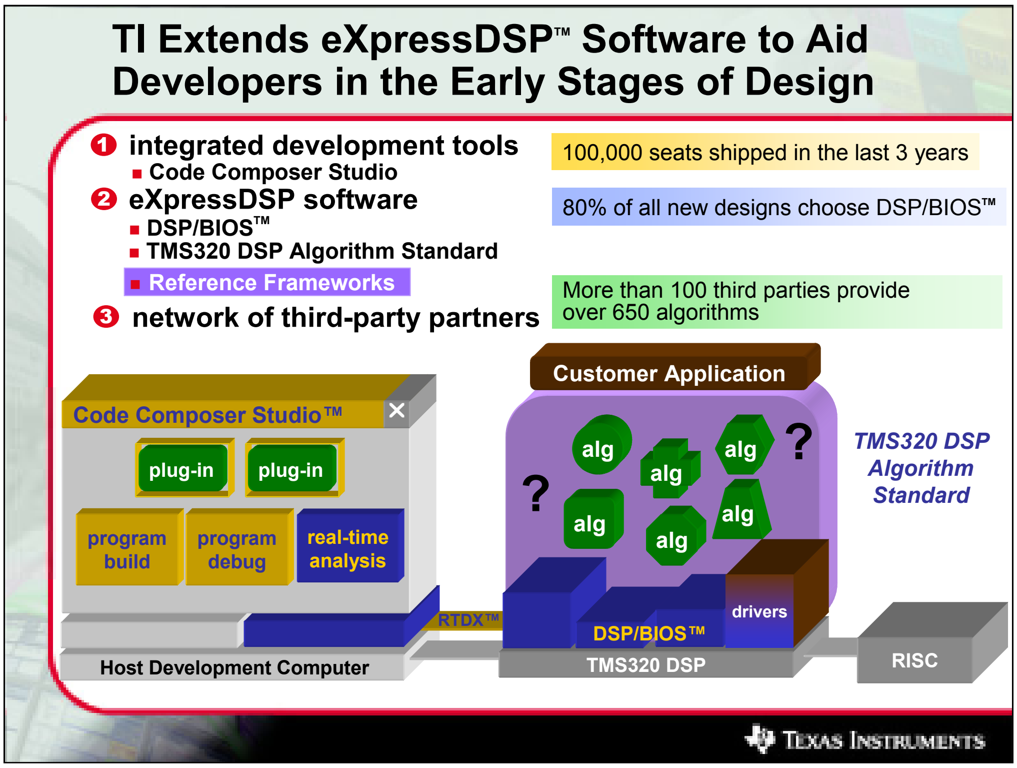
TEXAS INSTRUMENTS

So before we get into the details, let's take a quick look at what's really new with these Reference Frameworks.

1) TI will provide multiple Reference Frameworks. The emphasis of their use and adoption is on the "getting-started" phase of a project. They're designed to be usable in a wide variety of end-applications on both the C5000 and C6000 DSP platforms.

2) The Reference Frameoworks are provided in 100% C-source code. This allows great portability across projects, modularity, enabling simple cut and paste techniques, and finally great adaptability to a wide spectrum of applications.

3) Reference Frameworks are specifically designed to eliminate much of the experimentation and guesswork required in the architecture of a software framework for the product.



TI Extends eXpressDSP™ Software to Aid Developers in the Early Stages of Design

1. integrated development tools
   - Code Composer Studio
2. eXpressDSP software
   - DSP/BIOS™
   - TMS320 DSP Algorithm Standard
   - Reference Frameworks
3. network of third-party partners

100,000 seats shipped in the last 3 years

80% of all new designs choose DSP/BIOS™

More than 100 third parties provide over 650 algorithms

The Reference Frameworks are an extension of our existing eXpressDSP software and development tools strategy. Let's quickly review the other elements of our software initiative, launched in 1999, and how Reference Frameworks enhance the overall offering.

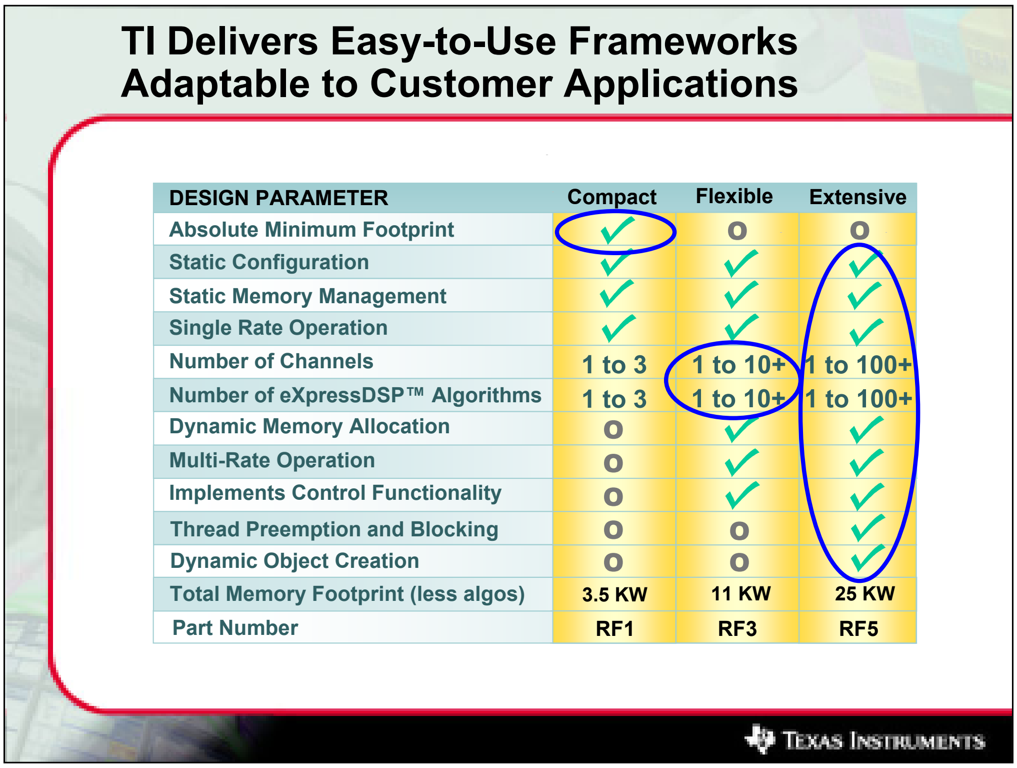
First is a host development environment, Code Composer Studio, shown on the left here in yellow. This is the premier DSP integrated development environment, designed and specifically tuned with the DSP design engineer in mind. All of our development tools find a home under this common umbrella. In addition, as you'll see in a minute, many other people also have the capability to add to this development environment. Since its introduction, about 100,000 copies of Code Composer Studio have been shipped.

Second is eXpressDSP software for the target DSP. First in this area is DSP/BIOS, shown here in blue. This is a small, scalable, and highly robust set of kernel modules designed to run on TI DSPs. Back in the late 1990s we found many customers spending large amounts of valuable time designing and building home grown kernels. Not only is little value added in this area, but it's historically proven difficult to get real robustness, and portability at this level of a system. In response, TI launched DSP/BIOS and licensed it with every TMS320 DSP device. There are no run-time royalties, and the code base has a ten year history of robustness. 80% of our all our customer design starts now opt to use DSP/BIOS for their designs.

Another part of eXpressDSP software is the TMS320 DSP Algorithm Standard also known as XDAIS. Algorithms make up a key part of the code running on the DSP. Up until eXpressDSP there were no standards for the usage and interoperability of algorithms on the DSP. This often resulted in frustration and much wasted time trying to complete the system integration. The standard changes this by clearly defining programming rules and guidelines for all algorithms. This ensures a much better integration experience for the end-user, much improved portability to different types of systems, and allows more accurate apples to apples comparisons of similar algorithms from different sources.

The third element of the software initiative is TI's extensive network of more than 500 third parties. Examples of this are third parties that produce "plug-ins" to the Code Composer Studio environment. A great recent example of this concept is the Matlab plug-in from The Mathworks. For the first time, Matlab users can have their favorite tool integrated directly into Code Composer Studio. In addition, there are over 100 different vendors offering more than 650 software algorithms ready to run on one or more of TI's DSPs. Everyone of these algorithms has been written to comply with the previously mentioned algorithm standard, XDAIS. Third party software is depicted in green throughout this presentation.

Although eXpressDSP software has made a big difference in reducing the need to reinvent the wheel at several different levels of a TMS320 DSP system, there are still significant places where the user is left to experiment or guess the best architectural approach for a particular system. Examples of this include deciding which pieces of the DSP/BIOS kernel are best suited for the particular application, which hardware drivers need to be written and used and the I/O methodology to be chosen for those drivers. Additionally, different strategies can be employed depending upon the long-term need for system flexibility. And this brings us to the purple section of the diagram, what we will now refer to as the Reference Framework.



TI Delivers Easy-to-Use Frameworks Adaptable to Customer Applications

DESIGN PARAMETER	Compact	Flexible	Extensive
Absolute Minimum Footprint	✓	o	o
Static Configuration	✓	✓	✓
Static Memory Management	✓	✓	✓
Single Rate Operation	✓	✓	✓
Number of Channels	1 to 3	1 to 10+	1 to 100+
Number of eXpressDSP™ Algorithms	1 to 3	1 to 10+	1 to 100+
Dynamic Memory Allocation	o	✓	✓
Multi-Rate Operation	o	✓	✓
Implements Control Functionality	o	✓	✓
Thread Preemption and Blocking	o	o	✓
Dynamic Object Creation	o	o	✓
Total Memory Footprint (less algos)	3.5 KW	11 KW	25 KW
Part Number	RF1	RF3	RF5

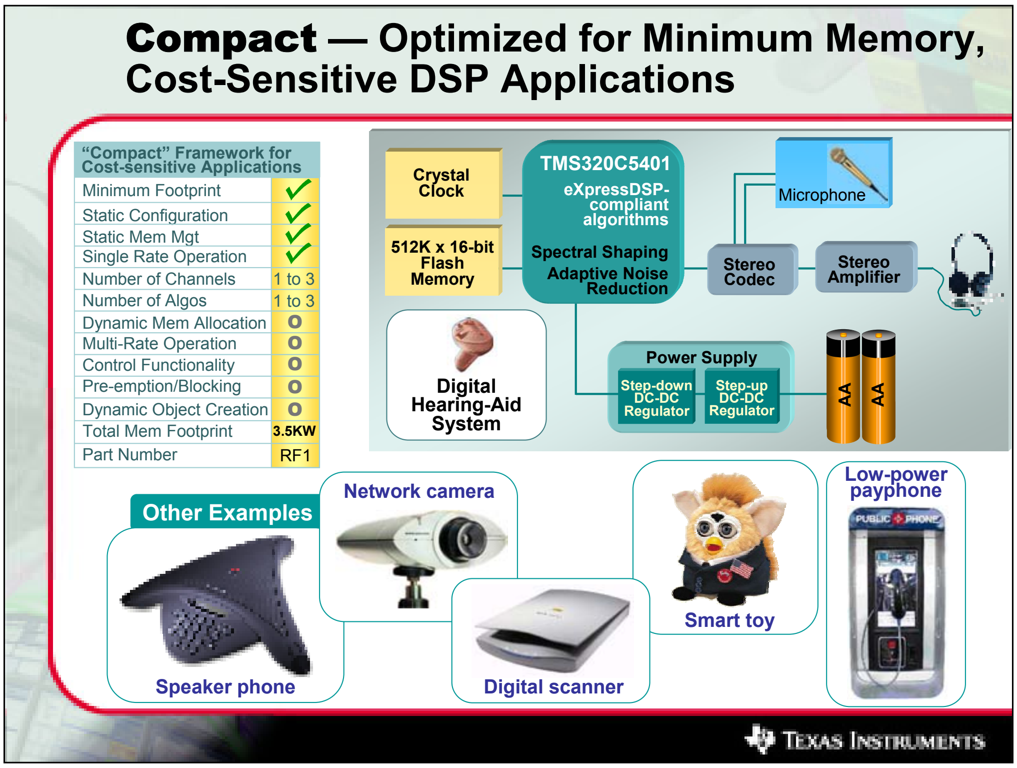
Now we've described the make-up of a reference framework, let's take a look at their characteristics. The table shows the details of the first three Reference Frameworks. There are 12 basic characteristics that each of these designs will display listed down the left hand side. Some characteristics are simple yes/no and others provide working ranges to help the user pick the appropriate level.

Let's walk through these frameworks.

The "Compact" framework, also referred to as RF1, is designed to be the absolute minimum footprint possible and yet still retain all of the elements of eXpressDSP. It is designed for static object configuration and will not support run-time or dynamic object creation. Memory management is also completely static and run-time or dynamic memory management is not supported. The framework has been optimized to support a small number of channels and a small number of algorithms. Some flexibility is given up here but the reward is the absolute minimum footprint. In fact, RF1 can be implemented in about 3.5 KW on a TMS320C54x platform. Another limitation of this level is so-called single rate operation. This means that all algorithms and all channels must run at the same data rate, i.e. x frames of y words every z milliseconds. Thus, by definition, this level cannot support multi-rate operation. There is no thread preemption or blocking in this level. There is no support provided for either a control thread or communication to a general purpose processor. If you think your system can fit into these characteristics and that you are very resource constrained then this level should be the most appropriate. If you need more flexibility, then consider one of the higher level designs.

The "Flexible" framework, also known RF3 is significantly different from RF1. The biggest change is that absolute minimum footprint has been traded for flexibility. Compare the checks and crosses between level 3 and the previously discussed level 1. Even though level 3 still only supports static object creation, there is now support for dynamic memory management i.e. data buffers can be configured and managed at run time. The added flexibility also allows the use of more channels, typically up to 10, and more different algorithms, also typically up to 10. Another big difference is that level 3 supports multi-rate operation. What this means is that different algorithms can run at different rates say one at 10 mS frame rates and another at 20 mS. Finally, this level provides an additional thread specifically to allow control of the DSP by an external entity such as a host processor. As already mentioned, this flexibility comes at a price in terms of footprint. A typical implementation of this level 3 will be about 11 KW on a TMS320C54x DSP generation.

The highest currently defined framework is the "Extensive" framework, also known as RF5. This is for system designers who are looking for extensive flexibility and where system footprint is not the top priority. Like RF3, RF5 supports static object creation, plus both static and dynamic memory management. It will support up to a very large number of algorithms and channels, plus allow both single rate and multi-rate operation. Full thread preemption and blocking are enabled, as is a dedicated control thread. Clearly all of this flexibility comes at a price in terms of system footprint. A typical implementation of this level 5 will be about 28 KW on a TMS320C64x™ DSP generation.



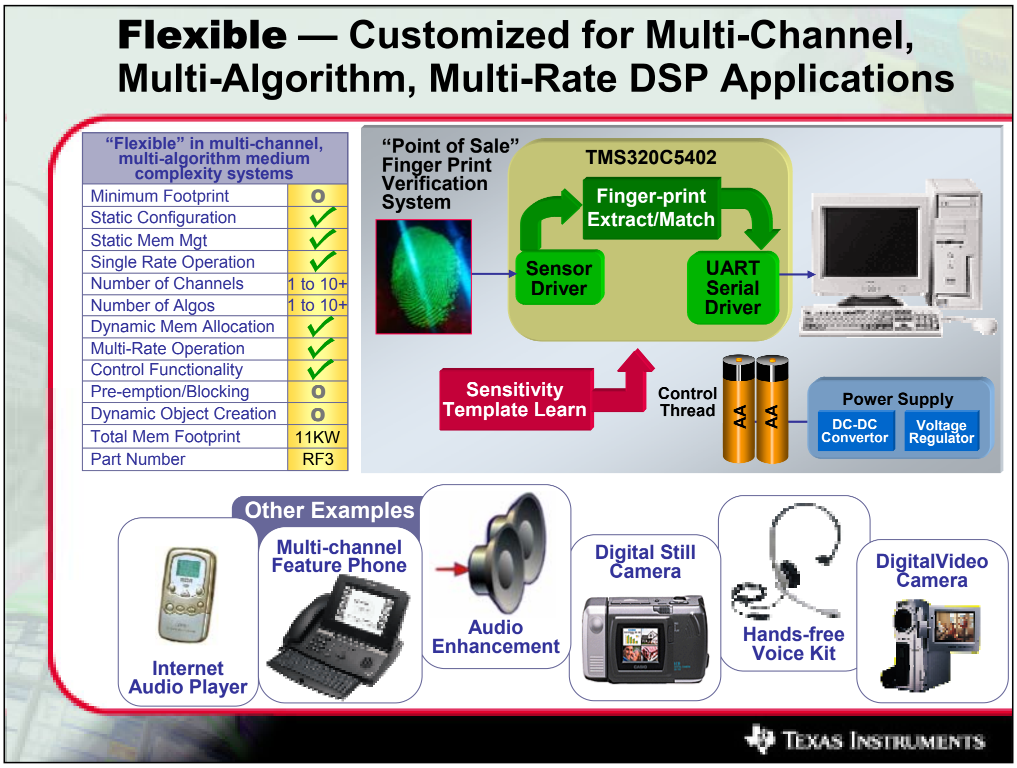
Compact — Optimized for Minimum Memory, Cost-Sensitive DSP Applications

"Compact" Framework for Cost-sensitive Applications	
Minimum Footprint	✓
Static Configuration	✓
Static Mem Mgt	✓
Single Rate Operation	✓
Number of Channels	1 to 3
Number of Algos	1 to 3
Dynamic Mem Allocation	O
Multi-Rate Operation	O
Control Functionality	O
Pre-emption/Blocking	O
Dynamic Object Creation	O
Total Mem Footprint	3.5KW
Part Number	RF1

Now let's review a couple of real product examples that can make use of these Reference Frameworks.

Digital Hearing Aids are a great example of a product that can leverage the power of DSP. Key care-abouts in such a design are keeping both the memory footprint and the power consumption to an absolute minimum. This example was provided by our Digital Hearing Processor group. It mapped very quickly to the Compact framework, RF1. The system runs two algorithms across two channels, one for each ear. The system can be completely statically configured and there is no need for any dynamic configuration capabilities. The entire system including the algorithms comfortably fit into the 8KW internal memory available in the TMS320C5401.

Some other examples are shown that might be ideal candidates for leveraging this Compact framework. These include the speakerphone, a simple network camera, a scanner, toys , and low power payphones.

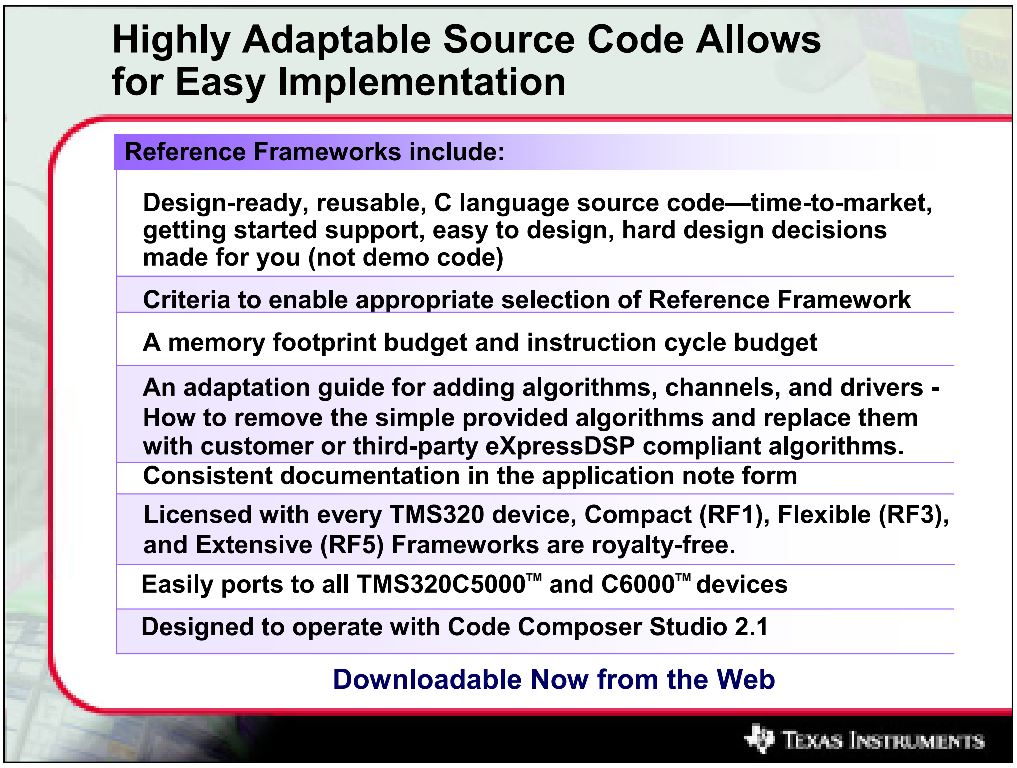


Flexible — Customized for Multi-Channel, Multi-Algorithm, Multi-Rate DSP Applications

"Flexible" in multi-channel, multi-algorithm medium complexity systems	
Minimum Footprint	o
Static Configuration	✓
Static Mem Mgt	✓
Single Rate Operation	✓
Number of Channels	1 to 10+
Number of Algos	1 to 10+
Dynamic Mem Allocation	✓
Multi-Rate Operation	✓
Control Functionality	✓
Pre-emption/Blocking	o
Dynamic Object Creation	o
Total Mem Footprint	11KW
Part Number	RF3

Our biometrics group was able to leverage the Flexible framework, RF3, to quickly build a system that demonstrated a personal security system that checks a fingerprint against a known good print. Again, the entire framework plus the algorithms mapped into the internal 16KW memory of the TMS320C5402. As shown by the table, the design needed more flexibility than offered by the Compact framework, RF1.

Some other examples that are a good fit for this framework are an internet audio player, a multi-channel feature phone, a digital still camera, and a hands-free voice kit.



Highly Adaptable Source Code Allows for Easy Implementation

Reference Frameworks include:

- Design-ready, reusable, C language source code—time-to-market, getting started support, easy to design, hard design decisions made for you (not demo code)
- Criteria to enable appropriate selection of Reference Framework
- A memory footprint budget and instruction cycle budget
- An adaptation guide for adding algorithms, channels, and drivers - How to remove the simple provided algorithms and replace them with customer or third-party eXpressDSP compliant algorithms.
- Consistent documentation in the application note form
- Licensed with every TMS320 device, Compact (RF1), Flexible (RF3), and Extensive (RF5) Frameworks are royalty-free.
- Easily ports to all TMS320C5000™ and C6000™ devices
- Designed to operate with Code Composer Studio 2.1

Downloadable Now from the Web

TEXAS INSTRUMENTS

So let's take a closer look at exactly what is involved in eXpressDSP Reference Frameworks.

First, the frameworks are robust, design-ready C source code. They are very portable to different platforms including your own hardware. The code is reusable from project to project and it's easy to cut and paste just the pieces that serve your needs or interests. This is not demo code. It has been extensively tested, will be fully supported and maintained by TI, and updated as necessary.

Second, clear criteria are provided to lead the user through the selection process for which reference framework is the most appropriate for their application.

Next, we provide accurate budgets for both memory footprint and MIPs usage for each of reference frameworks . This way, the user knows exactly up front what space, MIPs are left to run the "real" algorithms in the system.

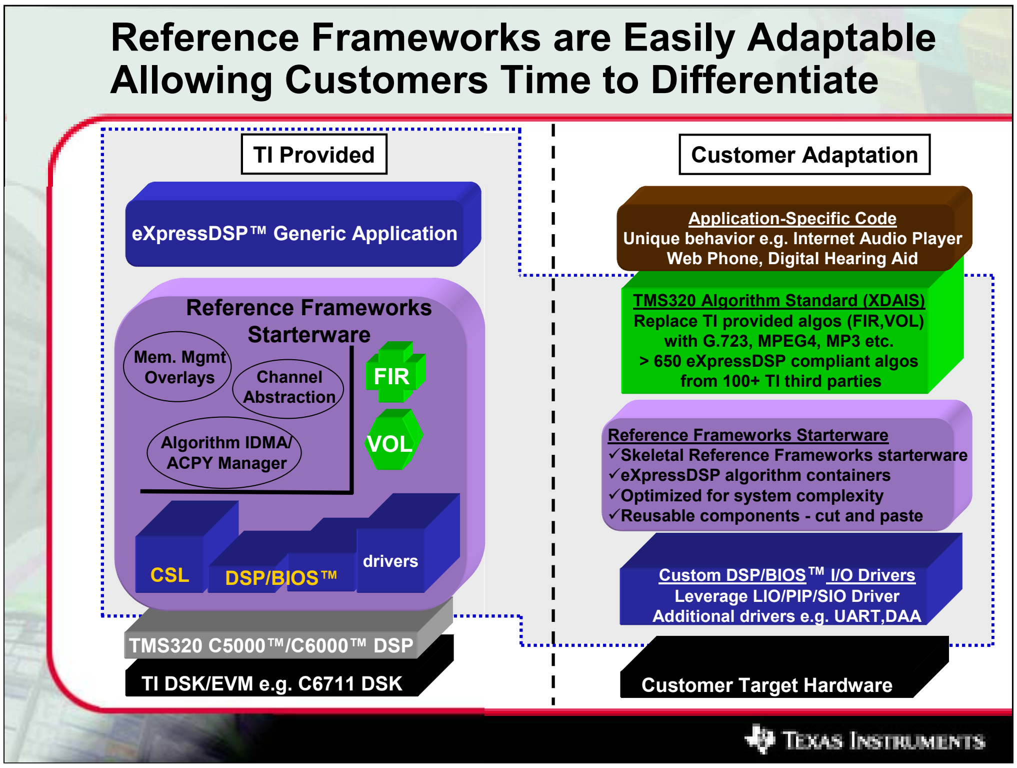
We also provide detailed adaptation guides. Examples of this include how to replace the supplied generic algorithms with more complex algorithms such as voice coders, modems, MPEG, JPEG, speech algorithms and so on. In addition, we describe how to add additional channels to a system and also how to add one's own drivers for special purpose hardware.

Each Reference Framework is documented in the form of a standard TI application note identified by the four letters SPRA. To make it even easier to identify which application note refers to which level of design, we have assigned matching numbers i.e. the level 1 design application note number is SPRA791, level 3 SPRA793 an so on. Just as important, we've maintained a consistent look and feel across the application notes so that it's easy to jump from one to another and feel comfortable with the format.

The Compact, Flexible, and Extensive Frameworks are provided with a royalty-free run-time license for use on any TMS320 DSP device.

All Reference Frameworks are designed to be portable to both the C5000 and C6000 TMS platforms.

Finally, Reference Frameworks are designed to work with the latest version of Code Composer Studio, version 2.1.



Reference Frameworks are Easily Adaptable Allowing Customers Time to Differentiate

TI Provided

eXpressDSP™ Generic Application

Reference Frameworks Starterware

Mem. Mgmt Overlays

Channel Abstraction

Algorithm IDMA/ ACPY Manager

FIR

VOL

CSL

DSP/BIOS™

drivers

TMS320 C5000™/C6000™ DSP

TI DSK/EVM e.g. C6711 DSK

Customer Adaptation

Application-Specific Code
Unique behavior e.g. Internet Audio Player
Web Phone, Digital Hearing Aid

TMS320 Algorithm Standard (XDAIS)
Replace TI provided algos (FIR,VOL)
with G.723, MPEG4, MP3 etc.
> 650 eXpressDSP compliant algos
from 100+ TI third parties

Reference Frameworks Starterware
✓Skeletal Reference Frameworks starterware
✓eXpressDSP algorithm containers
✓Optimized for system complexity
✓Reusable components - cut and paste

Custom DSP/BIOS™ I/O Drivers
Leverage LIO/PIP/SIO Driver
Additional drivers e.g. UART,DAA

Customer Target Hardware

Texas Instruments

Now let's take a look at the process of taking a base reference framework from TI and adapting it to your unique environment. First, let's take a look at what is provided by TI.

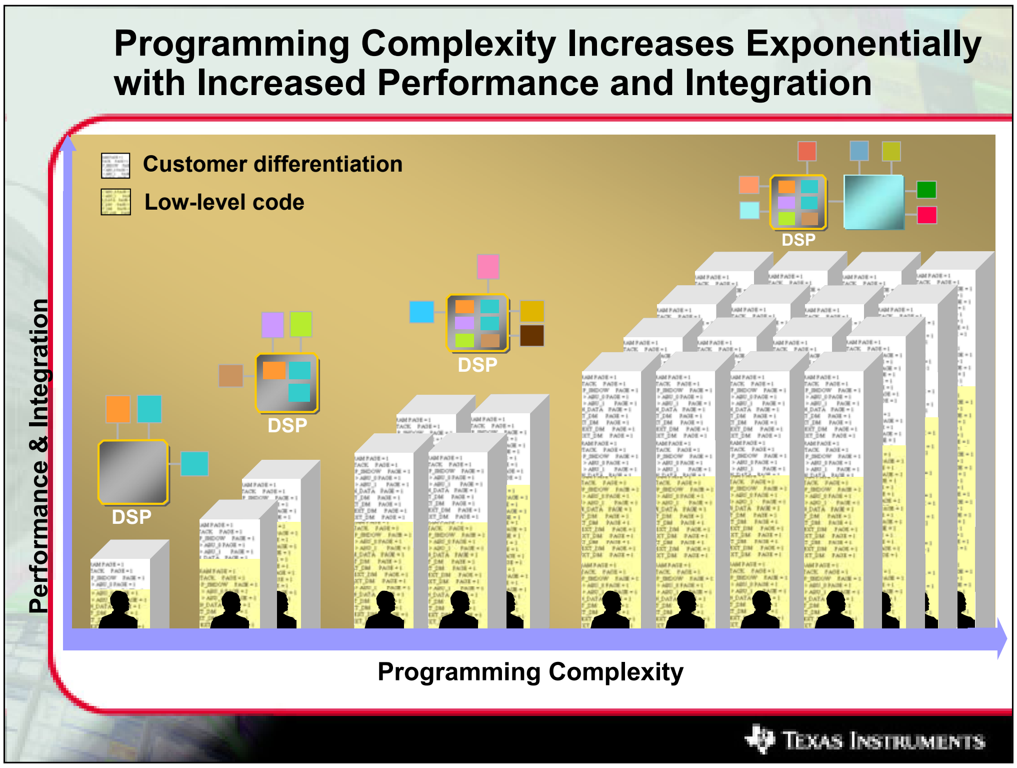
At the hardware level, our reference frameworks are illustrated running on some of our most common development platforms like the DSP Starter Kits, DSKs, or Evaluation Modules, EVMs. There is typically at least one of these for each family of TI DSPs. Next comes the actual starter-ware software that you will craft your final application from. This contains several discrete elements. At the lowest level are the required DSP/BIOS and Chip Support Library or CSL components needed for the level of design in question. Clearly, RF1 contains less modules than RF3 and RF3 requires less modules than RF5. In addition to these base modules are drivers specifically designed to move data on and off the starter kit board or evaluation module. The framework also contains the code that enables memory overlay schemes, a DMA manager for algorithms requiring DMA, and channel abstraction. The framework is also fitted with simple algorithms like FIR filters and volume control as a way to illustrate how and where algorithms fit into the final system. These algorithms can be easily replaced by the algorithms that are appropriate for your system. Finally, there needs to be a controlling application that, in this generic case, is fairly basic.

So what exactly needs to be done to modify this reference framework to be usable in your final application. At the lowest level, the hardware, the DSK or EVM is likely to be replaced by your own target hardware. Even though the DSP itself may be the same, the surrounding hardware peripherals are likely to be different. This is where we've made it possible to leverage the existing driver model, and also to add drivers of your own while still maintaining the same overall system look and feel.

The main part of the framework is that part of the code where the user has selected the most appropriate starting point to begin modifications. As already mentioned, it contains algorithm containers designed to house any eXpressDSP compliant algorithm, it has been optimized for system complexity, and it is made of reusable source code modules that can be easily cut and pasted as needed for the final application.

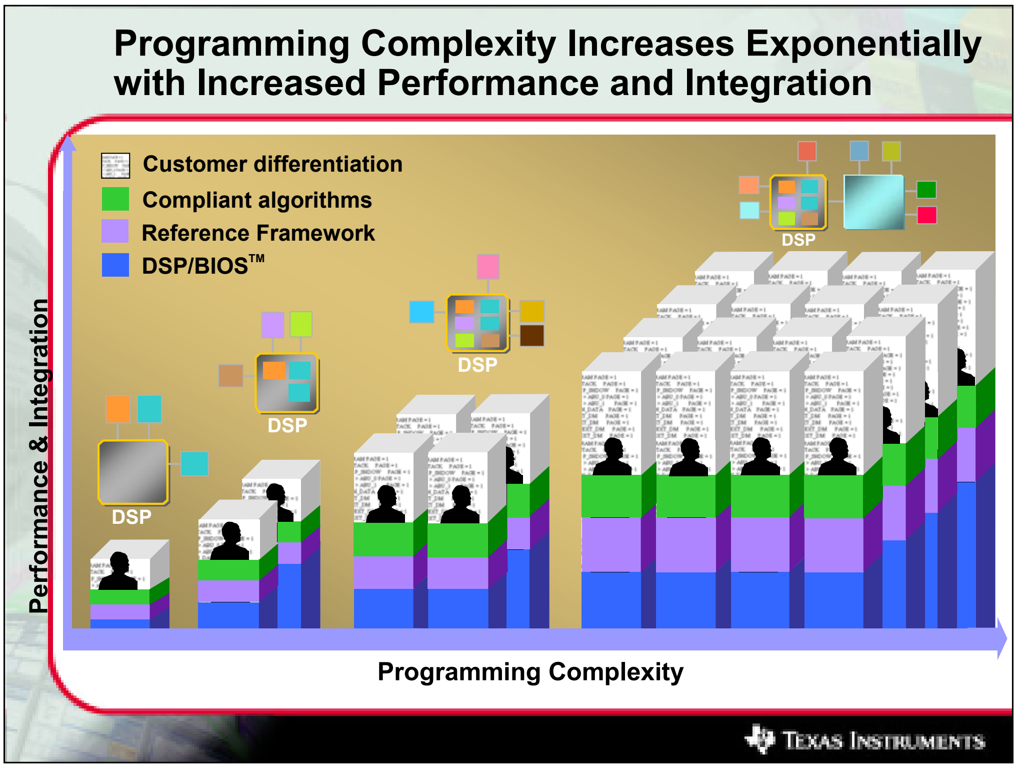
Next, the simple FIR and VOL algorithms can be exchanged for the actual system algorithms whether they be voice codecs, echo cancellers, music codecs, modems, fax, telephony, imaging, video and so on. Any eXpressDSP compliant algorithm will quickly and simply integrate into this environment. Over 650 such algorithms are available from more than 100 TI DSP third parties, or you can develop your own to install into this environment.

Finally, it will be necessary to write the application specific code to control the final system. Clearly, the application/user level requirements for say an MP3 player are quite different from those of a digital hearing aid.



Programming Complexity Increases Exponentially with Increased Performance and Integration

This graphic illustrates that whether you are building a small system as shown on the left or very complicated system as shown on the right, the system really breaks down into two fundamental types of code. At the bottom of the work is usually a lot of low level code. In most cases there is little reward or competitive advantage to recreating this code.



Programming Complexity Increases Exponentially with Increased Performance and Integration

This is where eXpressDSP software and the Reference Frameworks come into play. Leverage the DSP/BIOS kernel, an appropriate Reference Framework, and any of the eXpressDSP compliant algorithms from the TI third parties. By doing this a large chunk of the "messy" low level code can be almost immediately operational, leaving you to work on the value-added parts of the system.

Download and Implement Tested Source Code in Your Project Today

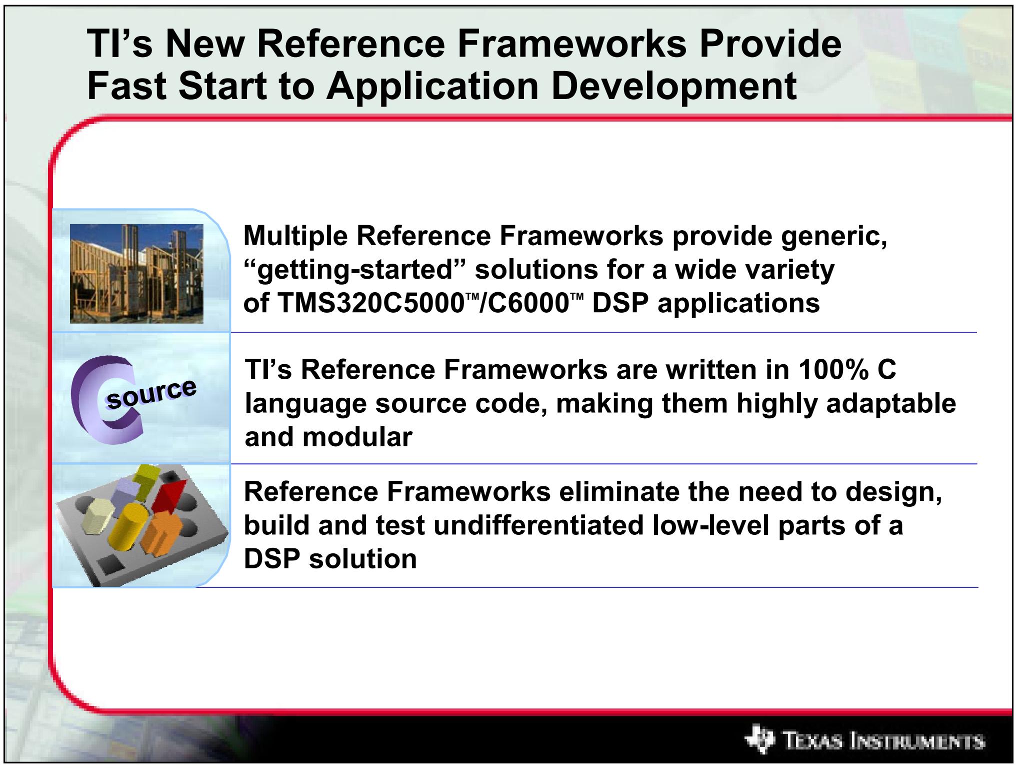
1. Visit the Reference Framework site: www.dspvillage.ti.com/rframeworks
2. Use specified selection criteria to determine which Reference Framework fits your application
3. Download appropriate application notes and source code
4. Start immediate implementation on the TI hardware platform of choice using Code Composer Studio v2.1

Texas Instruments

So how can you get going with Reference Frameworks?

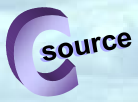
First, visit the Reference Framework web site listed here.

Study the provided selection criteria to help choose the right starting point. Download the most appropriate application notes and the source code from the web site. Start immediate implementation and experimentation on the TI development platform of your choice. Make sure you have upgraded to at least Code Composer Studio version 2.1 to properly leverage the Reference Frameworks.

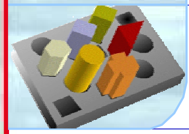


TI's New Reference Frameworks Provide Fast Start to Application Development

Multiple Reference Frameworks provide generic, "getting-started" solutions for a wide variety of TMS320C5000™/C6000™ DSP applications



TI's Reference Frameworks are written in 100% C language source code, making them highly adaptable and modular



Reference Frameworks eliminate the need to design, build and test undifferentiated low-level parts of a DSP solution

Texas Instruments

So let's summarize what we've discussed in the presentation. Our new eXpressDSP software reference frameworks provide many new benefits:

1) Ready to use application code designed for a multitude of applications. We think that virtually every application can leverage some or all of one of these base designs. This is the answer to "so how do I get started?".

2) Reference Frameworks are 100% C-source code, making them highly adaptable. Either take everything or cut and paste the appropriate parts into your final application. There are no restrictions or limitations.

3) Reference Frameworks eliminate the need to design, implement, and test undifferentiated low-level parts of your DSP solution. This will provide a competitive advantage over those who choose to reinvent the wheel.

I'd like to thank you for taking the time to listen to this presentation today. Again, if you need any more information visit our web site at www.dspvillage.com.