ANALOG DEVICES

Blackfin **O**nline **L**earning & **D**evelopment

Presentation Title: Lockbox™ Secure Technology on Blackfin Processors

Presenter: Phil Giordano, Senior Applications Engineer

Chapter 1: Introduction

Subchapter 1a: Agenda

Hello and welcome. My name is Phil Giordano. I'm an Applications Engineer with Analog Devices and today I'm going to be talking to you about Lockbox™ Secure Technology on Blackfin processors.

Before viewing this module, it's recommended that you are somewhat familiar with security and cryptography concepts as well as have some experience with embedded processors and/or systems.

We're going to talk today about defining your security needs, a highlight of Lockbox technology on Blackfin processors and then we'll get into some in-depth detail on security features present on our Blackfin devices that include Lockbox secure technology.

We'll go over one-time programmable memory, we'll give you an authentication process that's performed by these Blackfin devices. We'll review the debug and test features and then we'll wrap up with a summary and provide you with a list of resources and references.

Subchapter 1b: Defining Security Needs

So, the first thing we need to ask ourselves as developers of secure embedded systems are what are our security needs and requirements.

So, exactly, what is it we are trying to protect, we need to define who our adversaries are, who is likely to be attacking our products? What level are these adversaries? Are they typically consumers, are they users, or are they your competitors? What resources are available to these adversaries? Are they a typical consumer with not very many resources available to them, not very much knowledge in approaching embedded systems? Are they your competitors, who might be well funded organizations for example? Are they extremely well funded and have vast resources at their disposal such as foreign governments or universities for instance. We also have to understand where the threat of attack is rooted. Is it that someone is trying to steal your code or data or are they trying to reverse engineer your product? Before we can actually evaluate a solution, it's very important that we understand first, what our needs are and what our requirements are. In order to define the problem, then we can evaluate the solution.

Subchapter 1c: Lockbox Highlights

Some of the highlights of Lockbox secure technology on our Blackfin processors are presented on this slide.

Lockbox , incorporates a secure hardware platform. It helps to ensure confidentiality and integrity for both your code and your data. It also helps to ensure authenticity in that the source of your code and data is truly the entity you expect it to be. The secure platform provides you with a secure mode of execution, a secure storage for on-chip keys and data parameters in hardware. There is an on-chip secure ROM that houses our firmware and there's also ways to protect the on-chip RAM or memory on the Blackfin device.

Access to the code and data in the secure domain is monitored by hardware. This prevents any unauthorized accesses to the secure domain. The on-chip ROM is secure in such a way that it prevents unauthorized tampering with the code that resides in that ROM. So this helps to establish a root of trust for the device.

Protection for the on-chip memory or RAM provides you with integrity, guaranteeing integrity for your code and data. It also helps to ensure confidentiality for your trusted code and data. The user defined cipher keys, ID's, and other data parameters that you wish to put on this part, can be securely stored in an on-chip one-time programmable memory array.

In every single processor that leaves the Analog Devices factory incorporates a unique chip ID so if you, for instance, bought 10,000 Blackfin devices with Lockbox technology on them, you would have 10,000 unique chip ID's, uniquely identifying each and every one of those processors. You'll see shortly just how useful that can be in your applications.

Subchapter 1d: Lockbox Benefits

Some of the benefits of Lockbox secure technology are as follows:

We help to ensure authenticity or origin verification for your code and data. This basically helps you to ensure that say a firmware update that's provided by your factory onto the device is actually originating from you and not from some malicious user who might be trying to substitute their own firmware in place of yours.

Lockbox also helps to ensure integrity for your code and data. Developers can use a digital signature authentication process to help ensure that your application code and your data content of the storage media, for instance, your boot device, has not been altered in any way once it's been transferred onto the Blackfin device for code execution or for utilization in your application. Integrity is verified by Lockbox's authentication of digital signatures.

Lockbox also helps you to ensure confidentiality for your code and data. Cryptographic encryption or decryption supports situations that require the ability to prevent unauthorized users from seeing or using your code or data. Lockbox's secure processing environment basically helps you to ensure this confidentiality.

Another benefit of Lockbox is to help ensure renewability in your system. Renewability basically refers to updating your system components to enhance your security. Lockbox's unique chip ID helps to enable end users to identify each and every Blackfin processor and hence the device that it resides within. This Lockbox feature can be used in support of revocation or renewability of licenses in case of security violations, for instance in digital rights management systems. The unique chip ID in combination with a trusted DRM agent, which is provided by the developer or a source OEM, will help to enable developers to implement renewability in DRM systems. The unique chip ID also allows developers to uniquely identify each and every device. This can help you to blacklist a device or remove it entirely from your system if for some reason that device has its security compromised or the system's security compromised. The unique chip ID also has another feature in that it can allow the developer to basically bind a particular Blackfin device to a boot source. This can help prevent cloning of your systems.

Chapter 2: Blackfin Security Features
Subchapter 2a: Features Overview

Now I'll talk about some of the security features on Blackfin devices. There's a one-time programmable memory array on each and every Blackfin that is Lockbox enabled. This is an array of nonvolatile memory. It's 64K bits in size; it's a serial array so its one-bit wide. It's partitioned up into a public area and a private area. The private area is only given privileged access. The public area is open in all modes of operation, for all intents and purposes, it's an unsecured area.

You can use this public area to store information that can be utilized during the authentication process. You can use it to store ID values and other data parameters that are not necessarily required to be confidential.

The private area of OTP memory can be utilized to maintain secret information. Information that is confidential that you don't want anyone except an authorized user to have access to in terms of visibility or read access to it. This can be used, for instance, to hold cipher keys for decryption and encryption processes.

As I mentioned before there is a unique chip ID that's in every single Blackfin device before it leaves the factory. This can be used to prevent cloning of products, to bind the Blackfin device to a particular boot source, and uniquely identify each and every Blackfin. This unique chip ID is stored in the public area of OTP memory.

There is also a secure ROM array on every Blackfin that is Lockbox enabled. We use this to store firmware that is actually used to perform the authentication process. We'll talk about that in some more detail in upcoming slides.

Lockbox enabled devices also incorporate a secure state machine. The secure state machine flow progresses through several different modes of operation. An Open Mode which is basically unsecured is the default mode of the processor upon power-up and out of reset. There is also a Secure Entry Mode. This is the mode in which authentication of digital signatures is performed within and it's a privileged mode of access in which memory protection is in place. There is also a Secure Mode. This is basically the secure environment in which you, the developer, can execute sensitive code and access secret information and data. You also have access to the private area of OTP memory when operating within the secure mode.

We also provide a hardware monitor that ensures firmware execution. So, in other words, we make sure that we maintain control over the program sequencer within the Blackfin and prevent any kind of unauthorized or unexpected branching during firmware execution. We also provide you, the developer, with complete control over the secure processing environment in the form of software control access to

some registers which we call the secure system switches. This basically gives you control over all the memory protection and all the other protection mechanisms that are involved in creating the secure processing environment.

Subchapter 2b: Security Scheme

The first bullet on slide 10 concisely describes the security scheme. If you take away nothing else from this presentation, understand this first bullet. The security scheme is based upon the concept of authentication of digital signatures using standards based algorithms and provides a secure processing environment in which to execute code and protect assets. That sums up Lockbox in a statement.

The security features in all of our Lockbox enabled products are completely optional. The developer never has to even utilize security if you don't wish to use it in your application. The security features don't have to, in any way, complicate your programming model, they don't get in the way of your booting process, they never have to be used whatsoever.

You can optionally utilize security features by personalizing your Blackfin device. The way that you personalize the device is to store a public key into the public area of OTP memory, the one-time programmable memory array.

We use elliptic curve cryptography and typically you generate a public key and a private key. The private key you store securely off-chip, it's never seen by anybody else except you, the developer, or the entity that wishes to digitally sign a message. The public key is stored on the Blackfin processor in public OTP memory. Once the public key is stored there you are now free to utilize all the security features within the Blackfin.

Subchapter 2c: Standards-Based Algorithms

Lockbox technology uses standards based cryptographic algorithms. We felt very strongly about using standards based algorithms as opposed to proprietary algorithms for many reasons. The standards based algorithms have been available in public domain for a number of years. They've withstood attacks by both legitimate cryptoanalysis and by hackers. We utilize elliptic curve cryptography as the asymmetric, or

public key cipher, and also SHA-1, as the secure one-way hash. These are utilized to generate your digital signatures and are also used in the digital signature validation process, or authentication process, that is carried out on the Blackfin processor. We implement ECDSA elliptic curve digital signature algorithm, signature verification, on our BF54x Blackfin family of products and BF52x family of products. These will be used and cited throughout this presentation as the examples for Lockbox enabled products.

Subchapter 2d: Secure State Machine

Now we'll talk a little bit about the secure state machine on Blackfin processors. As I mentioned before there are three modes of operation in the secure state machine.

There is an Open Mode which is the default mode that the processor goes into upon reset out of the boot process. All the secured system switches are deactivated in this mode. The chip is, for all intents and purposes, open, no access restrictions are in place with the exception of one restriction. The private area of OTP memory in which you can store secret or confidential data information is not accessible to anyone in Open Mode.

The next mode that the secure state machine flows through is the Secure Entry Mode. This is the mode in which authentication is carried out. This is a privileged mode of access. There are memory restrictions and protection features in place. In fact, all the secured systems switches are activated in Secure Entry Mode. In this mode the firmware is executing directly out of the on-chip ROM and is executing code that carries out the digital signature authentication process on the Blackfin.

And finally, Secure Mode is the last and the most secure mode of operation on the Blackfin device. Once the authentication process results in success, the digitally signed messages are verified to be trusted. The device will then go into Secure Mode. In this mode of operation, the developer's trusted code is now allowed to execute directly on the processor and you now have access to all of the protection mechanisms. You can control the secure processing environment. You now have access to the private area of OTP memory where you may have stored secrets, and you have control over the secured systems switches which basically govern what protection mechanisms are in place on the processor.

Subchapter 2e: Secure System Switches

These secured systems switches are basically a number of bits within registers that allow you to enable or disable various protection mechanisms. So it is software control over the hardware protection mechanisms in Secure Mode and also in Secure Entry Mode; the firmware has access as well to some degree.

The secured systems switches allow you to disable all avenues of attack in support of secure processing environment. They allow you to protect on-chip memory, for instance L1 instruction memory, L1 data memory and on-chip L2 memory in the case of the Blackfin BF54x products that contain L2 memory. You can protect this on-chip memory from unauthorized direct memory accesses or DMA accesses.

The secure system switches also allow you to disable JTAG emulation instructions. By default, JTAG emulation is disabled upon entry into Secure Entry Mode and into Secure Mode.

Subchapter 2f: Secure State Machine Diagram

This is a state diagram depicting the secure state machine. As you can see here, upon power up and reset we entry into Open Mode by default. The arrows show you the state flow, from Open Mode you progress into Secure Entry Mode. This is where the authentication process is performed. If for any reason, this authentication process ends in failure, you'll be exited out of Secure Entry and returned back to Open Mode. There is no way for you to progress on through to Secure Mode. Assuming the Secure Entry Mode and the authentication process that's executing within that mode results in success, the secure state machine will advance to Secure Mode. Once in Secure Mode, you now have the secure processing environment in place and you can allow your trusted code to execute directly on the Blackfin processor. Upon exit from Secure Mode, you return back into Open Mode in which there are no restrictions in place whatsoever except for the access restrictions to the private area of OTP memory.

Chapter 3: One-Time-Programmable (OTP) Memory
Subchapter 3a: OTP Memory

Now lets talk about the on-chip OTP memory, or One- Time Programmable memory. As I mentioned before, this is an array of memory that's on-chip, it's non-volatile memory. So any information that you

place within the on-chip OTP memory, will be persistent throughout power cycling and throughout reset on the processor.

This array is divided up just about equally into two areas, a public area and a private area. The public area is accessible at all modes of operation as we saw before in the secure state machine Open Mode, Secure Entry Mode and Secure Mode.

All modes have access to the public area. This is typically used to store non-secure information. Information that does not need to be maintained as confidential. This information could typically take the form of your public key cipher, for instance, that's used for the authentication process, developer ID's, product ID's, the Unique Chip ID that is on the processor is also placed in this area of memory.

The private area of OTP memory basically holds any information that you want to keep secret or confidential. Typically, you would use this to store private keys that could be used for decryption of data or for other validation purposes. This memory array is completely programmable; readable, writable, by you the developer. It does not require Analog Devices to preprogram it before it leaves our factory. So once you obtain your Blackfin device, you are free to personalize it and store any information that you wish within this memory array. This is very helpful in security applications because now the developer does not have to share your key material with other untrusted parties. You generate your keys, you store them on the Blackfin yourself. You don't need to share them with Analog Devices or anyone else.

We also provide you with the ability to write- protect the one-time programmable memory after programming.

Subchapter 3b: OTP Memory Map

This is a memory map that depicts the layout of one-time programmable memory. The example we're using here is on the Blackfin BF54X processors. I believe this is identical on the BF52X processors that way as well. As you can see, there are a number of areas that are utilized within this array for page protection. There are page protection bits.

The array is organized as an array of pages. Each page is 128 bits in size, there are 512 pages totaling 64K bits in the array. There are page protection bits, and a Unique Chip ID.

There are some areas that are also reserved for access. Analog Devices utilizes some of this on-chip memory to program certain things such as trim values for the analog peripherals and Unique Chip ID is preprogrammed. There is also error correction that's implemented to maintain the integrity on this array. There are some pages set aside specifically for the error correction codes themselves.

There is an area that is unsecured general purpose storage in both the public and private area of the array.

This area over here basically describes who has access to this, Analog Devices specifically or the developer or the combination of both.

The array is divided up just about evenly between the public area and private area. 64K bits is basically 8 kilobytes so 4K bytes of public memory and 4K bytes of private memory.

As I mentioned before, we provide the ability to utilize error correction to ensure data integrity whenever making write and read access to this memory array. We utilize a very simple Hamming code. It's a single error correction, double error detection. For every 64 bit access that you make using error correction there is a corresponding 8 bit error correction code. As you saw in the previous slide, there is an error correction code that resides in pages on the device in both the public area and the private area.

The Unique Chip-ID's preprogrammed before leaving the factory. The customer key, the public cipher key that you utilize for authentication process, is stored in a specific area within OTP as you saw on the previous slide and this is necessary to utilize security features on the processor.

The vast majority of the OTP memory is available for developer's use even though Analog Devices uses a small portion of this memory for our own uses. About 50,000 bits are available to you, the developer.

Subchapter 3c: Firmware and On-chip ROM

The on-chip ROM houses firmware which manages the security features on the processor. There are hardware extensions that help protect secrets. The firmware basically supports digital signature authentication and a secure state machine flow through the various modes of Open Secure Entry and Secure modes. Blackfin processors, the BF54x and the BF52x family that we use on this site as examples throughout this presentation vary slightly in their micro-architecture in terms of the implementation and the amount of ROM that we utilize on these processors. The BF54x processors include a 64 kilobyte, L1 instruction ROM, this operates in the core clock domain at core clock frequencies.

The Blackfin BF52x processors incorporate a 32 kilobyte on-chip boot ROM and the security firmware as well as the boot kernel resides in this boot ROM, it's shared. It operates in the system clock, or SCLK, domain and it runs at SCLK frequencies. The BF54x processors are targeted more towards the higher performance applications.

The BF52X processors are targeted more towards low power and low cost applications. This on-chip, ROM basically ensures integrity for the security firmware. In other words, integrity ensures unauthorized writes are prevented. So, for all intents and purposes, this ROM ensures that the firmware is tamper proof.

Chapter 4: Authentication Process on Blackfin
Subchapter 4a: Authentication Overview

Now, lets talk a little bit about the authentication process on Blackfin. And what we mean by the authentication process is specifically the authentication of digital signatures.

In this graphic representation, what we're showing here and what I want to call your attention to is the top portion of this slide shows you the off-chip message preparation. The bottom portion shows you what's actually occurring on the Blackfin itself. So everything that's going on up here is done off-chip. This is typically done on a host PC where you prepare your application code and data using for example our VisualDSP++ tools. This code or data can be un-encrypted or plain text or could be encrypted cipher text if confidentiality is required.

Once you have prepared your application code, and now you want to ensure it's integrity, it's authenticity, you're going to digitally sign that application code. The way that you do that is, you're first going to compute a hash upon your application code using SHA-1, secure one-way hash.

We're now going to take the result of that hash, the hash digest, and we're going to encrypt that small packet of information with the private key, the private elliptic curve cryptography key. Once we encrypt that hash result, you now essentially have a digital signature. For all intents and purposes, the digit signature is a fingerprint that uniquely identifies your message, your application code.

You now take a combination of the application code and the digital signature and you bring those in to the Blackfin's on-chip memory through booting or through DMA. Now you've got your digitally signed message, your combination application code and your digital signature on the Blackfin internal memory.

You're now going to go through the authentication process. The first thing you're going to do or the Blackfin's going to do, is decrypt the digital signature to obtain the original hash value. The Blackfin is also going to compute the hash value on the application code. This is going to yield two hash digests, the original hash digest and a calculated hash digest. We've going to compare these two hash values and if they are identical, that verifies that the application code did not change in any way; it did change inadvertently during the transfer to on-chip memory, did not change due to malicious tampering by an attack, for instance. Now this application code can be considered authenticated, trusted, and you can now give this application code full control over the Blackfin processor within the secure processing environment.

Subchapter 4b: Authentication Example

We're going to go through this authentication process end-to-end. In this example, what we're going to show you is a Blackfin BF54x device that's shown over here in the blue box. Everything within this blue box can be considered within the Blackfin device itself. Also, pay attention to the borders around this area, around the processor, for all intents and purposes. They'll change as we progress through this process and that will indicate when we're transitioning the secure state machine from the Open or unsecured mode into the privileged modes of access; the Secure Entry Mode and the Secure Mode.

In this particular example, we're showing the Blackfin device and we're going to be booting from a flash device. Most applications that involve Blackfin devices typically include some external memory. In this case, the BF54x can utilize DDR SDRAM.

So, the components that are depicted within the Blackfin device, the core, this is going to indicate actually where the program sequencer is executing code from. It can execute code from the L1 instruction memory, this can contain application code, it can execute code directly from the L1 secure entry ROM where the firmware resides that performs the authentication process. The core can also access data that is stored in the L1 data memory. This L1 memory can only house data. It cannot hold code as well.

We also have L2 unified memory on the BF54x devices. This can house both code and data.

Blackfin devices also contain a boot ROM. This holds the boot kernel that you utilize during the boot process and we also contain the public and private one-time programmable arrays.

So the very first thing that you need to do in order to utilize security features on the Blackfin is to personalize the device by storing your public key within the public area of OTP memory. The next thing you need to do is to prepare your application code and to digitally sign as we saw in the authentication process earlier. One other thing that you need to do is to prepare a very small piece of code, can be an unsecured piece of code. We're going to call this the kernel. And this kernel is going to basically bring in the application to be authenticated and it's also going to make a request for the authentication process to begin. The kernel can also contain other information such as error handling, it can be a portion of your real-time operating system. The kernel can be pretty much anything you want it to be that basically controls or initiates this process on the device.

All this information is then going to be stored in your boot source. In this case, we're booting from a flash device.

The Lockbox secure technology and the security features of the Blackfin are not restricted to just this one particular model. This is just one example that we're utilizing.

We're now going to perform the boot process. You can see here that the core is executing from the boot ROM, it's executing the boot kernel. And it's doing so in order to DMA in the small unsecured kernel code into the L1 instruction memory. Once that is complete, now the boot process is complete.

We're going to begin execution of that small kernel in the L1 instruction memory. You can see the core is pointing to the kernel and this kernel is now going to do several things. The first thing its going to do is bring in your digitally signed application code and place that into the L1 data memory. So both your digit signature and your application code and data are now in on-chip memory in the data memory portion of the Blackfin's memory. The next thing this kernel is going to do is request authentication and this is going to basically make a call to the firmware that resides in the secure entry ROM. We call this the Secure Entry Service Routine. Now if you notice the border around the Blackfin chip has changed in color and this is to depict the fact that we're now transitioning the secure state machine into a privileged mode of access, we're now in Secure Entry Mode. All the protection mechanisms are now activated and enabled on the device.

We now have a secure processing environment and the core is now going to begin executing the firmware directly out of the secure ROM. This firmware is now going to begin the authentication process.

The first thing that is going to be done in the authentication process is to perform a hash on the code that was brought into the L1 data memory. This will result in a hash result or a digest.

The next thing we're going to do is access the public key that resides in the public OTP memory. We're going to use this so we can decrypt the digital signature and obtain the original hash digest that resided in boot flash.

Now we're going to make a comparison of these two hash digests, the original hash digest and the calculated hash digest. Then we're going to see if they are identical. If they are identical, then this means that the authentication process resulted in success. So now we're going to take steps to allow the application code which can now be considered trusted or authenticated.

We're now going to allow this application code to execute on the Blackfin processor. It's currently sitting in L1 data memory . In order to execute on the processor, it has to first be moved to the L1 instruction

memory. So the firmware will actually conduct the DMA operation to copy the application code from data memory into the L1 instruction memory. And finally, the application code itself, now that it's considered trusted or authenticated, will be allowed to be execute directly on the processor out of L1 instruction memory.

The processor transitions into Secure Mode. We now have all of the protection mechanisms still enabled so we got unauthorized access to all of the on-chip memory level, L1 instruction memory, L1 data memory, L2 memory, all of that is protected from unauthorized DMA accesses, direct memory accesses. The JTAG emulation instructions have been disabled ever since we came into Secure Entry Mode that's maintained through Secure Mode, it's still disabled, and now you have access to the private area of OTP memory and any secrets that reside there. This application code now has complete control over the secure processing environment because it now also has control over the secure systems switches so it can change the memory protection scheme that's in place, it can access information that resides in OTP memory and can basically conduct whatever needs to be conducted within the application itself.

That's pretty much the process, end- to- end.

Chapter 5: Debug and Test Features
Subchapter 5a: Security and JTAG

Now let's talk a little about the debugging test features that are on these processors. One important consideration for security is JTAG. All of our Blackfin's processors support the JTAG test port and through this test port, you can perform in-circuit emulation. You have a great deal of access to the on-chip memory and resources of the Blackfin through emulation. So it's very important that we control enabling and disabling of this very powerful feature supported by our processors.

When this secure state machine is operating in Open Mode, all JTAG functionality is fully enabled, there are no restrictions in place. Upon entry into the Secure Entry Mode, in which the authentication process is performed, JTAG emulation is automatically disabled by default. It can be configured once you are in Secure Mode and you have access to all the security protection features in that mode. Firmware now is going to control all the execution out of the core and JTAG emulation can not be re-enabled in this mode. So this ensures that the firmware execution of code to conduct this authentication process cannot be

tampered with in any way. Control of the processor in terms of branching is not going to happen in this case.

Upon transition into Secure Mode, JTAG emulation is still disabled. Previously it was disabled in Secure Entry Mode if that continues. Now once you're in Secure Mode, your trusted application code can configure JTAG emulation. You can re-enable JTAG emulation in Secure Mode.

We've given you a couple different ways to do this to aid in your debug in the development process. You can enable JTAG just for the current Secure Mode session. In other words, when you exit Secure Mode and then go back into Open Mode, the JTAG emulation that was enabled is enabled in Open Mode and then the next time you go into Secure Mode through the authentication process, JTAG will be disabled by default again.

The second way that you can enable JTAG is in a persistent or sticky manner. You can enable JTAG once you're in Secure Mode to be persistent. In other words, the next time you that you exit Secure Mode and you go through the authentication process once again, JTAG will be enabled by default in Secure Mode when you enter it with your application code. This is really to facilitate your debug process. So imagine this scenario, you've got some problematic code that you're trying to debug. The problem only arises once you get into Secure Mode. By changing that code in order to enable JTAG, you've gotten rid of or changed your scenario such that the problem goes away and you can't debug your code. By enabling this persistent mode of JTAG emulation by going through the authentication process one time, and enabling JTAG emulation to be persistent, you can now exit Secure Mode, go back into Open Mode and then go through the authentication process one more time with your problematic secure function unaltered. The next time, this secure function now has access to the secure processing environment within the Secure Mode of operation, JTAG will be enabled by default and you can now break in with the emulator and you can conduct debug operations and see exactly why your problematic code is not behaving as you expect. It's a very powerful feature. It can be cleared in any mode and it will also be cleared by default when you go through a power cycle or when you reset the processor.

By default, there are a number of public JTAG instructions that we do not disable at any point in time. Public JTAG instructions that are defined in the 1149.1 IEEE specifications for JTAG such as boundary

scan and bypass mode are always enabled. So security will not get in your way when you're trying to perform debug and testing on the rest of your board, or the rest of your system.

Chapter 6: Summary and Conclusion
Subchapter 6a: Summary

So to summarize, this bullet came up before and I mentioned that it was the most concise statement to describe Lockbox secure technology. The security scheme is based upon the concept of authentication of digital signatures using standards based algorithms and provides a secure processing environment in which to execute code and protect assets.

This is a mixture of hardware and software mechanisms that are combined to provide you with a number of benefits which include authenticity or origin verification, integrity; in other words we prevent any unauthorized writes to your code and data.

Confidentiality. This prevents unauthorized reads of your code and data.

Renewability. Allows you to enhance you to update or upgrade the security in your system without having to tear the entire system down if a portion of your system is at any time compromised.

Security features are completely optional.

If you decide that you're going to utilize Blackfin processors because they are particularly suited for your application because they offer fantastic low power capabilities or a great peripheral mix and you decide you don't want to use the security features, you don't have to. They don't get in your way. They don't complicate your programming model. For all intents and purposes, Lockbox enabled Blackfin devices look just like our earlier released Blackfin devices that did not have Lockbox capability built into them, if you should decide never to use security features. Blackfin will boot up and come up out of reset in Open Mode with no access restrictions in place whatsoever.

This basically provides you, the developer, with a platform for e-commerce applications with the ability to implement IP and code protection in your applications and will also support digital writes management platforms.

Subchapter 6b: Conclusion

So I believe that I've shown today that Lockbox secure technology offers a very flexible and programmable platform for helping you secure your code and data.

Lockbox provides publicly accessible, user programmable, non-volatile, one- time programmable memory that enables the developer to program their own device ID's, it helps to ensure that these device ID's and other data stored in this non-volatile memory remain tamper proof.

Lockbox also provides you with a private area of non-volatile, one-time-programmable memory.
This allows you to store data and other parameters such as private cipher keys in a way that will maintain their confidentiality. In other words, this information will not be accessible. It will be completely invisible to unauthorized users and remain tamper proof.

Lockbox's Secure Mode provides a secure processing environment in which only your authorized trusted code is allowed to execute on the Blackfin processor. And only your trusted and authenticated code will have complete access to the protection mechanisms and to any confidential information you may have stored on the Blackfin device or within the rest of your system.

Chapter 7: Resources and References
Subchapter 7a: Resources and References

At the end of this presentation is a list of resources and references. Also, these slides are available for download so please note that there are speaker notes included on many of these slides within this presentation.

The Analog Devices website, www.analog.com/Blackfin includes a great deal of information to help you get up to speed quickly with Blackfin devices. For specific questions, please click on the ask a question button.

Additional links are provided here as well as textbook references to help you come up to speed quickly on cryptography and embedded systems concepts that have been described throughout this presentation.

Subchapter 7b: Cryptography Made Easy

If you are new to security and cryptography, I have listed a number of links here that I call "Cryptology Made Easy". They describe these concepts in very easy to understand layman's language. So I strongly recommend that you take a look at these.

Subchapter 7c: Glossary

The terms that I've used throughout this presentation are defined in some detail here in a glossary and also on the slide depicting some of the acronyms.

That concludes our presentation of Lockbox Secure Technology.

Thank you very much for joining me today.