LatticeXP2 Advanced Security Programming Usage Guide

November 2010 Technical Note TN1212

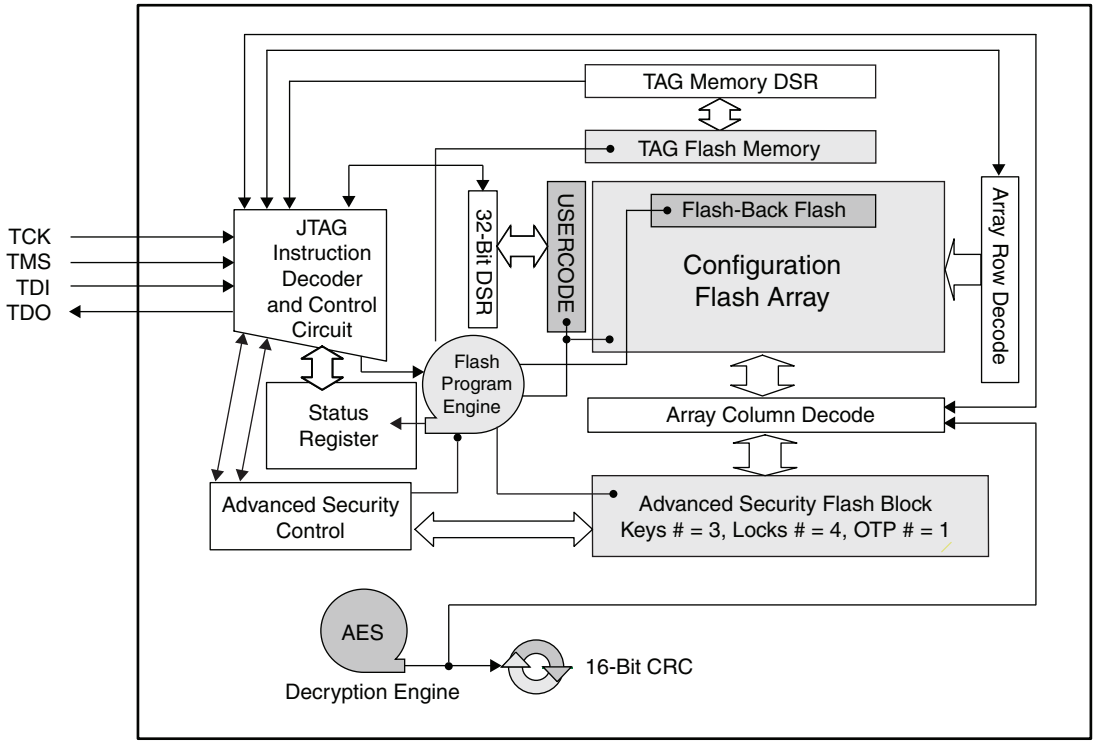
Introduction

The need for security on FPGA programmable logic devices has never been greater than today. As FPGA's start to play a role in larger and more critical system components, it is very important to protect the design from copying, reverse engineering, and tampering. Lattice has incorporated the Advanced Encryption Standard (AES) decryption core into the new LatticeXP2 devices and has also included the Flash-based lock technology. Together, they provide leading-edge security in a programmable logic device. Secure remote in-system programming (ISP) is now possible with AES encryption capability for the programming file during electronic transfer.

The LatticeXP2 devices contain state-of-the-art circuitry to make the Flash-based devices secure during and after programming. The configuration data (JEDEC file) loaded into LatticeXP2 can be decrypted prior to being written to the FPGA core using the AES 128-bit block cipher standard. The AES encryption key is stored in on-chip, non-volatile Flash memory. To successfully program a LatticeXP2 device that has the 128-bit encryption key programmed into it, a JEDEC file encrypted with the same 128-bit encryption key must be used.

This document explains the capabilities of this new security feature and how to take advantage of it.

Figure 1. LatticeXP2 Basic Block Diagram



Definitions

Erase

Clear all the Flash cells state to a logical one (1) (a.k.a. open fuse).

Program

Write into the selected Flash cells state a logical zero (0) (a.k.a. close fuse).

Configure

Write the pattern into the SRAM fuses.

Direct Mode

The device is in programming mode with all the I/O pins kept at tri-state.

Background Mode

The device is in programming mode with all the I/O pins remain operational.

Encryption

Use a password (Key) and an algorithm to scramble a file.

Non-Volatile Fuse

Fuses that keep the fuse state when power is turned off.

Key Code

The Binary Key pattern for encryption or decryption.

Key Code Size

The fixed length of the Key Code in bits. For the LatticeXP2 devices, it is 128 bits.

Key Lock Fuse

When programmed, the Key Lock fuse prevents the Encryption Key Code from being read out.

Decipher Key

The Key Code used for encryption or decryption.

Trusted Area

The real or virtual space that houses all confidential and high security material.

Unencrypted

No encryption action has taken place.

Encrypted

The encryption action has taken place.

Decrypt

Apply the reverse encryption process on an encrypted file.

Public Key

The Key Code that is not confined only in the Trusted Area.

Private Key

The Key Code that is confined only in the Trusted Area.

Authentication

The algorithmic validation process to determine a Pass/Fail results.

JEDEC File (.JED File)

The programming data file as defined by JEDEC 42.1C standard. The programming file is expressed in the ASCII 1 and 0 format. The file is printable. 3rd party programmers use it to support large volume production programming.

FPGA and PLD Security Background

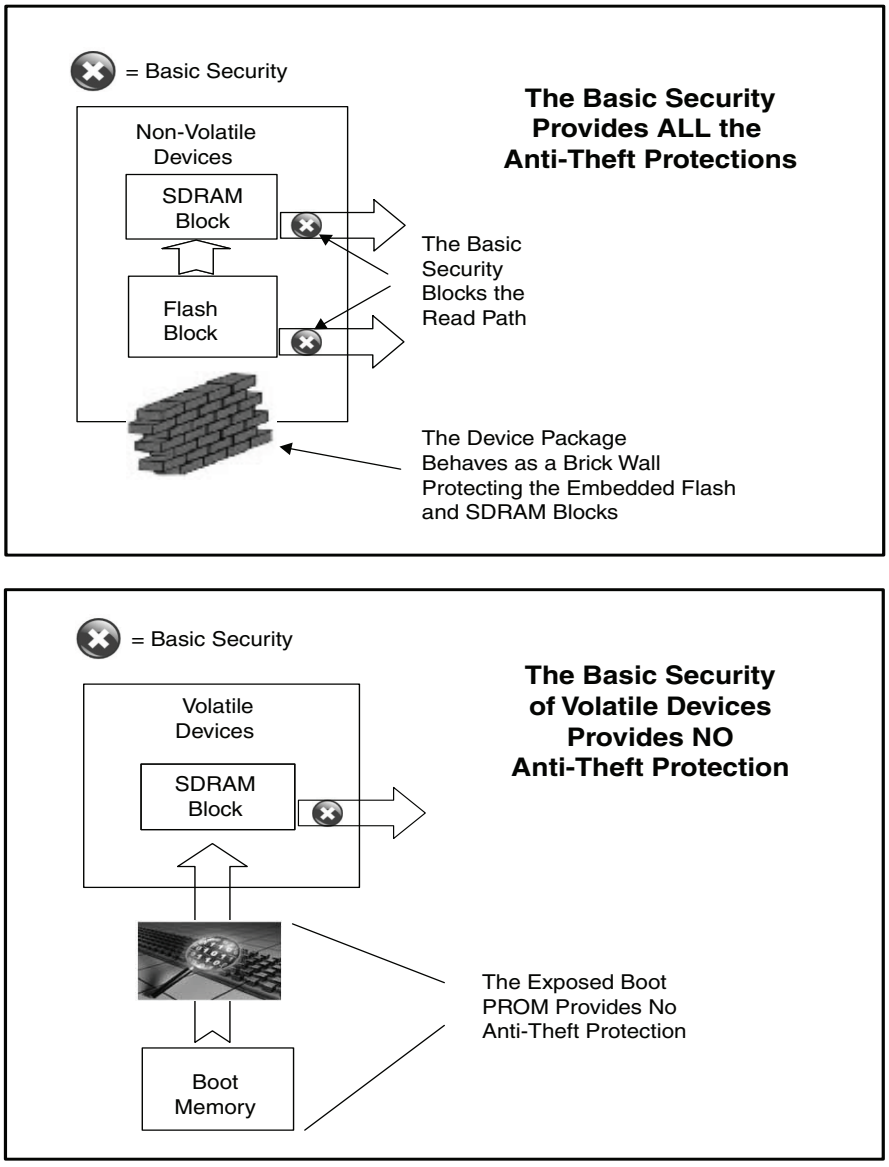
Non-Volatile Devices Have a Security Advantage

It is a well established fact that non-volatile FPGA and PLD devices are inherently much more secured than volatile devices. The advantage is derived from the simple fact that the Flash or EE fuse block is integrated and embedded into the same die.

Volatile Devices Security Features

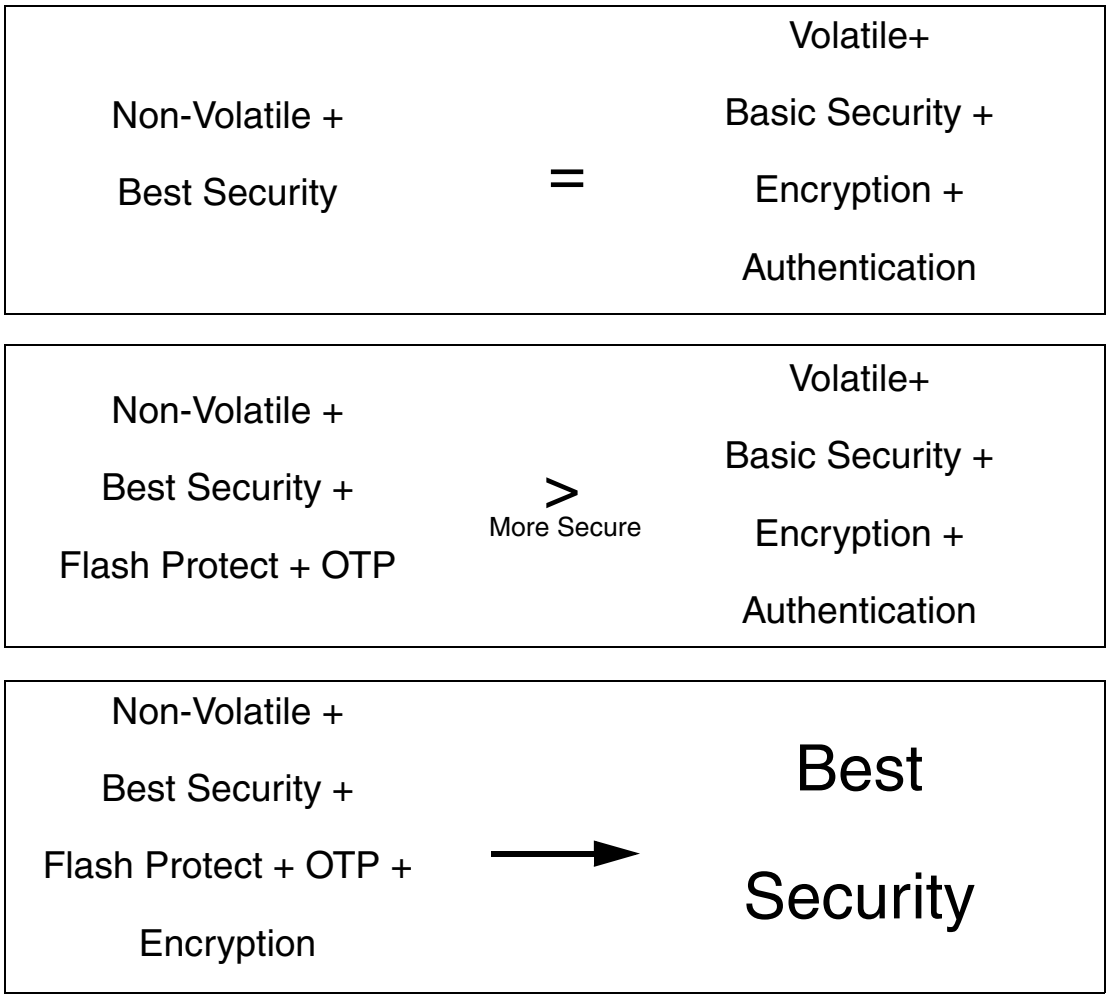
Volatile devices, on the other hand, rely on external Flash memory devices to store the bitstream. Thus, they inherently lack security. To compensate, encryption, and exotic authentication scheme must be deployed to bring the security to be in par with non-volatile devices.

Figure 2. Non-Volatile vs. Volatile Device Security



In order to compensate for the inherent vulnerable security of the volatile devices, encryption and authentication schemes are mandatory to be in par with non-volatile devices. This is illustrated in Figure 3.

Figure 3. Non-Volatile Device Security Advantages



There are many levels of security. They can span from the most basic to the highly sophisticated.

Purpose of Security

The main purpose of security is all about protection. The protections are:

1. Un-authorized Copying
2. Reverse Engineering
3. Over Production (where extra copies of a device and programmed and sold to a third party)

Basic Security

Non-volatile FPGA and PLD devices are all equipped with the time proven simple and yet effective basic security known as the read back blocking. This security scheme serves as the foundation of all advanced security feature. The reason is quite obvious that if the read back is defeated, the hackers can then simply read the pattern out of the device and security is compromised.

Highly Sophisticated Security

Authentication is where most of the development effort taking place to create the exotic IP in the name of security. They are expensive to support and yet the effectiveness is unknown. It is expensive in the following sense:

- Extra resources from the user logic (LUT counts) will be required for the authentication IP. The more exotic the IP, the more the resources will be required.
- An on board CPU will be necessary to run the exotic driver program to interact with the exotic authentication IP.

It does not matter how sophisticated the security scheme is, if the basic security does not work, the hacker can just read out the un-protected pattern programmed inside the device.

Non-volatile FPGA and PLD devices all have on-die non-volatile memory to store the pattern. Using the read back blocking security feature alone will be sufficient to protect the pattern. When the pattern reside inside a device is not available, it does not matter how effective the hacker skill in the art of breaking. The pattern remains safe and secured. Thus, a non-volatile device inherently is highly secured.

Volatile FPGA and PLD devices, on the contrary, depend on an external or exposed non-volatile memory to store the pattern. Read back blocking security alone inside the volatile device is useless for protecting the pattern stored in the exposed memory devices. The easiest method to protect the exposed bitstream is obviously by encryption. However, the fact of the matter is that the bitstream, albeit encrypted, still is fully exposed outside and thus can be copied and duplicated. Hence, authentication is introduced to help enhance the security of the exposed bitstream.

Encryption Key

Encryption is based on a static and pre-defined Key Code known as the Encryption Key. Once it is known, the exposed encrypted bitstream can be decrypted to compromise the bitstream. Non-volatile devices do not expose the encrypted bitstream, hence this cannot happen.

Dynamic Encryption Key

Authentication, in essence, is the scheme to insert a dynamic Encryption Key into the encryption scheme. The more exotic the scheme, the more sophisticated the method to pass the dynamic Encryption Key to the device. The simplest method is by passing on the Encryption Key once prior to device wake up. The more complicated one is by passing on the Encryption Key randomly and continuously.

The dynamic Encryption Key in some instance is considered more secure than the static Encryption Key. This is stemming from the assumption that once the device is removed from the board to a bench setup to hack, the dynamic Encryption Key will be lost to defeat the hacker. The static Encryption Key will remain and hacker will get to it eventually. This argument is valid only when the encrypted bitstream is available and the time to hack will be within the life cycle of the products. In other words, if it takes 20 years to break it, the product is already out of date therefore it serves no benefit to the hacker.

Caution

One must distinguish between the features that are really useful verses those features that only serve the purpose of enticing users to overinvest.

Examples of useful advanced security features are:

1. Read back blocking fuse(s).
2. 128-bit encryption for volatile devices.
3. Flash Protect, temporary or permanent.
4. Embedded Flash.

Examples of enticing security features that may lead to over-investing are:

1. 256-bit encryption for volatile devices.

2. Exotic authentication scheme.
3. Exotic encryption for non-volatile devices with secured embedded Flash.

The reason behind grouping the 256-bit encryption scheme into the over-invested enticing category is because of the fact that it is only mathematically correct that it is more secure than the 128-bit scheme. If the time it takes to break the 128-bit is already beyond the 10 year life cycle of the product, using the 256-bit encryption scheme to extend beyond 10 years is an act of over investing.

The reason behind grouping the exotic encryption for non-volatile devices in the over-invested enticing category is because of the fact that employing the encryption feature on a system that never needs field upgrade, will not enhance the security.

Summary

LatticeXP2 devices are inherently secured on the merit of the embedded Flash. The family also supports the advanced security features required to support different level of security appropriate to the users' particular application. Lattice does not recommend over-investing on security features that are costly to implement and yet serve no practical purpose in advancing the security of the user's system.

LatticeXP2 Advanced Security Features Summary

Note: The TAG Memory is available always for read and write. Thus, it is not affected by any security features setting.

Advanced Security Features

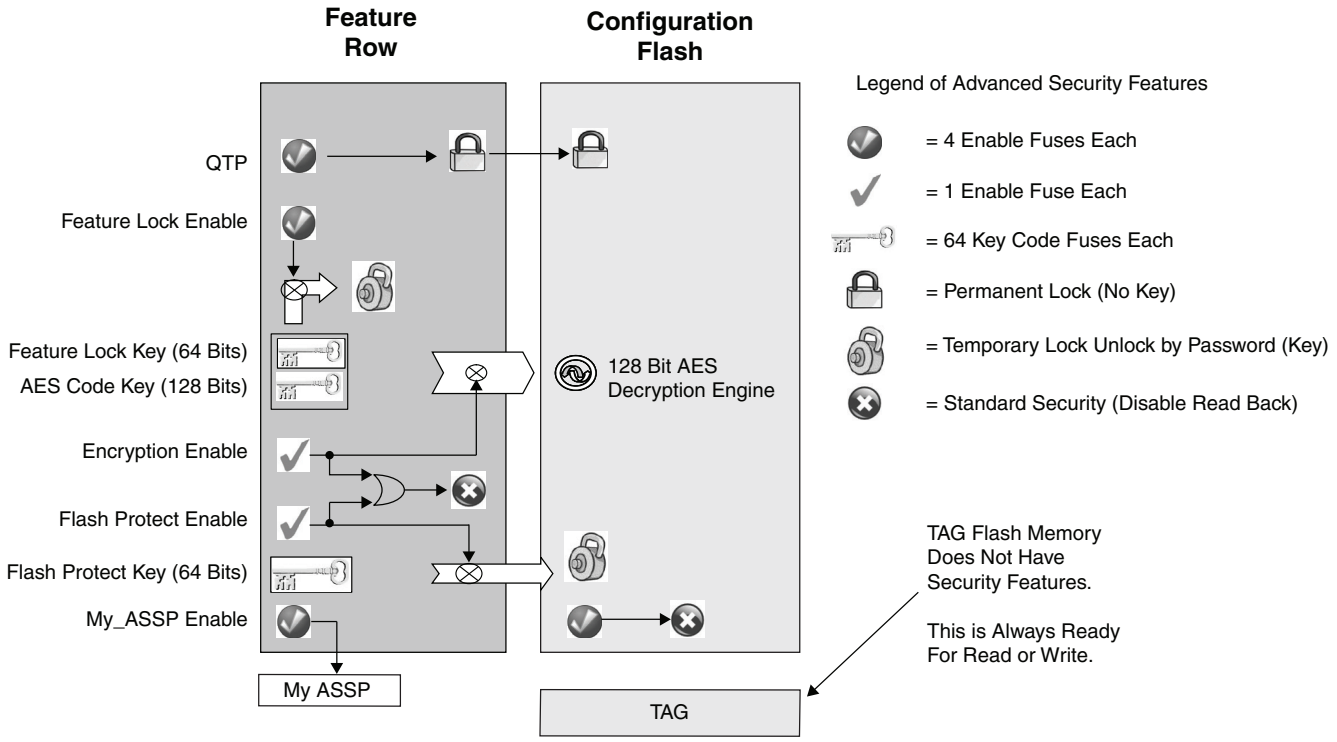
1. Flash Protection (Locks)
   a. OTP (One Time Programmable) can permanently lock all of the Flash fuses from erase or programming.
   b. Flash Protect can lock up all the Configuration Block Flash fuses from erase or programming unless the password is provided.
   c. Feature Lock can lock up the Feature Row Flash fuses from erase or programming unless the password is provided.

2. Encryption
   a. 128-bit Encryption Key
   b. AES Decryption Engine on die

3. Device Status Register
   a. Monitor Flash Protect Setting
   b. Monitor Encryption Setting
   c. Monitor Security Level Setting

4. Others
   a. My_ASSP: Supports customer specified IDCODE
   b. Standard Security
   c. SRAM CRC self check (One Shot SED)

Figure 4. Advanced Security Features Block Diagram



The Feature Row fuses are defined as follows to provide the Advanced Security feature support. The significant highest of interest are:

1. The 128-bit AES Key is formed by two 64-bit Flash fuse groups. The upper 64 bits (reference to bit 0 shifting in first) of the 128-bit of AES Key is also used as the Feature Lock password.
2. The Encryption Key and the password(s) will have no effect unless their respective enable fuse(s) are also programmed.
3. The Encryption Enable and the Flash Protect Enable fuse enable the features as well as secure the Feature Row from read back.
4. The 128-bit AES Key provided by user is not programmed into the Encryption Key fuses. Instead, it is first processed through the AES algorithm known as the scheduler to obtain the Last Round Code. The 128-bit Last Round Key is programmed into the 128 Encryption Key fuses.

Note: This fact is made transparent to users by writing both The Encryption Key user entered and the Last Round Key into the .bek file. Whenever ispVM displays the Encryption Key, it only display the one user entered. When programming, it will use only the Last Round Key.

Application Implications

If the Encryption Key is programmed into the device and the Feature Lock is enabled, the Feature Lock password must be the upper 64 bits of the Encryption Key Code. When reprogramming the Feature Row, ispVM will prompt the user to enter the 128-bit AES Key as the password. ispVM will shift the 128-bit Key into the device. However, the password register is only 64 bits in size, the lower 64 bits shifted in first will be flushed out of the password register, thus only the upper 64 bits will remain in the register. It is then used to un-lock the Feature Row for erase.

If Flash Protect and Feature Lock both are selected, Lattice recommends using different 64-bit Keys for Flash Protect and Feature Lock. The reason is that if they are the same, when reprogramming the device in the non-trusted area, the Flash Protect password must be provided. With the password, the non-trusted area could also erase the

Feature Row either intentionally or accidentally. In this case, when the device is returned to the user, it may not have Flash Protect enabled, and should be a reject.

Since the enable fuses also sets read back security, program the features in the Feature Row incrementally is theoretically possible but practically impossible. For that reason, it is strongly recommended to use ispVM to conduct programming instead of crafting the programming code.

Critical Points

1. The LatticeXP2 device can be in a JTAG daisy chain of devices if an encrypted JEDEC file is being programmed using the JTAG port. However, the LatticeXP2 is not fully IEEE 1149.1 compliant while it is being programmed with an encrypted JEDEC file.

   Reason: When data is being shifted into the LatticeXP2 and it is in a JTAG daisy chain, the data must be padded with trailer and header bits for the other devices in the JTAG chain. When the LatticeXP2 device is encrypted, the LatticeXP2 starts to decrypt the data shifted into the JTAG port, and will consider the trailer or header bits as valid data to decrypt.

   Work-Around Solutions:

   A. Slave SPI: Program the LatticeXP2 device using the Slave SPI port.
   B. JTAG: When processing the encrypted JEDEC, the encrypted data for each frame must be padded with dummy bits. The value of the dummy bits must be zero (0). If the LatticeXP2 is the only device in the JTAG chain, the padding is not required. The number of dummy bits depends on the position of the LatticeXP2 in the JTAG chain and can be calculated as follows.

      Number of zero (0) dummy bits = 128 – *x*

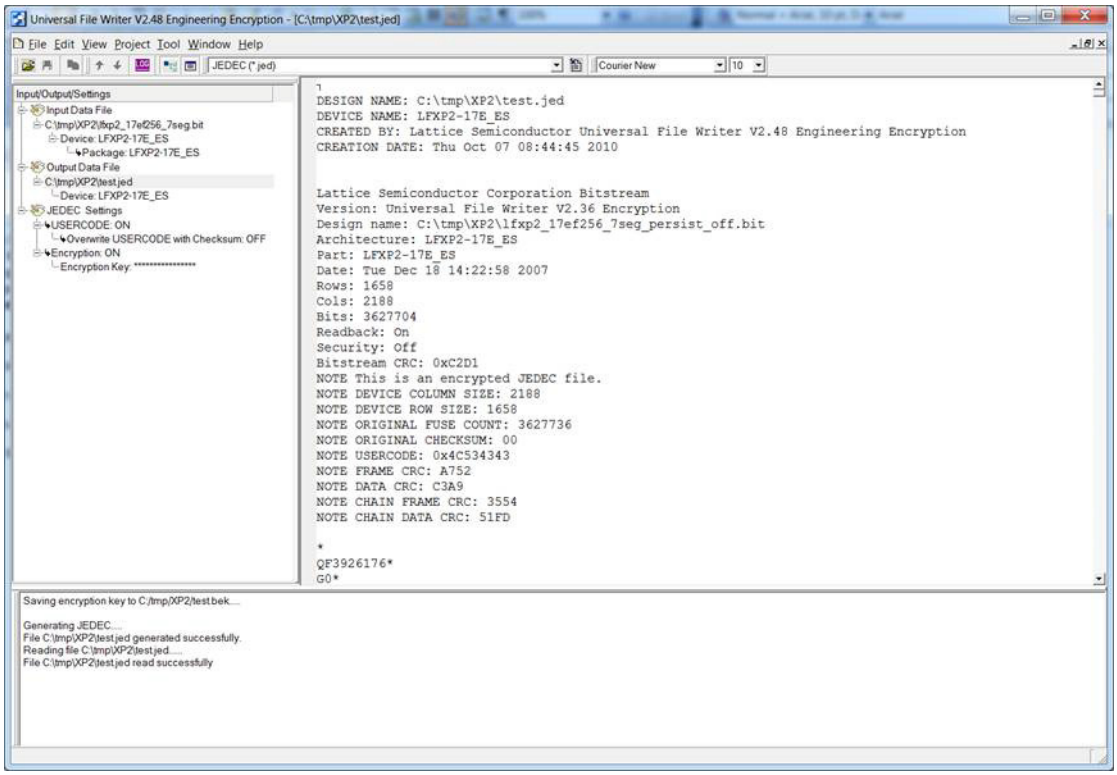
      Where: *x* = the position of the LatticeXP2 device in the JTAG chain (0, 1, 2 … 127).

      The first position = 0.

      - The maximum number of devices that can be in the JTAG chain is 128.
      - The LatticeXP2 devices can only be programmed with an encrypted JEDEC file sequentially. Turbo (concurrent) programming is not supported.
      - It is highly recommended that the LatticeXP2 device is the first device in the JTAG chain.

   C. Universal File Writer: The solution is available with ispVM System v17.7 or above. The AES Encrypted Files requires regeneration using ispUFW to enable the chain CRC information, as shown in Figure 5.

Figure 5. Using ispUFW to Enable the Chain CRC Information



LatticeXP2 Advanced Security Features Applications

There are five (basic and advanced) security features: Basic Security, OTP, Flash Protect, Feature Lock, and Encryption.

Theoretically it can provide 2^5 = 32 possible combinations. Lattice recommends users consider only 10 combinations since the rest of the combinations are not very useful. If there is a need for a combination that is not found on this document, it can be requested by sending an email to Lattice's technical support.

Instead of explaining the 22 unused combinations, this document would focus on justifying the 10 useful combinations. There are three principles determine if the combination is useful or not.

Principle 1: Encryption must be coupled with basic security.

Reason: When programming the device with an encrypted JEDEC file, the device will apply decryption algorithm to transform the encrypted JEDEC file into a normal (also known as unencrypted) JEDEC before programming the Flash fuses. If the basic security fuse is not set, the normal JEDEC can be read out from the device. All the effort of encryption will be wasted.

Principle 2: OTP can couple with basic security only.

Reason: When selecting the OTP feature, all Flash fuses except the TAG Memory, are prohibited from reprogramming. It renders the password base features and the encryption feature not applicable.

Principle 3: The Feature Row also needs protection.

Reason: It is important to note that the Feature Row and the Flash Block can be erased separately. Thus, there is a Feature Lock and Flash Protect to protect the Feature Row and the Configuration Flash block, respectively. Lattice strongly recommends selecting the Feature Lock when the Flash Protect and/or Encryption Key are permanently programmed into the device.

Note: Once the Feature Lock is enabled, in order to reprogram the device, including the passwords and/or Encryption Key residing in the Feature Row, ispVM will require the user to provide the Feature Lock password to erase the entire device first. Thus, Lattice recommends selecting the feature only if there is no need to reprogram the Feature Row.

The ispVM System is designed to make the complicated programming flow transparent to users. All user needs to do is decide which combination is needed, and then select the combination in ispVM. The ispVM System will carry out the many tasks in the correct sequence. This will save users the precious time from drilling into the technical detail of the complicated programming algorithm and the sequence.

Table 1. LatticeXP2 Advanced Security Selection Combinations

Feature	Advanced Security Selection Combinations									
	1	2	3	4	5	6	7	8	9	10
Encryption	✔	✔					✔	✔		
Feature Lock		✔			✔	✔		✔		
Flash Protect			✔	✔	✔	✔	✔	✔		
Basic Security	✔	✔		✔		✔	✔	✔	✔	
OTP									✔	✔

Simple User Guidelines in Selecting the Combinations

Please consider the following simple rules to select the combinations.

1. Select Encryption if the device will be programmed in the field (including field upgrades).

   Reason: During field upgrades, the pattern is exposed to the outside world. The only method to protect the pattern is by encryption.
2. Select Flash Protect if the device will be programmed by embedded CPU.

   Reason: The CPU will need to service multi-tasks and thus might accidentally fire the programming code. The Flash Protect offer extra protection.
3. Select Encryption and Flash Protect if the device will be programmed by embedded CPU in the field.

   Reason: In addition to reasons provided on guideline 1 and 2 above, the Flash Protect also will restrict the field upgrade only to the authorized personal.
4. Select OTP only if it is absolutely necessary.

   Reason: Using Flash Protect and then throw away the password is as good as OTP. Thus, users are recommended to think twice before using this feature.
5. Always select the Feature Lock when the Flash Protect password and/or Encryption Key are programmed and will not be changed.

   Reason: If the Feature Lock is not selected, the device can be erased by first erasing the Feature Row to wipe out password and the encryption key, and then by erasing the Configuration Flash. It would be undesirable when deploying all the advanced security features and leaving the device vulnerable to being completely erased.

Note: The ispVM System will program the standard security fuse automatically when programming the encrypted JEDEC file into the device. The purpose is to protect the pattern from being read out in case the user forgot to select the Feature Lock.

LatticeXP2 Advanced Security Features Deployment Process

The trademark feature of the ispVM System software is to let the user select the operation and it will take care of the details. The deployment of advanced security is bit more complicated. The support strategy still is similar. Users selects the combination, ispVM will carry out the programming task to enable the various features using the carefully crafted programming sequence. The details of the sequence are documented later in this document for interested reader.

There are two points the user can choose to deploy the advanced security feature.

1. The first point can be in ispLEVER. ispLEVER can generate the appropriate JEDEC file and the companion .bek file.
2. The second point can be in ispVM System. The ispVM System will perform JEDEC file conversion and output the .bek file.

Deferring the advantage security selection until using ispVM provides the highest flexibility. For example: The Flash Protect Key or Encryption Key can be selected and change on a fly.

However, from a security's point view, when an unencrypted JEDEC file is generated by ispLEVER, the Encryption Key must also be exposed. It will be up to users to pay extra attention to protect the unencrypted JEDEC file and the exposed Flash Protect Key and the Encryption Key pattern. Also, the programming operation must take place in a trusted area.

Users are urged not to send the Encryption Key code and encrypted JEDEC file to the same un-trusted area. If both are available, for the skill in the art of encryption, only a simple decryption program is needed to use the Encryption Key to convert the encrypted JEDEC into normal JEDEC to compromise the design.

Lattice strongly recommends the Encryption Key be programmed only in the trusted area. If the programming has to be done outside of the trusted area, then at the least use two separate un-related locations to program the Encryption Key and the Encrypted JEDEC file.

There are two methods to enter the various Protect passwords and the Encryption Key code into ispVM.

1. Use .bek file generated by ispLEVER or ispVM.
2. Enter directly onto ispVM GUI.

Lattice strongly recommends users to transmit Key and passwords using .bek files for better protection. If the Key or password is entered manually in the directly into the GUI, it is susceptible to human error. Typical human errors are reading, hearing, or typing the key in incorrectly.

When entered directly onto ispVM GUI, ispVM will save the user-entered Key and password(s) into a file as a record. Due to AES encryption is under Government export control, the ispVM System software available on Lattice's Web does not support encryption. Thus, it cannot save the encrypted .bek file. Users can obtain the encryption patch from Lattice's technical support to upgrade their ispVM System to support encryption. If the encryption patch is not installed, ispVM System will write the Key and the passwords into the .XCF file. The ispVM System will warn the user that the .XCF is not encrypted. Thus, it is users' responsibility to ensure the .XCF file will not fall into unauthorized hands.

LatticeXP2 Advanced Security Features Programming

The Advanced Security Feature Row programming can be performed on the traditional platforms.

1. ispVM System
2. SVF or STAPL files for ATE
3. Third party programmers
4. Lattice's Model300 desk top programmer

ispVME Embedded programming is not listed as the programming platform for the Feature Row. Programming the Feature Row fuses on embedded platform during field upgrade is not recommended due to the exposure of the Key code or password to the public domain. Field upgrade programming shall be restricted to the Configuration Flash only.

Lattice works continuously with the third party vendor to provide programming support for the advanced security features mentioned in this document. Users are encouraged to submit support request to System General or BPM Microsystems.

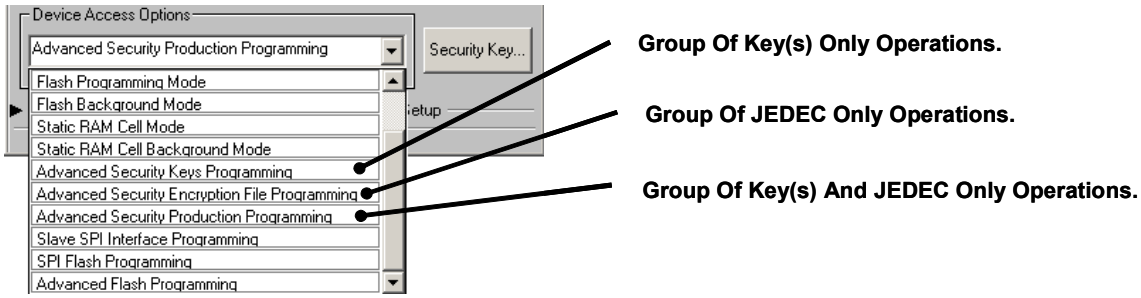
In order to minimize operator error and maximize the advanced security feature programming experience for majority of users, ispVM System provides well crafted programming flows to ensure successful programming.

A. The ispVM System cross checks the operation selected in the GUI with the advanced security settings in the JEDEC file before performing the programming action to minimize operator error. For example, the OTP fuse can only be programmed if the G field in the JEDEC file is greater than or equal to 2 and if the ispVM operation selected includes OTP fuse programming.

B. The ispVM System has predefine programming sequence of each combination supported. For example, if the OTP feature is selected, it must be the last fuse to be programmed.

C. The ispVM System provides three different categories of programming operations that will cover a majority of the users needs. The programming options are available in the Device Access Options in the Device Information dialog in ispVM System GUI as shown in Figure 6.

Figure 6. ispVM System Advanced Security Programming Categories

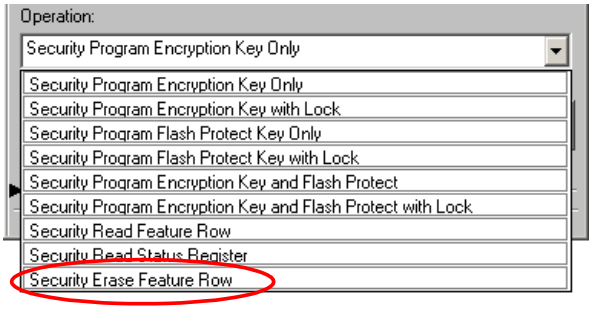


1. Key Programming Operations
   Operations in this category are for programming the Feature Row of blank (also known as erased) devices. These operations are available under Advanced Security Keys Programming mode under Device Access Options in the Device Information dialog in ispVM System.
2. Encryption (JEDEC) File Programming Operations
   Operations in this category are for reprogramming the Configuration Flash Block only. The user cannot change the Feature Row. These operations are available under Advanced Security Encryption File Programming mode under Device Access Options in the Device Information dialog in ispVM System.
3. Production Programming Operations
   Operations in this category are for programming both the Feature Row and the Configuration Flash of blank devices. These operations are available under Advanced Security Production Programming mode under Device Access Options in the Device Information dialog in ispVM System.

One operation falls into both the Key Programming and the Production Programming Operations categories, erasing the Feature Row. This operation prepares a device for reprogramming the Feature Row with a different Flash Protect Key, or to add or delete an advanced security feature. The Feature Row must be erased before any of those actions can be performed. While erasing the Feature Row, ispVM will also erase the Configuration Flash block to

ensure the device will be blank. After the device is erased, users can select an operation from the Key Programming and the Production Programming Operations categories to reprogram the Feature Row. In summary, changing any of the advanced security features is a two step process.

Figure 7. Erasing the Feature Row



It is beyond the scope of this document to describe or define the programming sequence of the advanced security features. It is impractical to provide adequate information for users to hand craft their own custom flow. Instead, users are encouraged to request the programming flow specific to their need from Lattice's technical support.

LatticeXP2 Device Status Register

The LatticeXP2 devices have an 8-bit status register, which indicates the status of the device, and is defined in Table 2.

Table 2. Device Status Register

Status Bit No.	Status Bit	Description	Bit State	
			0	1
0	Pass/Fail	The erase and programming status of Flash. SRAM is always 0 (pass).	Pass	Fail
1	Done Fuse	The state of the Flash or SRAM done fuse, depending on the device programming mode.	Erase	Program
2	Protect	The status of the Flash Protect Key Comparator. If the Feature Lock is not on, this bit set to 1 if key does not match.	Match	No Match
3	CRC	The status of the SED CRC in user mode.	Pass	Fail
	Erase Feature	The status of the Feature Row erase and programming enable bit. The device must be in Edit Mode.	No Match	Match
4	OTP	Indicates the status of OTP feature.	Off	On
5	Feature Lock	Indicates the status of Feature Lock.	Off	On
6	Encryption	The status of the encryption enable fuse. It is on if the fuse is programmed.	Off	On
7	Security	The state of the read back preventive security of Flash or SRAM fuses depends on programming mode.	Erase	Program

LatticeXP2 Security JTAG Instruction Set

The LatticeXP2 Security JTAG Instruction Set is listed in Table 3.

Table 3. LatticeXP2 Security JTAG Instruction Set

Security JTAG Instruction Set for LatticeXP2 Family	Code Bit 7…Bit 0	Target Register	Description
ISC_DISABLE	00011110	Bypass	Standard IEEE 1532 ISC instruction.
ISC_ENABLE	00010101	Bypass	Standard IEEE 1532 ISC instruction.
XPROGRAM_ENABLE	01010011	Bypass	Lattice JTAG instruction to enable Background Flash Mode to activate the Flash erase, programming, and read back command.
ENCRYPT_PROG_INC	01000000	128-bit Decrypt Register	Lattice JTAG instruction to support encrypted Flash programming.
PROTECT_SHIFT	01000001	64-bit Flash Protect	Lattice JTAG instruction to shift in the 64-bit Password.
LSC_REFRESH	00100011		Lattice JTAG instruction to Tri-state I/O and clear all SRAM fuses. If the done fuse is programmed, trigger a Configuration Flash to SRAM transfer. If the done fuse is not programmed, the Flash to SRAM transfer will not happen. Note: This command will not work if ISC_DISABLE is not issued. Please refer to the flow.
PROGRAM/STATUS	01010010	Program Status	Lattice JTAG instruction to read the one bit complete status register for Flash.
READ_STATUS	10110010	8-bit Status Register	Lattice JTAG instruction for non-destructive read of 8-bit status register.
PROGRAM_FEATURE	11000000	Feature Register	Lattice JTAG instruction to program the Feature Row Flash.
READ_FEATURE	11000100	Feature Register	Lattice JTAG instruction to verify the Feature Row Flash.
ERASE_FEATURE	11000011	Bypass	Lattice JTAG instruction to erase the Flash fuses in the Feature Row.
READ_16_CRC	11000101	16-bit CRC	Lattice JTAG instruction for non-destructive read from the 16-bit CRC register.
RESET_16_CRC	11000110	Bypass	Lattice JTAG instruction to clear the 16 bits CRC register.

Reprogramming a Secured Device

When reprogramming the embedded Flash on a secured device, the device will fail verification. This is due to the security fuse latch is not cleared by the Erase command. Below are the two work-around solutions for reprogramming a secured LatticeXP2 device.

Direct Programming Mode Flow

1. Erase the device.
2. Disable configuration mode by issuing the ISC_DISABLE command.
3. Issue the LSC_REFRESH command to wake-up the device and clear the security fuse latch.
4. Enable background programming mode back by issuing the XPROGRAM_ENABLE command and continue the programming flow.

Background Programming Mode Flow

1. Follow the regular Erase and Programming flow.
2. Skip the Flash verification.

Advanced Security Features

A list of Advanced Security features available on the LatticeXP2 is shown in Table 4.

Table 4. Advanced Security Feature Description Table

Resources/ Features[1]	Flash		Data File	Descriptions
	Bits	Block		
AES Key [2]	128	Advanced Security	Key File	The 128-bit user defined AES Encryption Key.
Feature Lock Key [3]	64			The password (or Key) to unlock the feature block for reprogramming.
Feature Lock Enable	4		NA	Enables the lower 64-bit of the AES Key as Feature Row Lock. Secures all Keys from being read.
AES Key Enable	1		NA	Enable the AES Key and the decryption engine.
Flash Protect Key [2]	64		Key File	The password (or Key) to unlock the Configuration Flash for reprogramming.
Flash Protect Enable	4		NA	Enable the Flash Protect Key.
my_ASSP	1		JEDEC Files [4]	Enable the replacement of JTAG IDCODE by the USERCODE.
OTP	4			Enable the One Time Programmable feature except TAG and User Flash.
Security	Multiple	Config.		Standard security fuses to protect the pattern from read back.

Notes:

1. All these feature settings do not affect the TAG Memory Module and the Flash Back Feature.
2. The 128-bit AES Key code and the 64-bit Flash Protect Key code are stored in the KEY file generated by Diamond or ispLEVER.
3. The AES Key and Feature Lock are shared. This effectively provides the highest protection to the AES Key. Once the AES Key is programmed into the device, reprogramming the device to a different AES Key will require shifting into the device the original 128-bit AES Key.
4. The my_ASSP, OTP, and Security selection are stored in the JEDEC file.

ispVM Programming Operations and Security Combinations

As explained before, advanced security features are not programmed into the device one feature a time. Instead, they are all programmed into the device simultaneously. The ispVM System takes care of the programming complexity by using carefully crafted programming flow for each operation listed. The user must select the correct operation to program the correct advanced security features into the LatticeXP2 devices. Figure 8, Figure 9, and Figure 10 serve look up tables to illustrate the relationship between the Advanced Security Feature selected and the corresponding programming operation.

As previously described, the operations are classified into three groups.

1. Key Programming Operations
   The purpose is to prepare the device for the Encryption (JEDEC) File Programming operations. The programmed devices do not function since a JEDEC file has not yet been programmed into the device.
2. Encryption (JEDEC) File Programming Operations
   The purpose is to program the JEDEC pattern into the device after a Key Programming operation has been completed. Field upgrades are JEDEC file only operations and are typically performed using background programming operations. The background operations are listed so that users can program the device or generate the VME file for embedded programming.
3. Production Programming Operations
   The purpose is to program all the user selected advanced security feature(s) and the JEDEC file into the device in one operation. Production programming using ATE, such as Agilent 3070 Board Tester or third party BSCAN tools would use this type of operations to generate the appropriate file(s).

Figure 8. Key Programming Operations and the Security Combinations

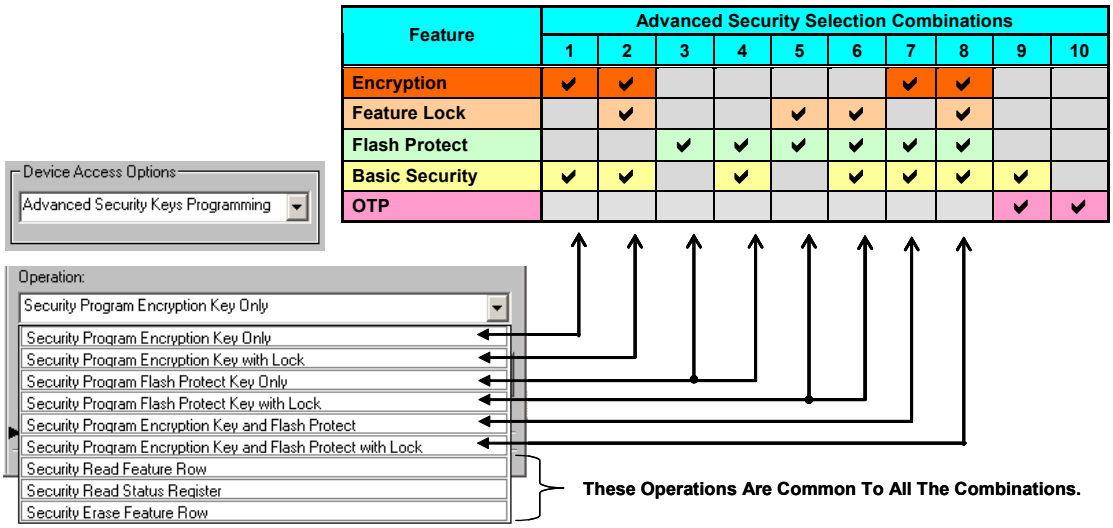


Figure 9. Encryption (JEDEC) File Programming Operations and the Security Combinations

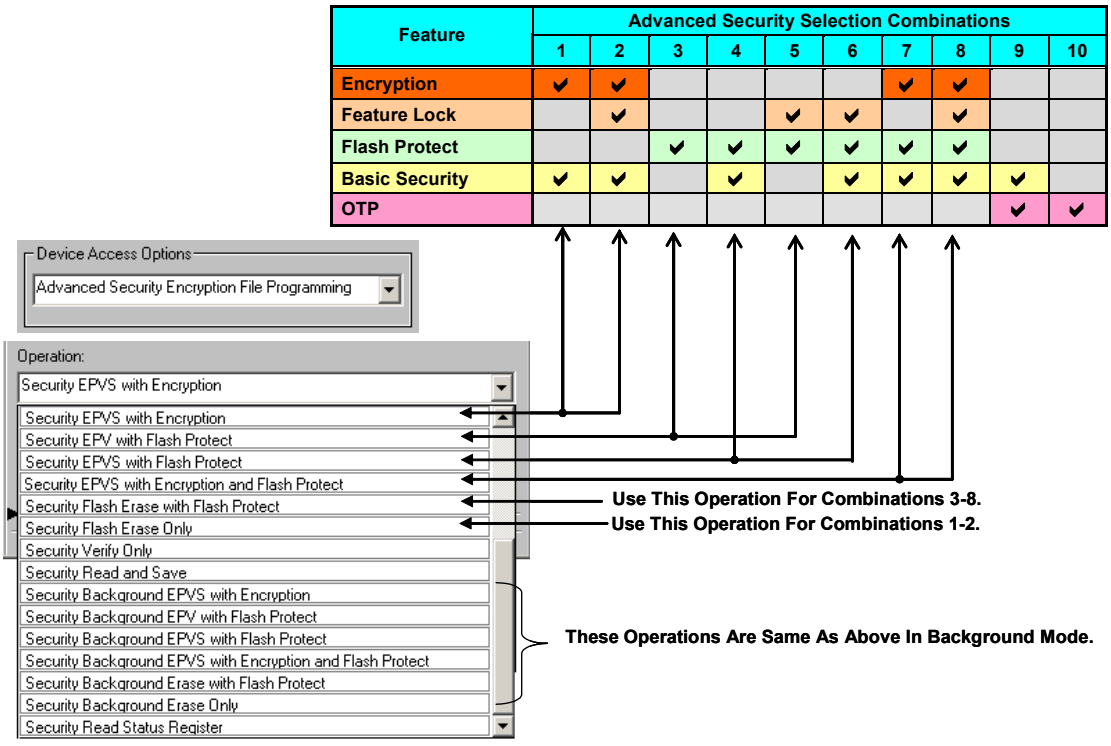
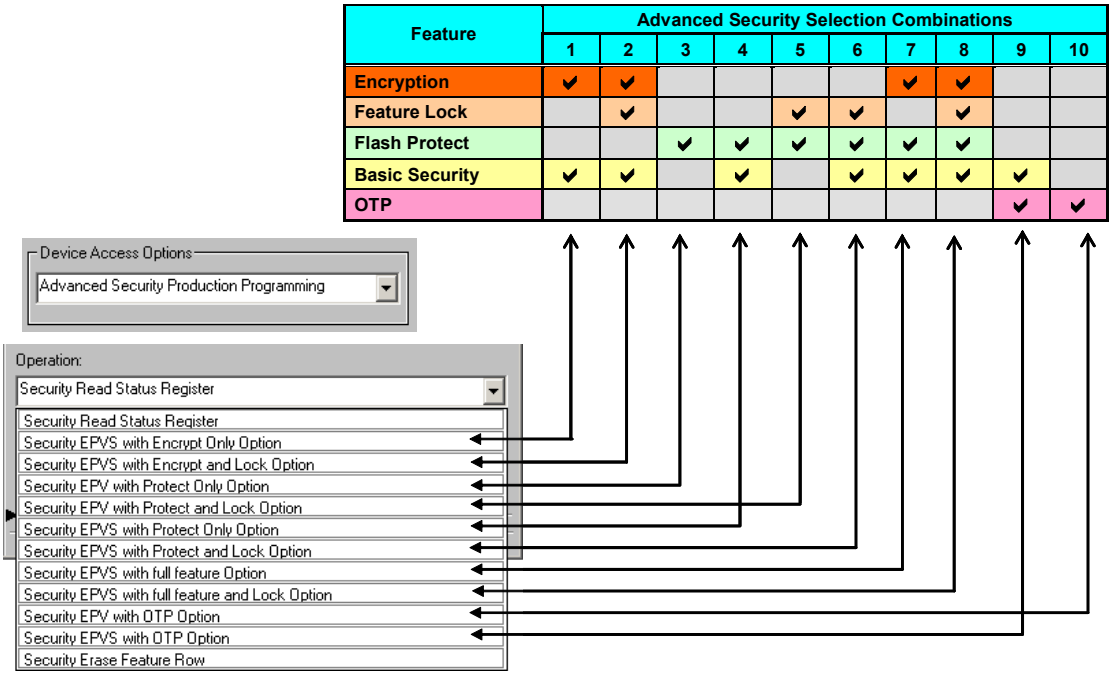


Figure 10. Production Programming Operations and the Security Combinations



Board Design using ispVM System

Table 5. Procedure for Programming the Flash Protect Key

Steps	Description	
1	Launch the device selection menu.	
2	Select the LatticeXP2 device.	
3	Select Advanced Security Keys Programming under Device Access Options.	
4	Select Security Program Flash Protect Key Only under Operation. There are two operations you can choose:	
	Operations	**Comments**
	Security Program Flash Protect Key Only.	Program the Flash Protect Key into the device.
	Security Program Flash Protect Key with Lock.	Program the Flash Protect Key and the Feature Lock Enable. This will prevent the Feature row of being read back out.
5	The Flash Protect Key dialog will automatically open after the operation is selected. It can also be opened by clicking on the Protection Key button.	
6	You can load the Flash Protect Key from an existing .key file by clicking the Load From File button	
6.1	This step is optional. You can unselect the Hide Password option to view the Password in the GUI.	
6.2	Enter the key file password and click OK, and jump to step 11. This step is optional if you used the default Password.	
	Enter the Flash Protect Key Manually into the GUI	
7	Select the Key format under Protection Key Format. If you want to use the Hexadecimal format, select the Hexadecimal.	
8	This step is optional. You can unselect the Hide Protection Key option to view the Key in the GUI.	
9	Enter the Flash Protect Key. If enter less than the full Key, then fillers will be padded on the left (the most significant) position. If enter more, then the GUI will block the excessive entry.	
10	Re-Enter the Flash Protect Key.	
10.1	This step is optional. You can save the Flash Protect Key to a .key file by clicking the Save To File button. The software will ask for a key file name, enter the name. This option is only available only if the ispVM System encryption patch is installed.	
10.2	This step is optional. You can unselect the Hide Password option to view the Password in the GUI.	
10.3	Enter the key file password and click OK.	
11	Click Apply to come back to the Device Information Dialog.	
12	Click OK to come back to the ispVM Windows. ispVM generates a chain description file, which can be saved as an .xcf file which will contain the Flash Protect Key.	
13	Select the GO button to program the Flash Protect Key into the device. For embedded programming, skip this step.	

Figure 11. Using ispVM to Program the Flash Protect Key into the Device

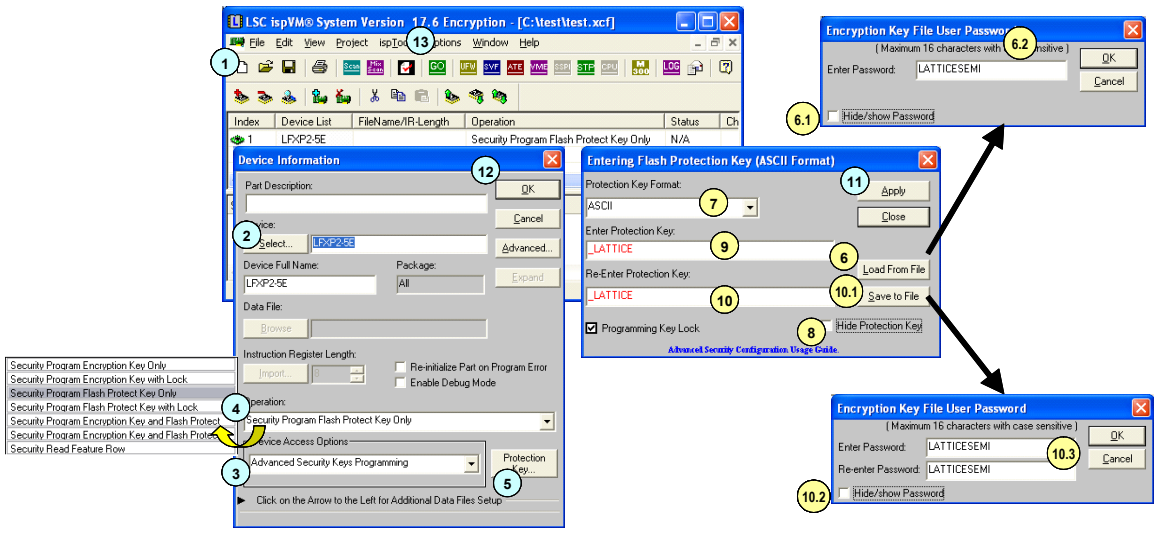


Table 6. Procedure for Programming the Encryption Key

Steps	Description	
1	Launch the device selection menu.	
2	Select the LatticeXP2 device.	
3	Select Advanced Security Keys Programming under Device Access Options.	
4	Select Security Program Encryption Key Only under Operation. There are two operations you can choose:	
	Operations	**Comments**
	Security Program Encryption Key Only.	Program the 128-bit AES Key into the device.
	Security Program Encryption Key with Lock.	Program the 128-bit AES Encryption Key and the Feature Lock Enable. This will prevent the Feature row of being read back out or reprogrammed.
5	Launch the Encryption Key dialog by clicking on the Security Key button.	
6	You can load the Encryption key from an existing .bek file by clicking the Load From File button.	
6.1	This step is optional. You can unselect the Hide Password option to view the Password in the GUI.	
6.2	Enter the key file password and click OK, and jump to step 11. This step is optional if you used the default Password.	
	Enter the Encryption Key Manually into the GUI	
7	Select the Key format under Encryption Key Format. If you want to use the Hexadecimal format, select the Hexadecimal.	
8	This step is optional. You can unselect the Hide Encryption Key option to view the Key in the GUI.	
9	Enter the Encryption Key. If enter less than the full Key, then fillers will be padded on the left (the most significant) position. If enter more, then the GUI will block the excessive entry.	
10	Re-Enter the Encryption Key.	
10.1	This step is optional. You can save the Encryption Key to a .bek file by clicking the Save To File button. The software will ask for a key file name, enter the name.	
10.2	This step is optional. You can unselect the Hide Password option to view the Password in the GUI.	
10.3	Enter the key file password and click OK.	
11	Click Apply to come back to the Device Information Dialog.	
12	Click OK to come back to the ispVM Windows. ispVM generates a chain description file, which can be saved as an .xcf file which will contain the Encryption Key.	
13	Select the GO button to program the Encryption Key into the device. For embedded programming, skip this step.	

Figure 12. Using ispVM to Program only the Encryption Key into the Device

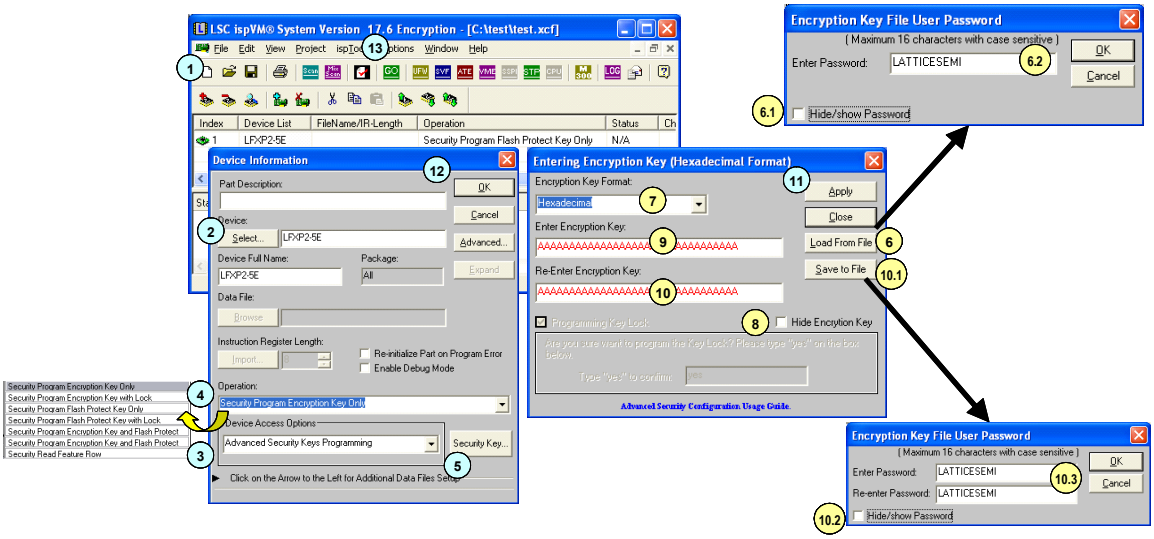


Table 7. Procedure for Programming the Flash Protect and Encryption Key

Steps	Description	
1	Launch the device selection menu.	
2	Select the LatticeXP2 device.	
3	Select Advanced Security Keys Programming under Device Access Options.	
4	Select Security Program Encryption Key and Flash Protect under Operation. There are two operations you can choose:	
	Operations	**Comments**
	Security Program Encryption Key and Flash Protect.	Program the Flash Protect Key and 128-bit AES Key into the device.
	Security Program Encryption Key and Flash Protect with Lock.	Program the Flash Protect Key, the 128-bit AES Encryption Key, and the Feature Lock Enable. This will prevent the Feature row of being read back out or reprogrammed.
5	The Flash Protect Key dialog will automatically open after the operation is selected. It can also be opened by clicking on the Protection Key button.	
6	You can load the Flash Protect Key from an existing .key file by clicking the Load From File button.	
6.1	This step is optional. You can unselect the Hide Password option to view the Password in the GUI.	
6.2	Enter the key file password and click OK, and jump to step 11. This step is optional if you used the default Password.	
	Enter the Flash Protect Key Manually into the GUI	
7	Select the Key format under Protection Key Format. If you want to use the Hexadecimal format, select the Hexadecimal.	
8	This step is optional. You can unselect the Hide Protection Key option to view the Key in the GUI.	
9	Enter the Flash Protect Key. If enter less than the full Key, then fillers will be padded on the left (the most significant) position. If enter more, then the GUI will block the excessive entry.	
10	Re-Enter the Flash Protect Key.	
10.1	This step is optional. You can save the Flash Protect Key to a .key file by clicking the Save To File button. The software will ask for a key file name, enter the name. This option is only available only if the ispVM System encryption patch is installed.	
10.2	This step is optional. You can unselect the Hide Password option to view the Password in the GUI.	
10.3	Enter the key file password and click OK.	
11	Click Apply and the Encryption Key dialog will automatically open.	
12	You can load the Encryption key from an existing .bek file by clicking the Load From File button.	
12.1	This step is optional. You can unselect the Hide Password option to view the Password in the GUI.	
12.2	Enter the key file password and click OK, and jump to step 17. This step is optional if you used the default Password.	
	Enter the Encryption Key Manually into the GUI	
13	Select the Key format under Encryption Key Format. If you want to use the Hexadecimal format, select the Hexadecimal.	
14	This step is optional. You can unselect the Hide Encryption Key option to view the Key in the GUI.	
15	Enter the Encryption Key. If enter less than the full Key, then fillers will be padded on the left (the most significant) position. If enter more, then the GUI will block the excessive entry.	
16	Re-Enter the Encryption Key.	
16.1	This step is optional. You can save the Encryption Key to a .bek file by clicking the Save To File button. The software will ask for a key file name, enter the name.	
16.2	This step is optional. You can unselect the Hide Password option to view the Password in the GUI.	
16.3	Enter the key file password and click OK.	
17	Click Apply to come back to the Device Information Dialog.	
18	Click OK to come back to the ispVM Windows. ispVM generates a chain description file, which can be saved as an .xcf file which will contain the Encryption Key.	
19	Select the GO button to program the Encryption Key into the device. For embedded programming, skip this step.	

Figure 13. Using ispVM to Program the Encryption Key and Flash Protect Key into the Device

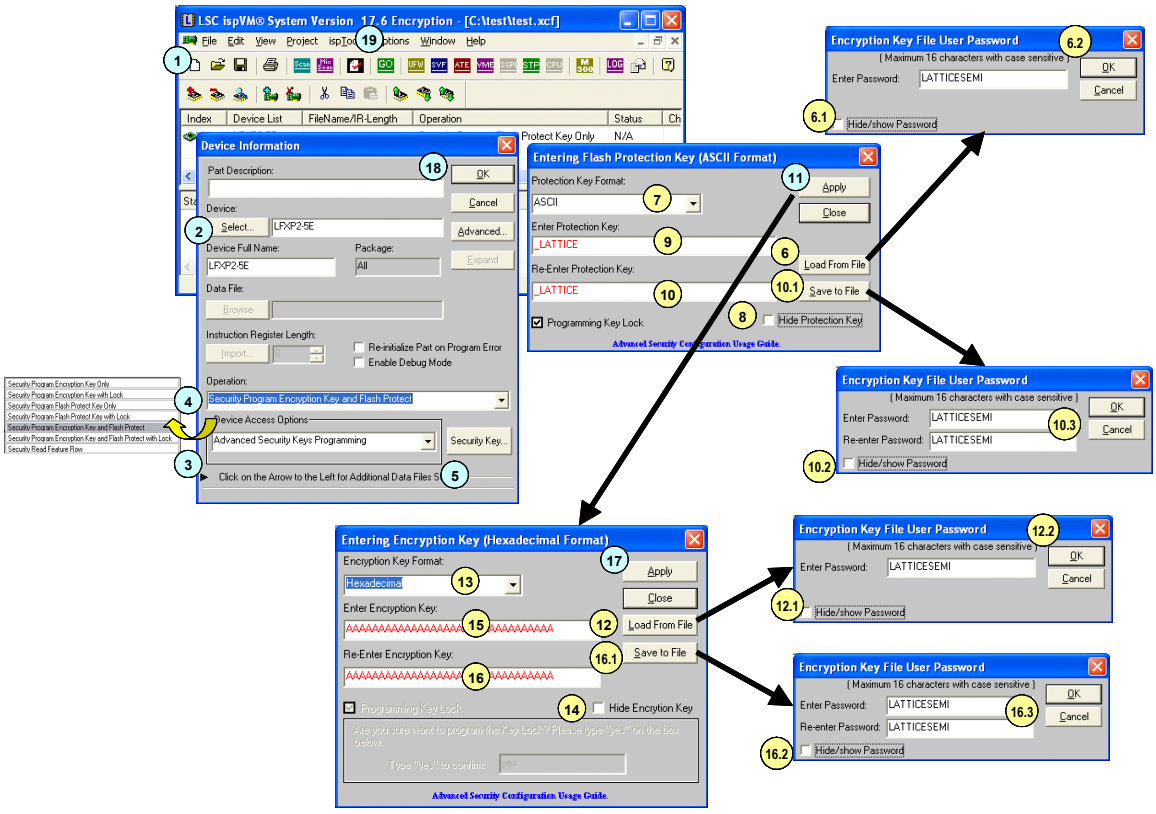


Table 8. Procedure for Erasing the Feature Row

Steps	Description	
1	Launch the device selection menu.	
2	Select the LatticeXP2 device.	
3	Select Advanced Security Keys Programming under Device Access Options.	
4	Select Security Erase Feature Row under Operation (or click on Security Key button) to launch the Erase Feature Row Options dialog.	
5	There are three options you can choose:	
	Options	**Comments**
5.1	The Feature Row locked using the Encryption Key. Select this option if the Feature Row was locked using the Encryption Key option selected.	Unlock the Feature row using the 128-bit AES encryption key as the password. This option should be selected after the Feature Row programmed with: **Security Program Encryption Key with Lock** or **Security Program Encryption Key and Flash Protect with Lock** operations.
5.2	The Feature Row locked using the Flash Protection Key. Select this option if the Feature Row locked using the Flash Protection Key option selected.	Unlock the Feature row using the 64-bit Flash Protect key as the password. This option should be selected after the Feature Row programmed with: **Security Program Flash Protect Key with Lock** operation.
5.3	Did not use the Feature Row Lock. Select this option if the Feature Row has not been locked.	Does not need to unlock the Feature Row. This option should be selected after the Feature Row programmed with: **Security Program Flash Protect Key Only** or **Security Program Encryption Key Only** or **Security Program Encryption Key and Flash Protect** operations.
6	Click on OK button.	
6.1	The Entering Encryption Key dialog will pop up. Repeat step 12 to step 16 of the Table 7.	
6.2	The Entering Flash Protection Key dialog will pop up. Repeat step 6 to step 10 of the Table 7.	
7	Click OK to come back to the ispVM Windows. ispVM generates a chain description file, which can be saved as an .xcf file.	
8	Select the GO button to erase the Feature Row. For embedded programming, skip this step.	

Figure 14. Using ispVM to Erase the Feature Row

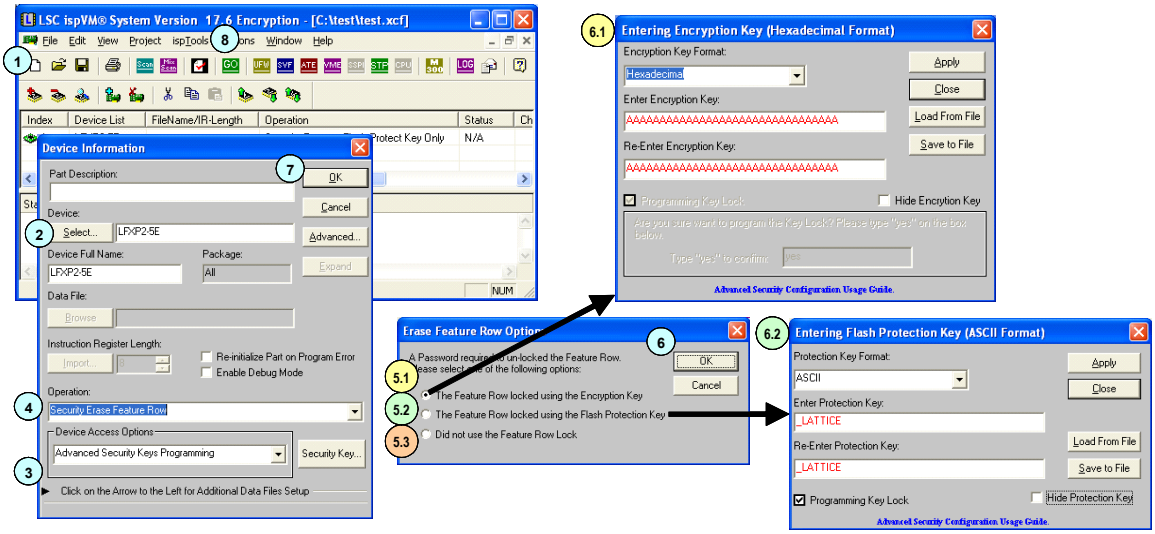


Table 9. Procedure to Create an Encrypted JEDEC File and Key File using ispUFW

Steps	Description
1	Launch the UFW on ispVM.
2	Select the JEDEC file as the output format.
3	Double click on the Input Data File and browse to the unencrypted JEDEC file. The device name will be extracted from the header of the JEDEC file automatically. **Note:** If the ispVM version is Encryption Enabled and the device name is the LatticeXP2 family, the Encryption option will show up automatically for step 5. Otherwise it will not be available.
4	Double click on the Output Data File and browse to or enter the output file name.
5	Right click on the Encryption option and set the Encryption to ON. **Note:** This option is only available in the encryption patch is installed, and is only available for JEDEC files for the LatticeXP2 family.
6	Right click on the Encryption Key option and click Edit Encryption Key to launch the Encryption Key setup dialog.
7	You can load the Encryption key from an existing .bek file by clicking the Load From File button.
7.1	This step is optional. You can unselect the Hide Password option to view the Password in the GUI.
7.2	Enter the key file password and click OK, and jump to step 11. This step is optional if you used the default Password.
	Enter the Encryption Key Manually into the GUI
8	Select the Encryption Key format under Encryption Key Format. If you want to use the Hexadecimal format, select the Hexadecimal.
9	This step is optional. You can unselect the Hide Encryption Key option to view the Key in the GUI.
10	Enter the Encryption Key. If enter less than the full Key, then fillers will be padded on the left (the most significant) position. If enter more, then the GUI will block the excessive entry. Command line will truncate the overflow.
11	Click the OK button. The prompt to save the Encryption file into a .BEK file will show.
12	Enter the .BEK file name. This step is mandatory due to an encrypted JEDEC must have some record of the Encryption Key used.
13	Click the Save button, and the prompt to enter the Encryption File Password dialog will open.
13.1	This step is optional. You can unselect the Hide Password option to view the Password in the GUI.
14	Enter the password. The password is restricted to ASCII characters only. If enter less than 16 characters, the fillers will be padded on the left position. If enter more, then the excessive entry will be blocked. his step is necessary to protect the .BEK file with a password.
15	Click the OK button to write the .BEK file.
16	Click the file generation button to generate the encrypted JEDEC file.
Notes	Comments
D1	This option is provided to overwrite the USERCODE with the file checksum.

Figure 15. Procedure for Encrypting JEDEC Files using ispUFW

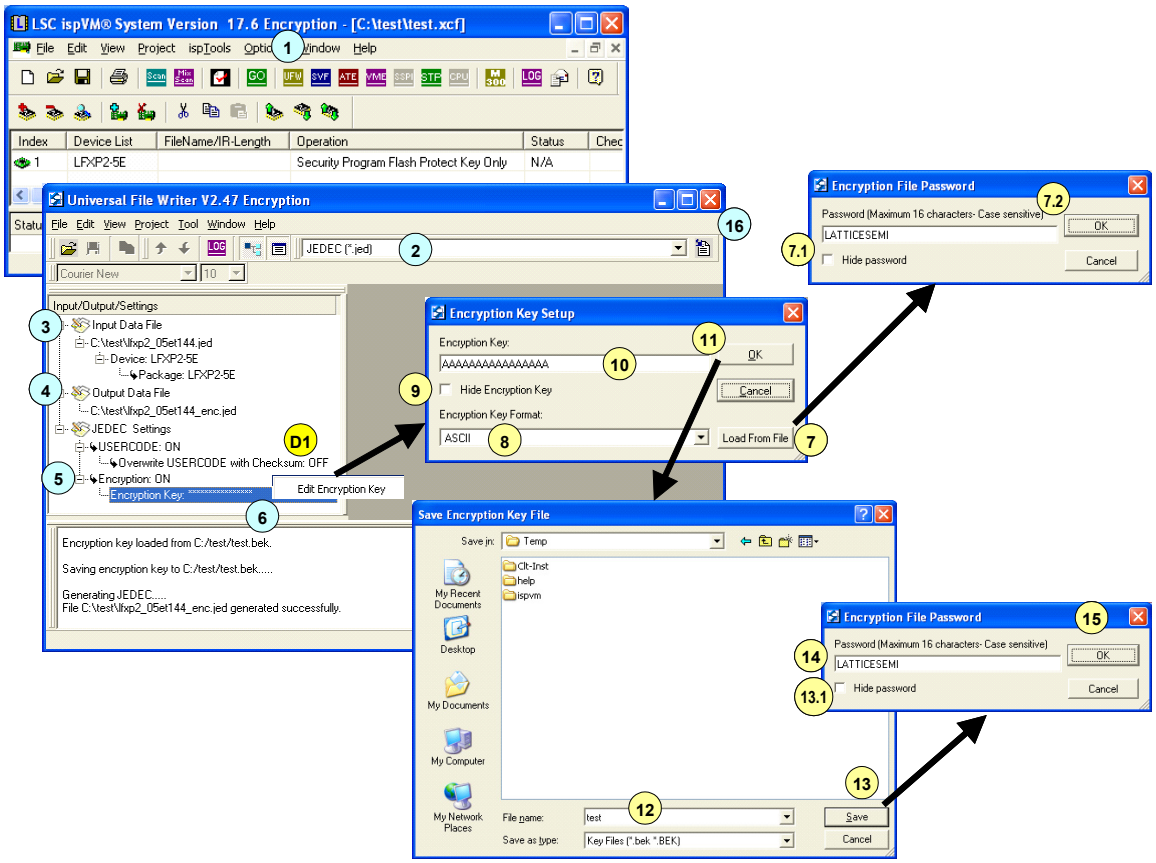


Table 10. Procedure for Program a Device with the Flash Protect Key Enabled using ispVM

Steps	Description
1	Launch the device selection menu.
	Select the LatticeXP2 device.
2	Browse for an **unencrypted JEDEC file**.
3	Select the Advanced Security Encryption File Programming under the Device Access Options.

Table 10. Procedure for Program a Device with the Flash Protect Key Enabled using ispVM (Continued)

Steps	Description	
4	Select the Security EPV with Flash Protect operation from the operation list. Below are the list of operations can be used with this flow.	
	Operations	**Comments**
	Security EPV with Flash Protect	Verify the Flash Protect password then Erase, Program, and Verify the device with an unencrypted JEDEC file.
	Security EPVS with Flash Protect	Verify the Flash Protect password then Erase, Program, Verify, and Secure the device with an unencrypted JEDEC file.
	Security Background EPV with Flash Protect	Verify the Flash Protect password then Erase, Program, and Verify the device with an unencrypted JEDEC file in background mode.
	Security Background EPVS with Flash Protect	Verify the Flash Protect password then Erase, Program, Verify, and Secure the device with an unencrypted JEDEC file in background mode.
	Security Flash Erase with Flash Protect	Verify the Flash Protect password then Erase the device.
	Security Background Erase with Flash Protect	Verify the Flash Protect password then Erase the device in background mode.
5	The Flash Protect Key dialog will automatically open after the operation is selected. It can also be opened by clicking on the Protection Key button.	
6	You can load the Flash Protect Key from an existing .key file by clicking the Load From File button	
6.1	This step is optional. You can unselect the Hide Password option to view the Password in the GUI.	
6.2	Enter the key file password and click OK, and jump to step 11. This step is optional if you used the default Password.	
	Enter the Flash Protect Key Manually into the GUI	
7	Select the Key format under Protection Key Format. If you want to use the Hexadecimal format, select the Hexadecimal.	
8	This step is optional. You can unselect the Hide Protection Key option to view the Key in the GUI.	
9	Enter the Flash Protect Key. If enter less than the full Key, then fillers will be padded on the left (the most significant) position. If enter more, then the GUI will block the excessive entry.	
10	Re-Enter the Flash Protect Key.	
10.1	This step is optional. You can save the Flash Protect Key to a .key file by clicking the Save To File button. The software will ask for a key file name, enter the name. This option is only available only if the ispVM System encryption patch is installed.	
10.2	This step is optional. You can unselect the Hide Password option to view the Password in the GUI.	
10.3	Enter the key file password and click OK.	
11	Click Apply to come back to the Device Information Dialog.	
12	Click OK to come back to the ispVM Windows. ispVM generates a chain description file, which can be saved as an .xcf file which will contain the Flash Protect Key.	
13	Select the GO button to program the Flash Protect Key into the device. For embedded programming, skip this step.	

Figure 16. Using the ispVM to Program an Unencrypted JEDEC File into the Device

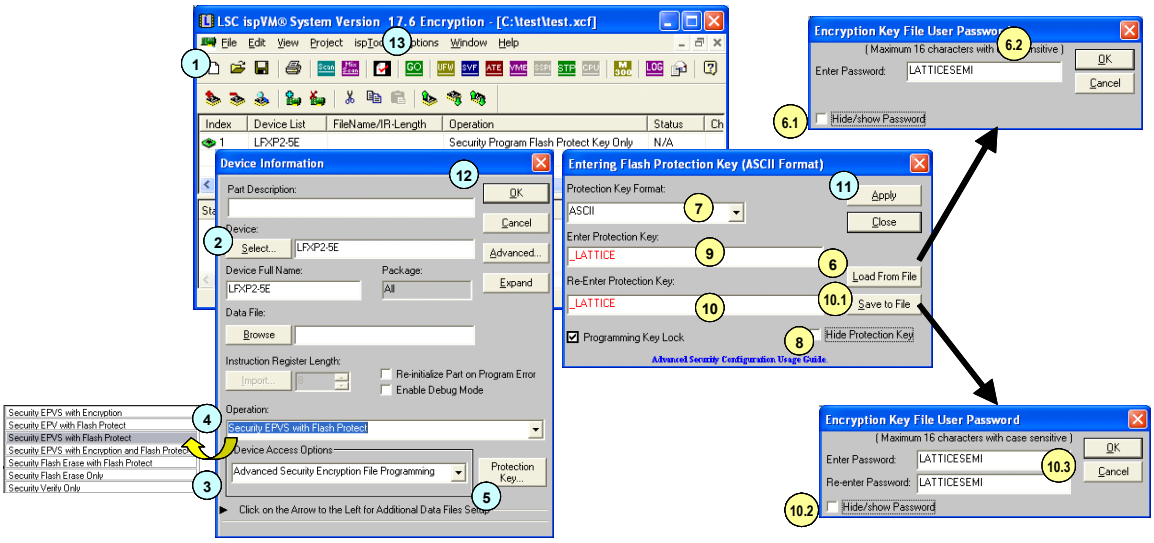
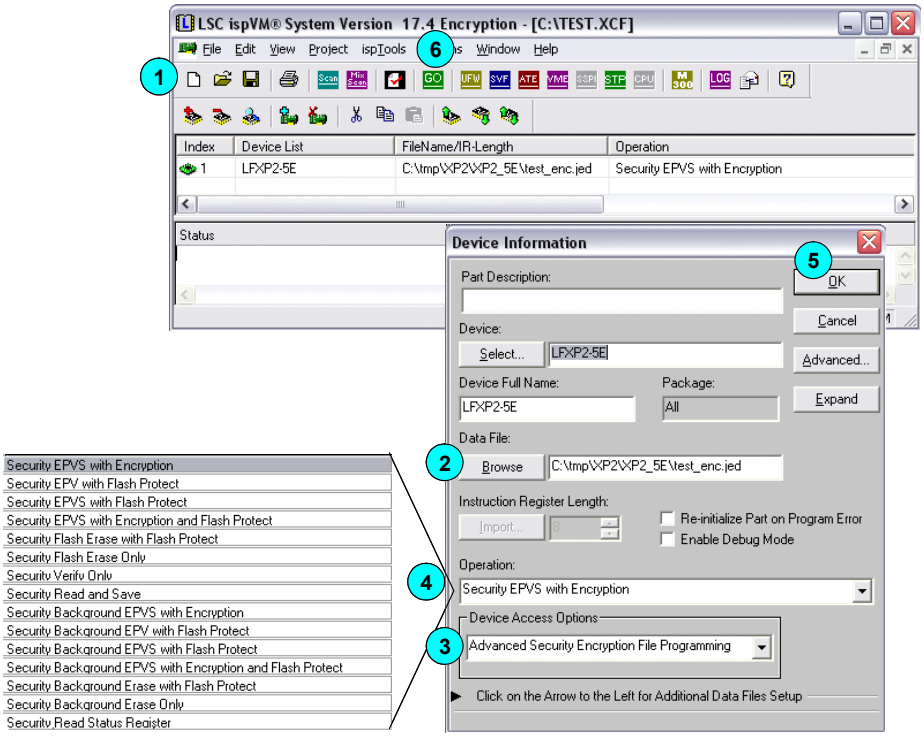


Table 11. Procedure for Program an encrypted JEDEC file into the Device using ispVM

Steps	Description	
1	Scan or select the LatticeXP2 device.	
2	Browse to and select the **encrypted JEDEC file**.	
3	Select the Advanced Security Encryption File Programming under the Device Access Options.	
4	Select the Security EPVS with Encryption operation from the operation list. Below are the list of operations can be used with this flow	
	Operations	**Comments**
	Security EPVS with Encryption	Erase, Program, Verify, and Secure the device with the encrypted JEDEC file.
	Security EPVS with Encryption and Flash Protect	Verify the Flash Protect password then Erase, Program, Verify, and Secure the device with the encrypted JEDEC file.
	Security Background EPVS with Encryption	Erase, Program, Verify, and Secure the device with the encrypted JEDEC file in background mode
	Security Background EPVS with Encryption and Flash Protect	Verify the Flash Protect password then Erase, Program, Verify, and Secure the device with the encrypted JEDEC file in background mode.
5	Click OK to close device setup dialog	
6	Click GO to program the device. For embedded programming, skip this step.	

Figure 17. Using the ispVM to Program an Encrypted JEDEC File into the Device



Manufacturing using ispVM System

Table 12. Procedure for Manufacturing LatticeXP2 Encryption Programming

Steps	Description	
1	Using ispVM (Figure 18): Launch the device selection menu. Using Model300 (Figure 19): Launch the Model300.	
2	Select the LatticeXP2 device.	
3	Browse to and select the **encrypted JEDEC file**.	
4	Select Advanced Security Production Programming under Device Access Options.	
5	Select Security EPVS with full feature Option under Operation. Below are the list of operations can be used with this flow.	
	Operations	Comments
	Security EPVS with full feature Option	1.) Program the Flash Protect Key and 128-bit AES Key into the device. 2.) Verify the Flash Protect password then Erase, Program, Verify, and Secure the device with the encrypted JEDEC file.
	Security EPVS with full feature and Lock Option	1.) Program the Flash Protect Key, the 128-bit AES Encryption Key, and the Feature Lock Enable. 2.) Verify the Flash Protect password then Erase, Program, Verify, and Secure the device with the encrypted JEDEC file.
	Security EPVS with Encrypt Only Option	1.) Program the 128-bit AES Key into the device. 2.) Erase, Program, Verify, and Secure the device with the encrypted JEDEC file.
	Security EPVS with Encrypt and Lock Option	1.) Program the 128-bit AES Encryption Key and the Feature Lock Enable. 2.) Erase, Program, Verify, and Secure the device with the encrypted JEDEC file.
	Security EPV with Protect Only Option	1.) Program the Flash Protect Key into the device. 2.) Verify the Flash Protect password then Erase, Program, and Verify the device with a regular JEDEC file.
	Security EPV with Protect and Lock Option	1.) Program the Flash Protect Key and the Feature Lock Enable. 2.) Verify the Flash Protect password then Erase, Program, and Verify the device with a regular JEDEC file.
	Security EPVS with Protect Only Option	1.) Program the Flash Protect Key into the device. 2.) Verify the Flash Protect password then Erase, Program, Verify, and Secure the device with a regular JEDEC file.
	Security EPVS with Protect and Lock Option	1.) Program the Flash Protect Key and the Feature Lock Enable. 2.) Verify the Flash Protect password then Erase, Program, Verify, and Secure the device with a regular JEDEC file.
6	The Flash Protect Key dialog will automatically open after the operation is selected. It can also be opened by clicking on the Protection Key button.	
7	You can load the Flash Protect Key from an existing .key file by clicking the Load From File button.	
7.1	This step is optional. You can unselect the Hide Password option to view the Password in the GUI.	

Table 12. Procedure for Manufacturing LatticeXP2 Encryption Programming (Continued)

Steps	Description
7.2	Enter the key file password and click OK, and jump to step 12. This step is optional if you used the default Password.
	Enter the Flash Protect Key Manually into the GUI
8	Select the Key format under Protection Key Format. If you want to use the Hexadecimal format, select the Hexadecimal.
9	This step is optional. You can unselect the Hide Protection Key option to view the Key in the GUI.
10	Enter the Flash Protect Key. If enter less than the full Key, then fillers will be padded on the left (the most significant) position. If enter more, then the GUI will block the excessive entry.
11	Re-Enter the Flash Protect Key.
11.1	This step is optional. You can save the Flash Protect Key to a .key file by clicking the Save To File button. The software will ask for a key file name, enter the name. This option is only available only if the ispVM System encryption patch is installed.
11.2	This step is optional. You can unselect the Hide Password option to view the Password in the GUI.
11.3	Enter the key file password and click OK.
12	Click Apply and the Encryption Key dialog will automatically open.
13	You can load the Encryption key from an existing .bek file by clicking the Load From File button.
13.1	This step is optional. You can unselect the Hide Password option to view the Password in the GUI.
13.2	Enter the key file password and click OK, and jump to step 18. This step is optional if you used the default Password.
	Enter the Encryption Key Manually into the GUI
14	Select the Key format under Encryption Key Format. If you want to use the Hexadecimal format, select the Hexadecimal.
15	This step is optional. You can unselect the Hide Encryption Key option to view the Key in the GUI.
16	Enter the Encryption Key. If enter less than the full Key, then fillers will be padded on the left (the most significant) position. If enter more, then the GUI will block the excessive entry.
17	Re-Enter the Encryption Key.
17.1	This step is optional. You can save the Encryption Key to a .bek file by clicking the Save To File button. The software will ask for a key file name, enter the name.
17.2	This step is optional. You can unselect the Hide Password option to view the Password in the GUI.
17.3	Enter the key file password and click OK.
18	Click Apply to come back to the Device Information Dialog.
19	Click OK to come back to the ispVM Windows. ispVM generates a chain description file, which can be saved as an .xcf file which will contain the Encryption Key.
20	Select the GO button to program the Encryption Key into the device. For embedded programming, skip this step.

Figure 18. Using the ispVM for Manufacturing Encryption Programming

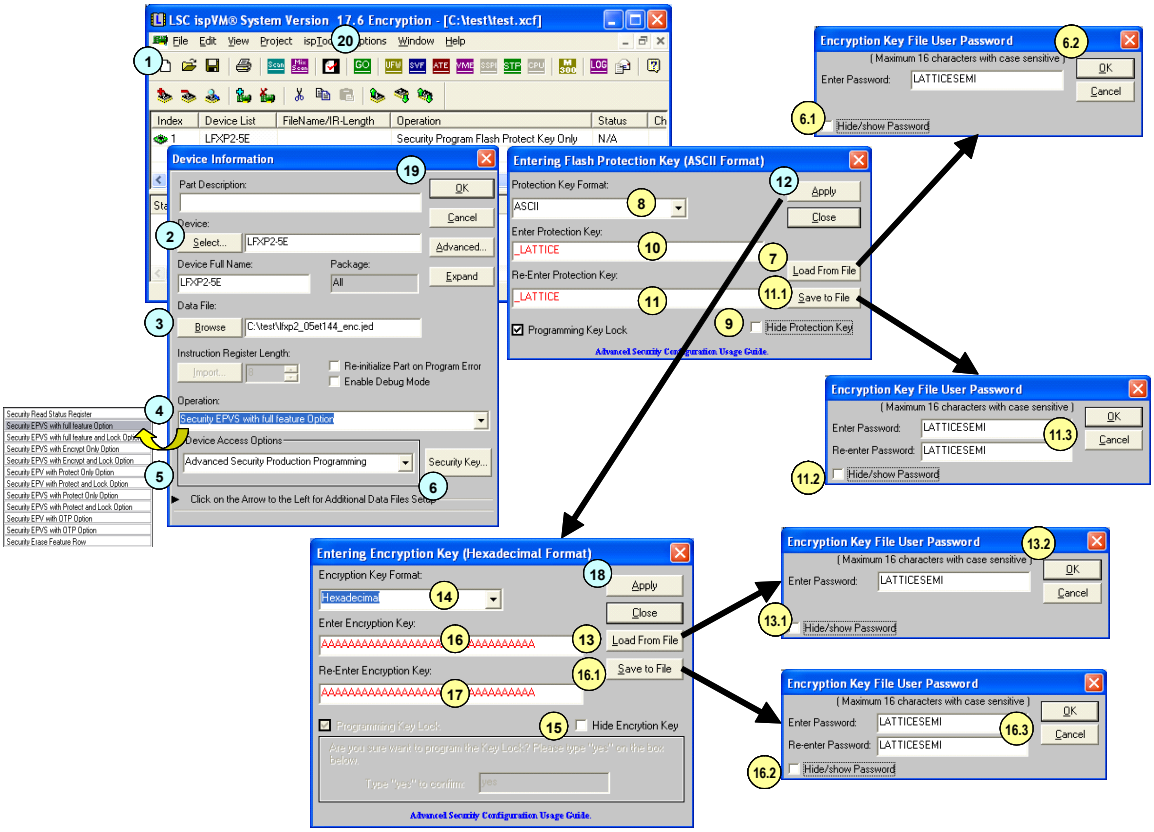


Figure 19. Using the Model300 for Manufacturing Encryption Programming

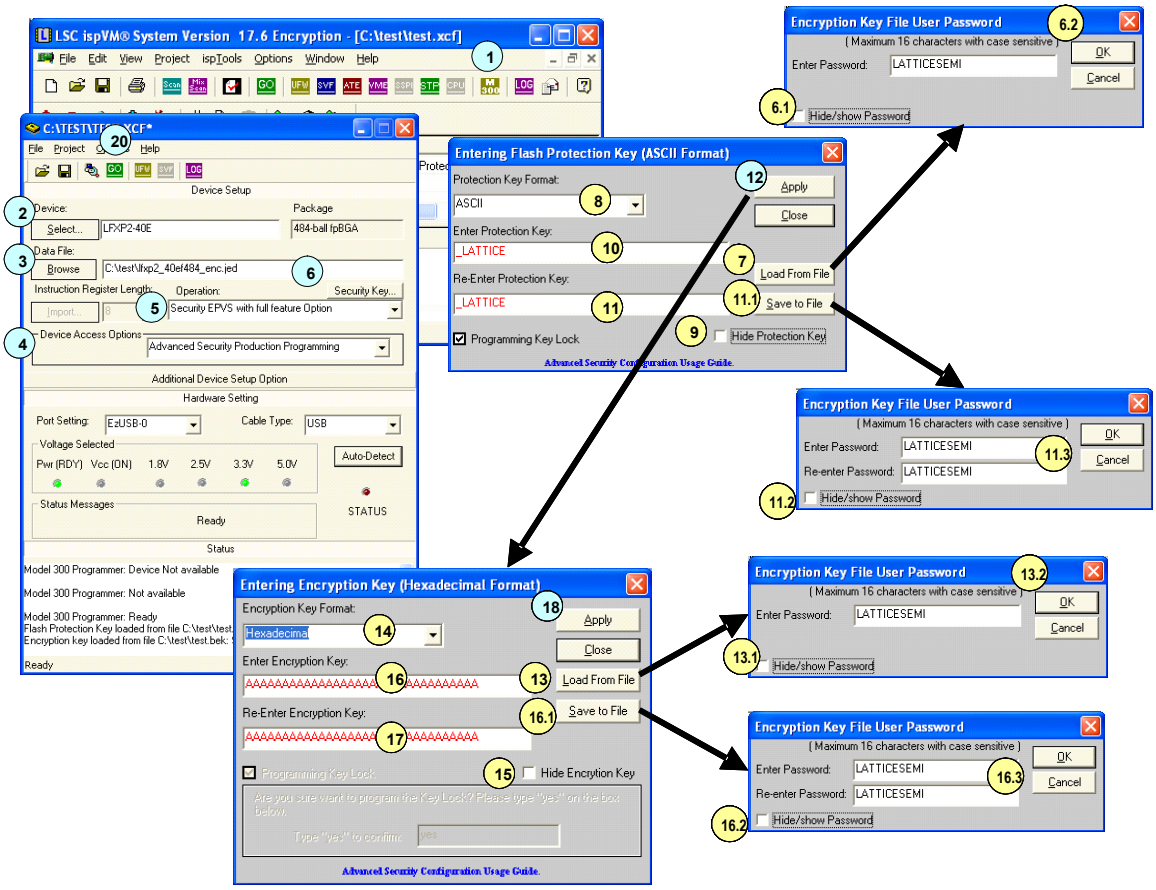
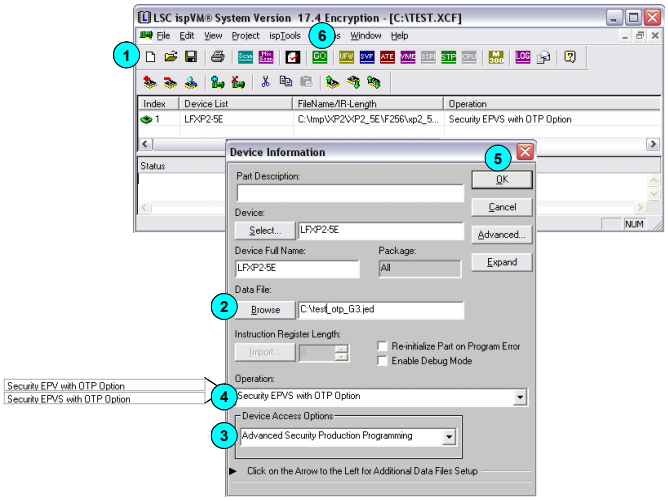


Table 13. Procedure for Program OTP Fuses using ispVM System

Steps	Description	
1	Scan or select the LatticeXP2 device.	
2	Browse for the JEDEC file with OTP option enabled (the G field equal to G2 or G3).	
3	Select the Advanced Security Encryption File Programming under the Device Access Options.	
4	Select the Security EPV with OTP Option operation from the operation list. Below are the list of operations can be used with this flow.	
	Operations	**Comments**
	Security EPV with OTP Option	Program the OTP fuses. Erase, Program, and Verify the device with a regular JEDEC file with the G field = 2.
	Security EPVS with OTP Option	Program the OTP fuses. Erase, Program, Verify, and Secure the device with a regular JEDEC file with the G field = 3.
5	Click OK to close device setup dialog.	
6	Click GO to program the device. For embedded programming, skip this step.	

Figure 20. Using the ispVM to Program OTP Fuses



Advanced Key Programming

Table 14. Procedure for Encryption Key Serialization Programming using ispVM System

Steps	Description
1	Repeat from step 1 to step 17 of the Table 12.
2	Save the chain setup XCF file contains: LatticeXP2 Devices The location of the encrypted JEDEC File The Security EPVS with full feature Option operation The original Encryption Keys (and /or Flash Protect Key)
3	Using a script file or the DOS mode to launch ispVM command line while entering the serialization keys Usage: **-encryption:** specify the advanced security options. [**-key:** specify the encryption key] [**-hex:** specify the encryption key in hex. The “hex_key” must be provided in hex and contain at most 32 characters] [**-ascii:** specify the encryption key in ASCII. The “ascii_key” must contain at most 16 characters] [**-protect:** specify the Flash protect key] [**-hex:** specify the Flash protect key in hex. 'hex_key' must be provided in hex and contain at most 16 characters] [**-ascii:** specify the Flash protect key in ASCII. 'ascii_key' must contain at most 8 characters] Example: ispVM.exe -infile c:\test.xcf -encryption -key -hex "000000004C6174746963652053656D69” -protect -ascii "_LATTICE” -o ispVM.exe -infile c:\test.xcf -encryption -protect -ascii "_LATTICE” -o
4	The software will first program the new input key and next program the encrypted JEDEC file using the Security EPVS with full feature Option operation.

Table 15. Procedure for Generating the Encryption Bitstream with Serialized Decription Key Using ispUVW

Steps	Description
1	Using a script file or the DOS mode to launch ispUFW command line while entering the serialization keys Usage: **-encryption:** Encrypt file. **-key:** Specify the encryption key **-hex:** Specify the encryption key in hex. The “hex_key” must be provided in hex and contain at most 32 characters **-ascii:** Specify the encryption key in ASCII. The “ascii_key” must contain at most 16 characters **-config_mode:** Specify the configuration mode of the encrypted bitstream. Possible values are “jtag burst”, “spi”, “spim”, “slave_scm”, and “slave_pcm”] Example: ispUFW.exe -device LFXP2-5E -infile "xp2-5e.jed" -encryption -key –hex "000000004C6174746963652053656D69" -oft -jed -outfile “xp2-5e_encryption.jed" Note: This option only available on the ENCRYPTION version. This option only supports the LatticeXP2 devices family.

LatticeXP2 Encrypted JEDEC File

Frame and Data CRC

When a LatticeXP2 JEDEC file is encrypted, two 16-bit CRC’s are calculate, a Frame and a Data CRC.

- Frame CRC
  The Frame CRC is calculated using only the first frame (row) of the JEDEC file. The purpose of the Frame CRC is to validate that the incoming JEDEC file was encrypted with the same key as is programmed into the device. If the keys did not match, this first CRC will fail, and the programming operation will stop to prevent an invalid pattern from being programmed into the device. If the Frame CRC passes, we know that the JEDEC is encrypted with the same key.
- Data CRC
  The Data CRC is calculated using all of the frames (rows) of the JEDEC file. The purpose of the Data CRC is to confirm that the device received the data correctly. If the Data CRC failed, that means the device did not receive the data correctly. A Data CRC failure could be due to noise or timing issues. If the device fails the Data CRC check, the Done fuse will not be programmed, which will prevent the device from configuring with an invalid pattern.

USERCODE

When a LatticeXP2 JEDEC file is encrypted, the USERCODE (U field) in the JEDEC file is used to store the Frame and Data CRC’s, as shown below.

UH*xxxxyyyy**
Where:

xxxx = Frame CRC

yyyy = Data CRC

The original USERCODE is stored as a comment in the Note section in the JEDEC file. An example is shown below.

```
NOTE USERCODE: 0x73686F74
```

References

- Lattice Technical Note TN1213, LatticeXP2 Slave SPI Port Usage Guide
- Lattice Technical Note TN1141, LatticeXP2 sysCONFIG Usage Guide
- Lattice Technical Note TN1142, LatticeXP2 Configuration Encryption and Security Usage Guide
- Federal Information Processing Standard Publication 197, Nov. 26, 2001. Advanced Encryption Standard (AES)

Technical Support Assistance

Hotline:1-800-LATTICE (North America)
+1-503-268-8001 (Outside North America)
e-mail: techsupport@latticesemi.com
Internet: www.latticesemi.com

Revision History

Date	Version	Change Summary
November 2010	01.0	Initial release.