Cryptographic Rights Management of FPGA Intellectual Property Cores

Tom Kean
Algotronix Ltd.
PO Box 23116
Edinburgh EH8 8YB United Kingdom

tom@algotronix.com

ABSTRACT

As the capacity of FPGA's increases to millions of equivalent gates the use of Intellectual Property (IP) cores becomes increasingly important to control design complexity. FPGA's are becoming platforms for integrating a system solution from components supplied by independent vendors in the same way as printed circuit boards provided a platform for earlier generations of designers. However, the current commercial model for IP cores involves large up-front license fees reminiscent of ASIC NRE charges. In order to match the IP core business model to the low to medium volume applications addressed by FPGA customers it is important to develop cryptographic techniques which allow IP core vendors to sell their product on a pay-per-use basis rather than through up-front license fees.

General Terms

Algorithms, Design, Economics, Security, Legal Aspects.

Keywords

FPGA, Intellectual Property, Cryptography, Rights Management.

1. INTRODUCTION

Improvements in process technology and device architecture have allowed very large circuits to be implemented on FPGA's. FPGA's can now implement designs with the equivalent of several million gates of logic and medium sized memory blocks. FPGA's with on board micro-controllers, implemented either directly in silicon or on the FPGA resources are available. FPGA's and Configurable System on Chip (CSoC) parts from companies such as Xilinx, Altera and Triscend are becoming 'platforms' for implementing entire systems.

The trend to implement entire systems on an FPGA is creating a market for intellectual property 'cores'. These are designs created by third parties which are sold to FPGA users to incorporate in larger systems. Examples of cores include bus inter-faces such as PCI bus, signal processing functions such as Reed Solomon Decoders and communications interface functions such as Serialiser / Deserialiser (SERDES). Leading FPGA manufacturers offer access to a catalogue of cores and customers expect to be able to create a large part of the functionality of their system using cores – thus reducing their time to market and engineering effort.

An important difficulty for the FPGA core industry is that there is no way for a core vendor to monitor how many times their core has been configured into FPGA's by a particular customer. For this reason it normal for third-party core vendors to have a one time 'license' charge to access the design files rather than a 'per use' charge. This is an undesirable business model since it means that a customer with a low-volume application must pay the same license fee as a customer who will sell millions of units. Further, customers have to pay the entire license fee 'up front' long before obtaining revenue from product sales. Customers might be willing to pay much more for intellectual property if the charges were proportionate to their own sales rather than a fixed up-front charge.

In order to make a return on the engineering time invested the core vendors are forced to charge high fees to access the core – which has the effect of pricing the core beyond the reach of users with low volume applications. Unfortunately, FPGA's have the greatest market advantage over mask-programmed Application Specific Integrated Circuits (ASIC's) in low and medium volume applications. By selecting an FPGA instead of a mask programmed ASIC customers have decided to pay higher per-chip costs in exchange for avoiding up-front NRE charges.

As process technology improves mask programmed chips implementing medium complexity functions become pad limited and no longer offer a cost advantage over programmable solutions. This should allow FPGA's to take over the market for many categories of medium complexity Application Specific Standard Product (ASSP) chips – such as PCI interfaces. However ASSP vendors are reliant on ASIC implementations to support their pay-per-use business model. Cryptographic rights management technology for FPGA's would allow ASSP vendors to address medium complexity functions more economically by supplying bitstreams for FPGA's (creating Virtual ASSP's). Customers who purchased a complete FPGA bitstream would not

FPGA 2002, February 24-26, 2002, Monterey, CA, USA.

require any FPGA design tools or experience allowing the FPGA vendors to sell to the large community of engineers who work at the board level with catalogue parts.

The poor match between the up-front license fee model and the FPGA customer base has deterred many companies with significant IP from making it available to FPGA users. At present, many of the available cores are supplied by the FPGA vendors – either free of charge or for nominal fees – in order to stimulate chip sales. As FPGA chip sizes continue to increase it will become impossible for FPGA vendors to provide all the necessary cores. It is in everyone's interest: FPGA customers, FPGA vendors and third party IP suppliers to find a business model by which core vendors can receive 'per-use' payments for their intellectual property in order to create a viable market for IP cores.

Previous work has considered cryptographic schemes for preventing piracy and reverse engineering of FPGA bitstreams [1-7] but this is believed to be the first proposal for a scheme to secure pay-per-use licensing of FPGA IP cores.

2. PARTIES TO THE FPGA IP TRANSACTION

Before discussing a cryptographic scheme to manage IP rights it is helpful to clearly define the various parties involved. The goal of a cryptographic rights management scheme is to create a framework in which the actions of various parties can be controlled in order to create business models better matched to market needs.

The '**End User**'– purchases equipment containing FPGA's. The end user may become a participant in the licensing process if the equipment allows downloading new designs into the FPGA after the equipment is delivered to the user. Other parties in the process may wish to limit the end users ability to 'clone' equipment by copying the FPGA bitstream file or to replace the FPGA design with one which changes the equipment's functionality. For example, in the case of a cellular telephone containing an FPGA the user might wish to reconfigure the FPGA to avoid service charges.

The '**FPGA Customer**' – manufactures equipment which contains FPGA's. To do this the customer requires bitstream files for 'user designs' which when loaded onto the FPGA cause it to perform the desired functions in the equipment. These 'user designs' may include intellectual property blocks or 'cores' which implement a portion of the required function.

The '**Designer**' – creates a complete design for an FPGA chip. The design may make use of one or more 'IP cores' purchased from Core vendors or obtained from the FPGA vendor. The design can be converted into a bitstream file for the FPGA chip using the FPGA vendor's implementation software. Often the 'Designer' and 'Customer' are the same organization but there is no reason why this should always be the case and so for the purpose of describing the protocols it is helpful to separate the two roles.

The '**Core Vendor**' – designs intellectual property cores for resale. These cores may be provided as Hardware Description Language (HDL) files, or in another suitable format such as a netlist for a particular FPGA manufacturer's Computer Aided Design (CAD) tools. Core vendors normally also supply substantial documentation and test sets for their design. Core vendors may sell direct to the 'Designer' or may have a marketing agreement for the FPGA vendor to distribute their product. FPGA vendors also generally provide Intellectual Property (IP) cores which can be incorporated in customer designs for their chips. These include simple functions (such as adders and multipliers) which are generally provided free of charge and more complex functions (such as PCI bus interfaces) which are provided for a fee. In some cases FPGA vendors license and resell cores from third party core vendors.

The '**FPGA Vendor**' – designs and manufactures FPGA chips.

The '**CAD Software Vendor**' – designs and sells CAD software tools. These include 'Implementation Tools' which map netlists describing user designs into bitstreams which can program FPGA's. Implementation tools include place and route and bitstream generation tools. The complete design flow also requires higher level synthesis and simulation tools. Today, implementation tools for a given FPGA are generally only available from the FPGA vendor. There is a general trend for FPGA companies to provide more and more of the complete tool flow. For the purposes of our model we have separated the functions of software vendor and FPGA vendor because marketplace dynamics may well force a separation of the functions in the future.

The '**Trusted External Party (TEP)**' – or 'trusted third party' is an organization which all parties to the transaction are willing to trust to behave fairly. The TEP has the role of facilitating the IP transaction by maintaining various secure databases and managing billing for accessing the IP. Trusted third parties are common in cryptographic protocols – for example certification authorities sign public keys issued by websites to indicate that the organization issuing the key is actually entitled to use it. As an example when one visits Amazon's website and sets up a secure connection to transfer a credit card number it is a Certification Authority such as Verisign corporation that guarantees that the public key obtained from the website which purports to be that of amazon.com actually is from amazon.com.

In the context of FPGA's the FPGA vendor is likely to act as the trusted external party since they have existing business relationships with all the parties to the transaction. However, since the trusted external party role is a distinct one and need not be fulfilled by the FPGA vendor it makes sense to describe the protocols as if the trusted third party is a separate organization.

3. THE RIGHTS MANAGEMENT PROTOCOL

The proposed rights management protocol has three phases: these occur when the FPGA is manufactured, when the designer incorporates cores into his design and during the manufacture of equipment containing FPGA chips. The protocol is dependent on additional cryptographic circuitry installed in each FPGA chip. This circuitry is a simple extension of that proposed by Algotronix to protect FPGA designs from piracy and reverse engineering [5], [6], [7]. The protocol described here also depends on a server computer on the internet operated by a Trusted External Party which maintains various databases and administrates charging for IP cores.

3.1 Creation of Security Information by FPGA Manufacturer

Algotronix has previously described details of a scheme to prevent 'cloning' and reverse engineering of FPGA bitstream information. This scheme is based on a secret key stored permanently on each FPGA chip. Many techniques are available for embedding the key on the chip for example, laser programming of fuses, embedding the key information in the device maskwork and use of antifuses or FLASH memory. A feature of the Algotronix anti-piracy scheme is that the secret key stored on the chip need not be known to anyone - including the FPGA manufacturer. In the present proposal, as well as containing a secret cryptographic key, known only to itself, each chip contains a unique ID or serial number. This ID is not secret and is made available on request through the programming interface.

In order to extend the scheme to support secure download and protection of IP cores it is desired to maintain a shared secret known only to the chip and an external party. This shared secret can be used to create a secure communications channel to the chip. Normally the shared secret is a cryptographic key which is used to encrypt messages using a symmetric cipher such as triple DES or AES.

An entity (for example a person or organization) who has physical possession of the chip or equipment containing the chip can then create a shared secret 'token' as follows:

1. The entity chooses a secret 'user' key and presents it to the chip through the programming interface (e.g. via JTAG) with header information indicating that this is a secret key.

2. The chip creates a random initial value (IV) using an on chip random number generator.

3. The chip encrypts the supplied secret key using its own on-chip secret key in cipher block chaining (CBC) mode using the initial value created in step two. In CBC mode the IV is exored with the first item of secret information prior to encryption. CBC mode encryption protects against chosen plaintext attacks and pat-tern analysis of the ciphertext [8].

4. The chip reports its identifier and the CBC encrypted data including the final checksum through the programming interface. This constitutes a 'token' (figure 1) which may be freely distributed and need not be kept secret but which, when presented to the FPGA at a later time allows the FPGA to determine the secret key and communicate securely with the organization who created the token.

5. The entity stores the token and the corresponding user key in a database indexed by the chip identifier. Using this database the organization can find the appropriate token and user key for a given chip at a later time.

Using this scheme anyone who has access to the FPGA can create a 'token' allowing secure communication with the FPGA once it leaves their possession. Since tokens are not stored on the chip itself there is no limit to the number of tokens that may be associated with a given chip. Tokens can be distributed freely over unencrypted links and stored in non-volatile memory along with FPGA bitstreams. When the FPGA that created the token loads it in it can recover the secret user key information and communicate securely with the entity that asked it to create the token.

As well as their use for IP rights management tokens may be used to support secure download of programming information to the FPGA in the field. In this case the tokens are created by the manufacturer of the equipment containing the FPGA prior to shipping the equipment to end-customers. The token is stored in non-volatile memory in the equipment containing the FPGA along with the original bitstream. When a field-download is to be initiated the FPGA provides its ID to the server computer operated by the equipment manufacturer. The manufacturer then encrypts the bit-stream to be downloaded with the user key corresponding to the ID and supplies the bitstream to the FPGA. The FPGA decrypts the token using its on-chip secret key to extract the user key necessary to decrypt the bitstream information. The transfer of the FPGA ID to the manufacturer server and the download of the bitstream information can take place over insecure communication channels.

For the purposes of enforcing IP licensing schemes it is attractive to create security tokens for all FPGA's at the time of manufacture. This can be done during final testing of the FPGA's at minimal cost.

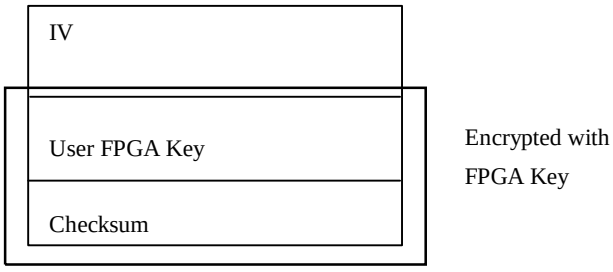

Figure 1. Cryptographic Token.

3.2 Use of IP by FPGA Designer

Where a user design incorporates one or more IP cores problems arise in protecting intellectual property. The user will run CAD tools provided by the FPGA vendor to create a bitstream from high level design files. However, core vendors may not wish to provide complete design files to the user – since this would allow the user to modify the vendor's core and circumvent any copy

protection mechanisms. Instead core vendors may choose to supply encrypted cores which can be processed by CAD tools but cannot be easily viewed or modified by the user.

For example, Altera provides encrypted cores for evaluation purposes, these cores can be simulated to make sure they have the desired functionality but they cannot be used to generate bitstreams and the source code cannot be viewed. Once the designer buys a license to use the core the unencrypted data is provided [9]. Several large CAD companies and FPGA vendors have developed similar technologies to support evaluation of IP cores.

In the proposed IP rights management scheme (figure 2), instead of refusing to generate bitstreams for designs that contain 'encrypted' IP cores the FPGA vendor CAD software creates special encrypted bitstreams (figure 3) containing copyright information for any IP cores that were used. If a user design contains no encrypted licensed IP the FPGA tools will generate a conventional unencrypted bitstream.

The trusted software needs to be able to decrypt the various encrypted cores to allow processing – this is an identical problem to that solved by Altera and others for evaluation of IP cores. Various solutions are available which depend on embedding secret information (e.g. a private key for use in a public key algorithm like RSA) in the trusted software. The trusted software also determines a new 'design' key to encrypt the resulting bitstream using a method which allows a program with access to TEP secret information to decrypt the bitstream at a later date. These cryptographic functions be implemented in hardware using a 'dongle' or expansion card in the user PC or in software - the hardware implementation allows better protection of secret key information but is less convenient.

A feature of this scheme is that no design information need be transferred to either the FPGA vendor or the TEP. The computer running the design tools need not be connected to the internet. This is advantageous since many designers would refuse to contemplate a security scheme which required that their design information was transferred to the TEP.

3.3 Programming of FPGA's by FPGA Customer

When it is desired to program an FPGA using an encrypted bitstream the bitstream is supplied to 'trusted' programming software which has embedded secret information provided by the TEP. This software runs on a computer connected to the internet and to the programming interface on the FPGA. The computer may be part of the test system for the equipment containing the FPGA.

The trusted programming software decrypts the bitstream file to extract the copyright information on any licensed IP cores included in the bitstream. It then interrogates the FPGA through its programming interface to determine its ID number (figure 4).

The trusted programming software makes contact with a server operated by the TEP over internet and provides that server with the copyright identifiers of the various cores, the ID number of the FPGA to be programmed and the billing information for the customer. The TEP computer bills the FPGA customer account for the various cores and looks up the FPGA chip identifier in its database to find the token and user key information created when the FPGA chip was manufactured. It then supplies this token to the programming software along with the corresponding user key. The communication between the trusted programming software and the TEP server is protected using a standard internet security protocol such as SSL.

The trusted programming software then re-encrypts the bitstream information using the key supplied by the TEP and appends the bitstream information to the token to create a complete set of programming information specific to this particular FPGA (figure 5). This information can be programmed into a local serial EPROM or stored elsewhere on the equipment containing the FPGA in exactly the same way as a conventional unencrypted FPGA bitstream.

The bitstream created by the programming software can only be used by one particular FPGA: as well as enforcing pay-per-use licensing the bitstream is protected against cloning and reverse engineering by this technique. There is no need for the FPGA to be physically adjacent to the programming software or for the link between the FPGA and the programming software to be secure – thus this scheme can be used to update FPGA configurations in the field via an internet connection.

When the FPGA loads the bitstream information it decrypts the token using its on-chip secret key to determine the TEP user key then uses the TEP user key to decrypt the programming information prior to configuration.

3.4 Potential Attacks on the Scheme

The scheme outlined above uses standardized ciphers (such as triple DES or AES) in a standardized mode – Cipher Block Chaining (CBC). These ciphers and modes are well understood and have a large enough key-size to resist key-search attacks (such as presenting very large numbers of potential tokens to the chip in the hope of finding the correct one). Triple DES in CBC mode is used in common protocols such as SSL and IPSEC and in the banking industry. It is believed that the algorithms used are not a significant weakness of the scheme.

The scheme also requires secret information to be stored on integrated circuit chips. Fundamentally, all cryptography is based on secret information and is only as strong as the physical security preventing the secret being accessed. There are many ways of storing secrets on integrated circuits and this is a subject of considerable study by the smartcard industry. More details of suitable techniques for FPGA's are given in [5] and [6].

The embodiment of the scheme described here requires secret information to be embedded in computer software running on general purpose computers. This choice is a tradeoff between ease-of-use and security. Information embedded in a computer program is much less well protected than information embedded in an integrated circuit. Reference [10] describes variants of the scheme in which design software runs on the TEP's secure server computers. These alternative schemes do not require secret information to be stored in software accessible to the FPGA designer or customer.

The scheme also requires a database on a server computer owned by the TEP to be kept secret. The threat here is of 'hacking' or malicious access by an 'insider' and is similar to that faced by banks and other organizations who maintain sensitive information on server computers. While the need to maintain a secure on-line database may be considered a weakness of this security scheme it is a well understood problem addressed successfully by many organizations.

4. SUMMARY

An important problem facing the emerging FPGA IP core industry is the mismatch between the up-front license fee business model and FPGA customer expectations. FPGA customers are used to paying a relatively high per-chip cost while avoiding up front non-recurring engineering charges. The IP industry, faced with the inability to enforce pay-per-use billing has adopted an ASIC like business model in which a large up-front charge is made to access IP, independent of the customers expected or actual unit volume. This prevents designers with low and medium volume applications from accessing IP cores.

Further, it has been unattractive for design houses to offer complete FPGA designs to end users as off the shelf solutions for niche markets. This 'Virtual' Application Specific Standard Product (VASSP) model is potentially a significant market since it opens FPGA technology to board level designers with no in-house FPGA design skills. In the VASSP model the design house does not release any design information beyond a data-sheet and the user does not require FPGA CAD tools except for the trusted device programming software.

A combination of simple cryptographic circuitry added to each FPGA and an internet service operated by a Trusted External Party allows convenient pay-per-use revenue collection on complete FPGA designs and IP cores incorporated in user FPGA designs.

This paper has described a conceptually simple implementation of a scheme for securing FPGA IP cores. Based on this infrastructure a wide variety of business models can be engineered to match the timing and manner of revenue extraction to the requirements of particular groups of customers. As an additional benefit, the cryptographic infrastructure on each chip can be used to secure bitstreams against reverse engineering and piracy and secure and control field upgrading of bitstream information. Reference [10] covers many possible variations, extensions and refinements to this approach.

5. REFERENCES

[1] Dipert, B., "Cunning Circuits Confound Crooks", EDN Magazine, October 12, 2000.

[2] Actel Corporation, "Protecting your Intellectual Property from the Pirates", presentation at DesignCon '98. Available from www.actel.com.

[3] Xilinx Inc., "Using Bitstream Encryption", in Chapter 2 of the Virtex II Platform FPGA Handbook available from www.xilinx.com.

[4] Austin, K., US Patent 5,388,157 "Data Security Arrangements for Semiconductor Programmable Devices"

[5] Algotronix Ltd., "Method and Apparatus for Secure Configuration of a Field Programmable Gate Array", PCT Patent Application PCT/GB00/04988.

[6] Algotronix Ltd., "Method of using a Mask Programmed Key to Securely Configure a Field Programmable Gate Array", European Patent Application EP01124330A2.

[7] Kean, T. "Secure Configuration of Field Programmable Gate Arrays", Proceedings of FPL 2001, Belfast, UK. Published as Springer LNCS 2147.

[8] Schneier, B., "Applied Cryptography", 2nd edition, Wiley, 1996.

[9] Altera Inc., "Evaluating AMPP and MegaCore Functions", AN-125, April 2000, available from www.altera.com .

[10] Algotronix Ltd., "Method of Protecting Intellectual Property Cores on Field Programmable Gate Array", unpublished pending patent application.

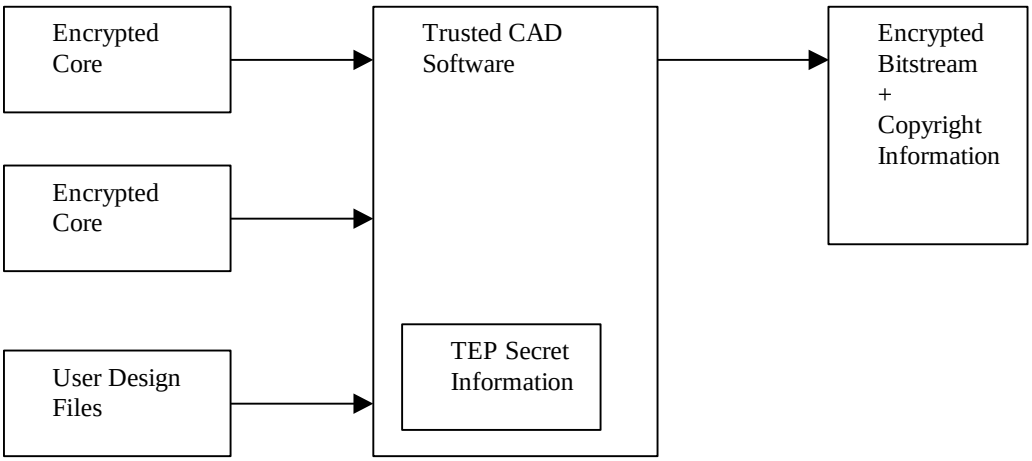

Figure 2. CAD Software Flow.

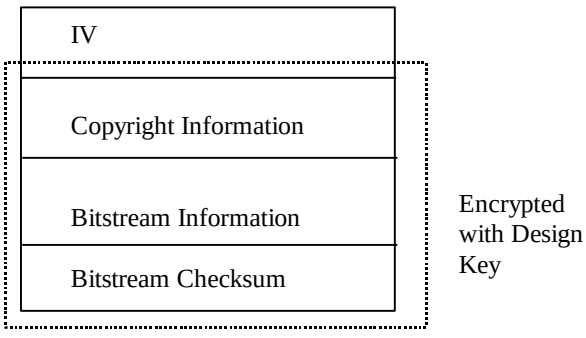

Figure 3. Encrypted Bitstream Format.

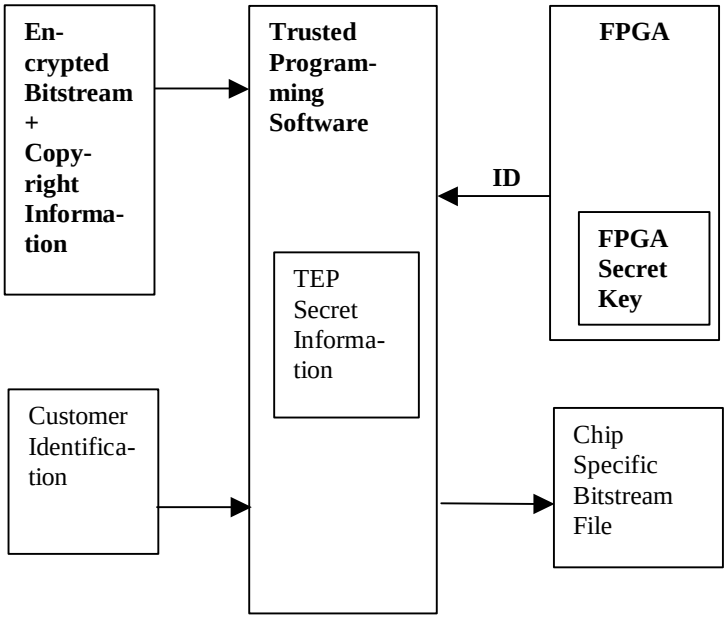

Figure 4. Programming the FPGA.

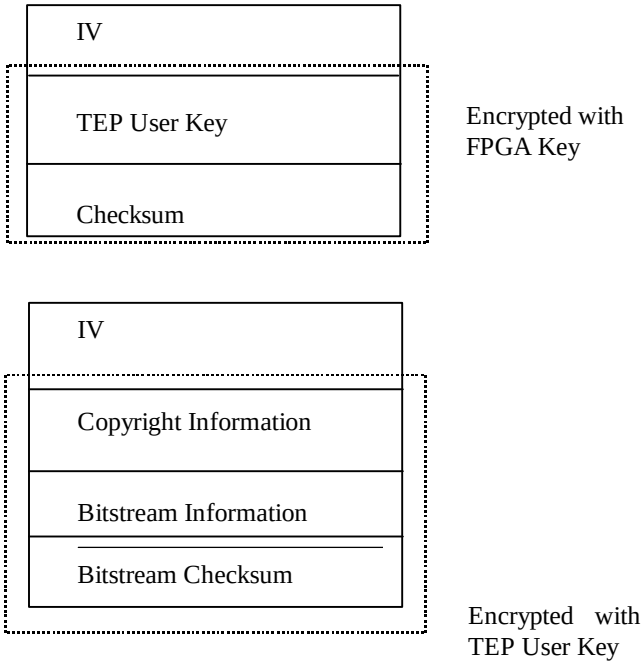

Figure 5. Chip Specific FPGA Bitstream Format.