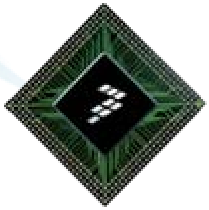
November, 2008

Secure Multicore Solutions (Crypto Acceleration, Deep Packet Inspection, Platform Trust)



PN103

Annie Huang
Technical Marketing

Abstract

- Multicore embedded processors are designed to deliver new levels of networking performance, while enabling high touch services including security functions such as content scanning and encryption. This presentation will cover the special considerations associated with parallel security processing, as well as the advantages of platform trust in a multicore system.

Who are the Bad Guys and What do They Want?

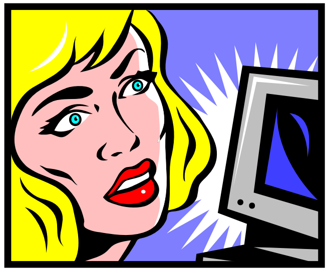
- The stereotypical attacker of the past (17-yr-old looking to make a name for himself) is being replaced by professional thieves and mercenaries. What motivates them?
- Theft of user data - loss of user data to an unauthorized party, where the network’s users had a reasonable expectation that such a loss would not occur, resulting in regulatory or reputational loss to the network owner and/or the networking OEM
- Theft of functionality - loss of control of the network’s functionality, such that users (or network owners) enable features they haven’t paid for, or unauthorized parties exploit the network's features to the detriment of authorized users
- Theft of uniqueness - loss of product differentiation through reverse engineering, duplication, and unapproved inter-operability

Stopping the Bad Guys

- Know your users: Strong Authentication.
- Prevent attackers impersonating or eavesdropping (on) your users.
- Scan the traffic for the signatures of attacks & unauthorized network usage
- Prevent the network itself from being compromised
- Protect your reputation and value add

Crypto Acceleration Technology

Mutual Authentication 101



Alice and Bob advertise their public keys
Alice's Public Key = $e_{(A)},N_{(A)}$
Bob's Public Key = $e_{(B)}$,$N_{(B)}$

They also each have a private key that is mathematically related to their public key

Alice's Private Key = $d_{(A)}$
Bob's Private Key = $d_{(B)}$

Step 1: Generate a unique, unpredictable message (random number)

Step 2: Encrypt the message with Bob's Public Key, and sends to Bob
$(\text{Message})^{e_B} \bmod N_B$

→

Step 3: Bob decrypts the message using his private key $d_{(B)}$
$\text{Message} = (\text{Message})^{d_B} \bmod N_B$

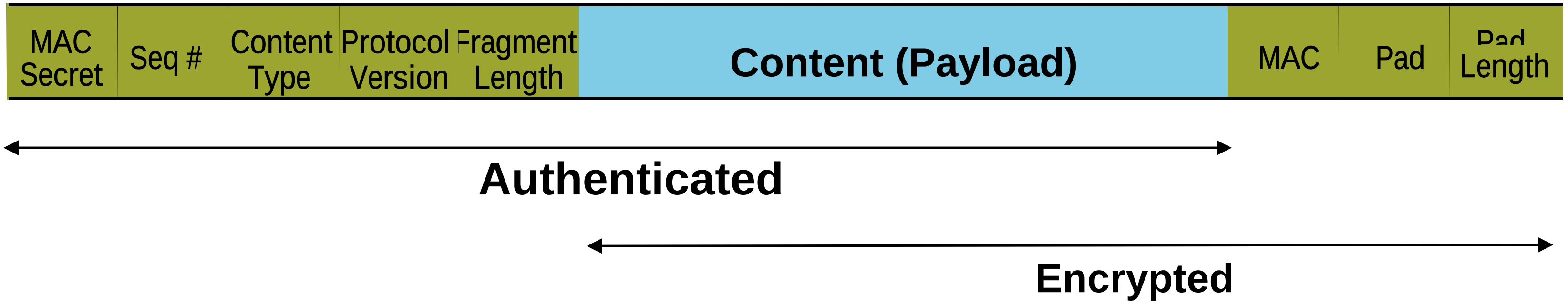
Step 4: Bob encrypt the original message with Alice's Public Key, and sends to Alice
$(\text{Message})^{e_A} \bmod N_A$

←

Step 5: Alice decrypts the message using her private key $d_{(A)}$
$\text{Message} = (\text{Message})^{d_A} \bmod N_A$

User Datagram Protection 101

SSL/TLS

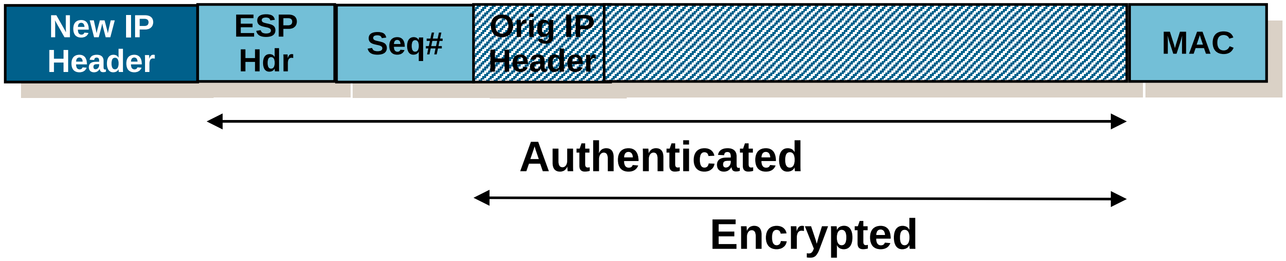


MAC Secret | Seq # | Content Type | Protocol Version | Fragment Length | Content (Payload) | MAC | Pad | Pad Length

Authenticated

Encrypted

IPsec ESP TUNNEL MODE

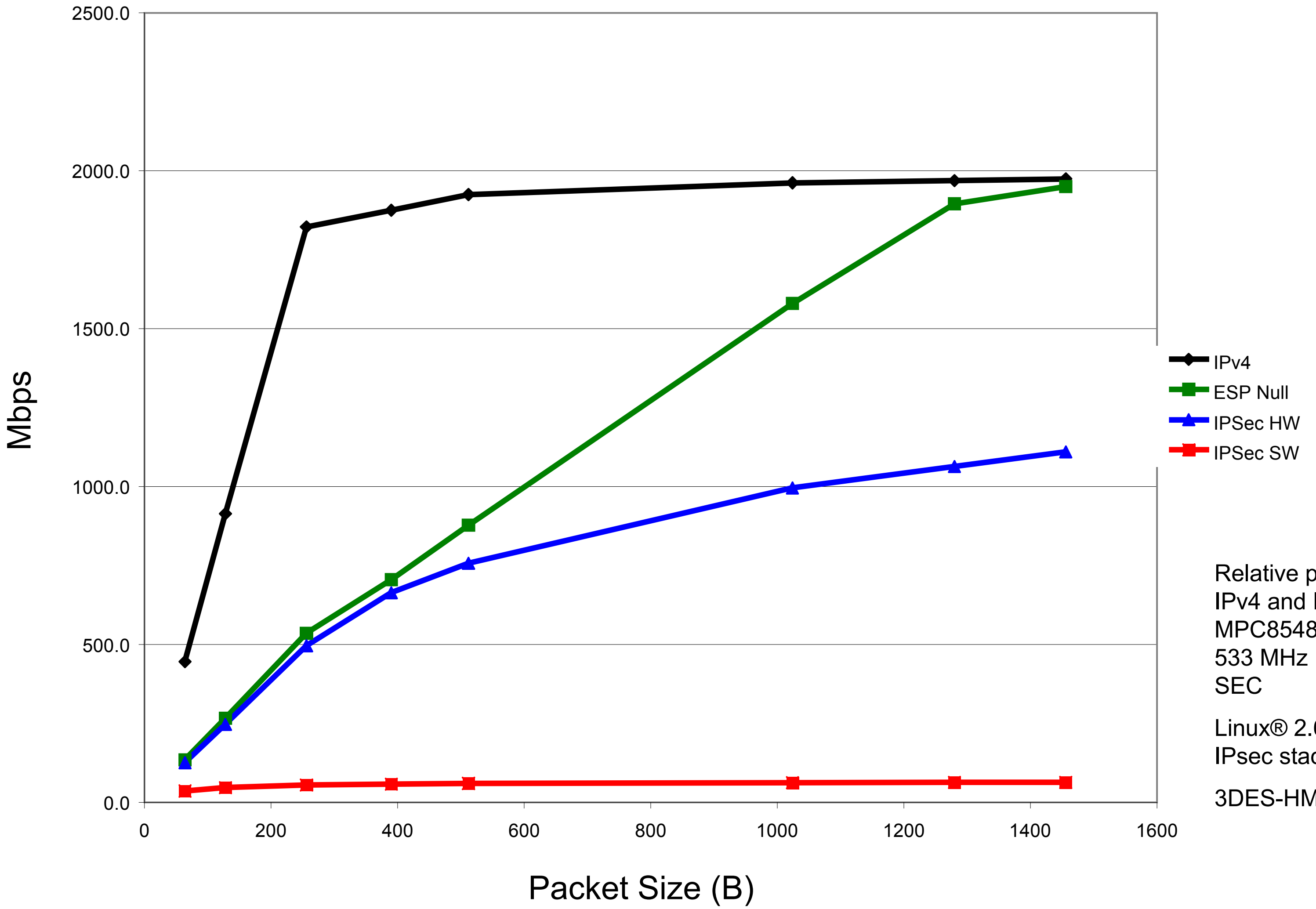


New IP Header | ESP Hdr | Seq# | Orig IP Header | MAC

Authenticated

Encrypted

Who Needs Crypto Acceleration?

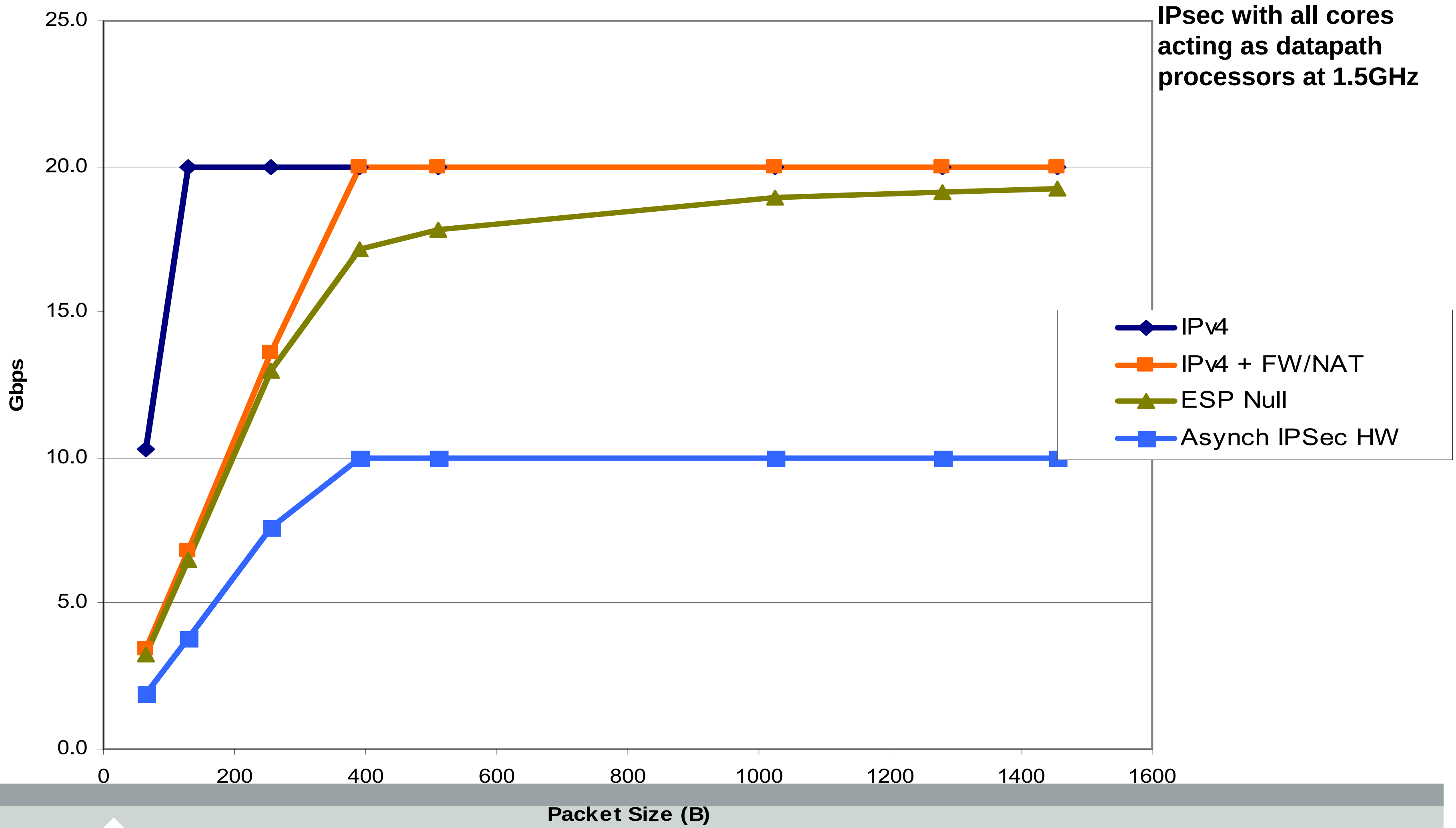


Relative performance of IPv4 and IPsec on Freescale MPC8548E, 1.3 GHz CPU, 533 MHz DDR, 266 MHz SEC

Linux® 2.6.11, 3rd party IPsec stack.

3DES-HMAC-SHA-1

QorIQ™ P4080 performance targets



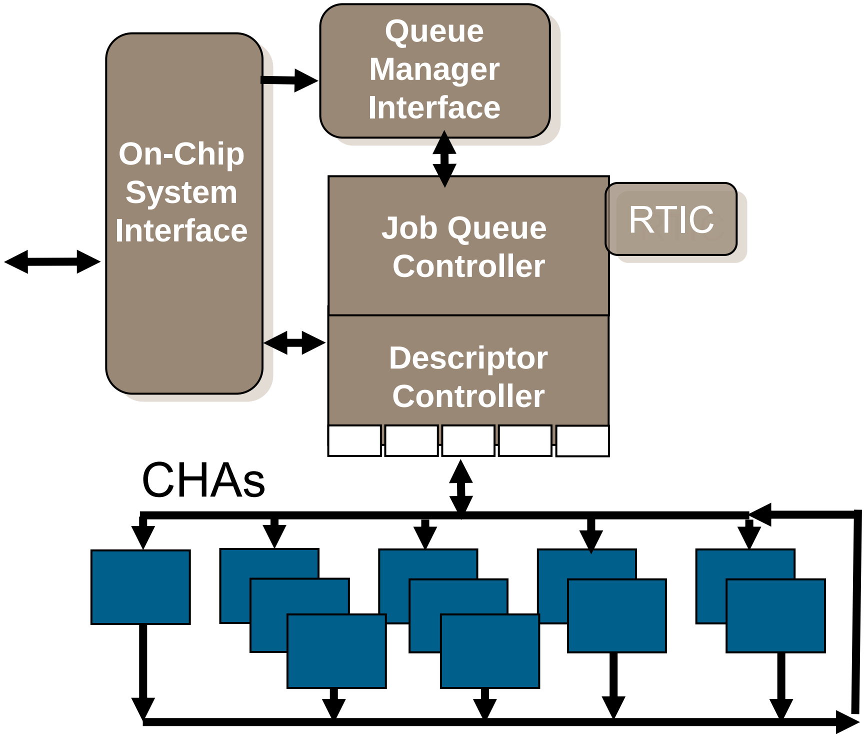
IPsec with all cores acting as datapath processors at 1.5GHz

SEC 4.0 – Next Gen Security Engine

- Public Key Hardware Accelerators (PKHA)
  - RSA and Diffie-Hellman (to 4096b)
  - Elliptic curve cryptography (1023b)
  - Supports Run Time Equalization
- Data Encryption Standard Accelerators (DESA)
  - DES, 3DES (2K, 3K)
  - ECB, CBC, OFB modes
- Advanced Encryption Standard Accelerators (AESA)
  - Key lengths of 128-, 192-, and 256-bit
  - ECB, CBC, CTR, CCM, GCM, CMAC,
  - OFB, CFB, and XTS
- Message Digest Hardware Accelerators (MDHA)
  - SHA-1, SHA-2 256,384,512-bit digests
  - MD5 128-bit digest
  - HMAC with all algorithms
- ARC Four Hardware Accelerators (AFHA)
  - Compatible with RC4 algorithm
- Kasumi/F8 Hardware Accelerators (KFHA)
  - F8 , F9 as required for 3GPP
  - A5/3 for GSM and EDGE
  - GEA-3 for GPRS
- Snow 3G Hardware Accelerators (STHA)
  - Implements Snow 3.0
- CRC Unit
  - CRC32, CRC32C, 802.16e OFDMA CRC
- Random Number Generator, random IV generation

- Header & Trailer off-load for the following Security Protocols:
  - IPSec, 802.1ae, SSL/TLS, SRTP, 802.11i, 802.16e

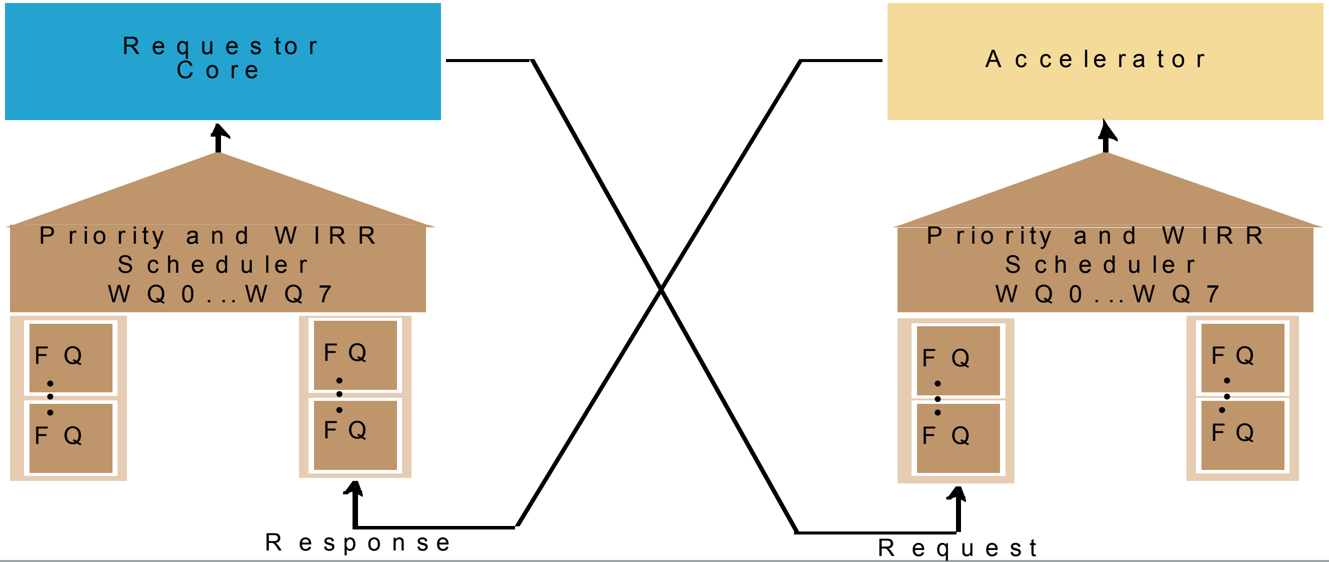
- Modular & Scalable with simplified device driver



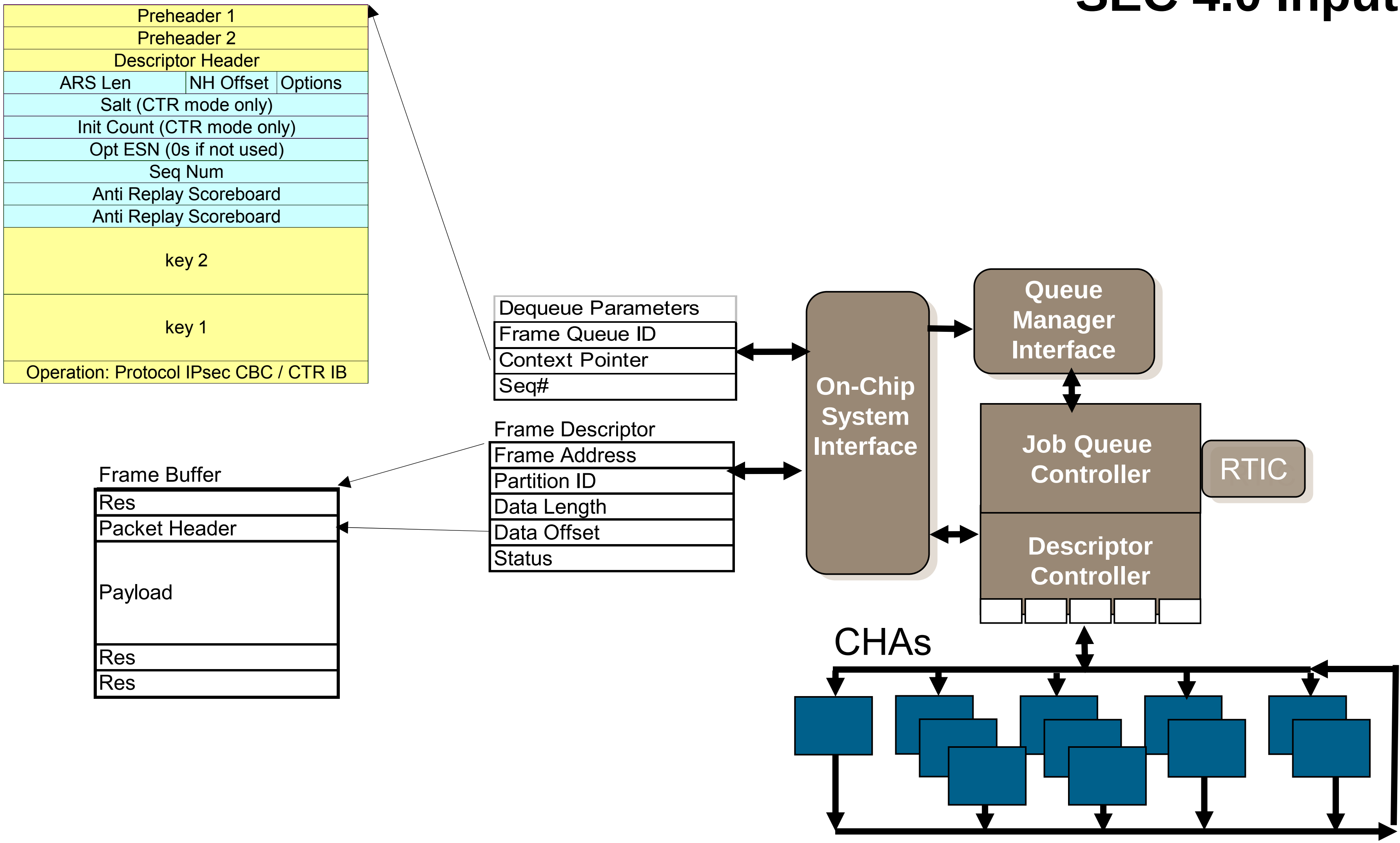
Accelerator Software Interface

SEC Drivers simplified and made common with other peripherals & accelerators

- Advanced programming model uses “Queue Drivers”
- Simplified, software-friendly interface
- Allows easy sharing of SEC by multiple CPUs
- Crypto requests are encoded in architected messages which are placed on queues
- Requests may include reference to existing session context, or include explicit context
- Responses (normal and error) flow back to software using same queue structures
- Provides common mechanism for scheduling and prioritization
- Provides common error reporting mechanism, interrupts for serious hardware errors only


Requestor Core
Accelerator
Priority and WIRR Scheduler WQ0...WQ7
FQ
FQ
FQ
FQ
Priority and WIRR Scheduler WQ0...WQ7
FQ
FQ
FQ
FQ
Response
Request

SEC 4.0 Inputs



Preheader 1		
Preheader 2		
Descriptor Header		
ARS Len	NH Offset	Options
Salt (CTR mode only)		
Init Count (CTR mode only)		
Opt ESN (0s if not used)		
Seq Num		
Anti Replay Scoreboard		
Anti Replay Scoreboard		
key 2		
key 1		
Operation: Protocol IPsec CBC / CTR IB		

Dequeue Parameters
- Frame Queue ID
- Context Pointer
- Seq#

Frame Descriptor
- Frame Address
- Partition ID
- Data Length
- Data Offset
- Status

Frame Buffer
- Res
- Packet Header
- Payload
- Res
- Res

On-Chip System Interface

Queue Manager Interface

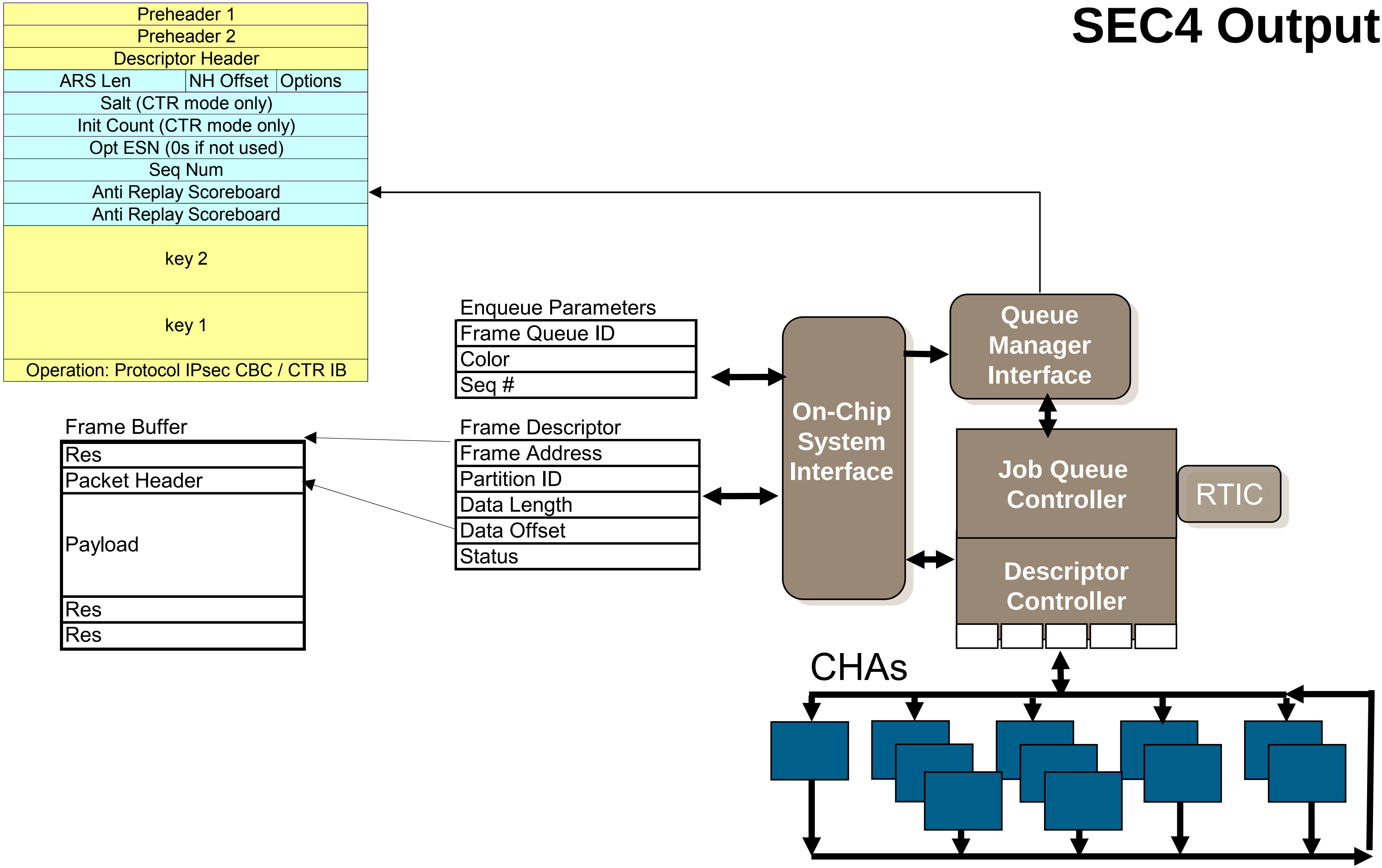
Job Queue Controller

RTIC

Descriptor Controller

CHAs

SEC4 Outputs



Preheader 1
Preheader 2
Descriptor Header
ARS Len | NH Offset | Options
Salt (CTR mode only)
Init Count (CTR mode only)
Opt ESN (0s if not used)
Seq Num
Anti Replay Scoreboard
Anti Replay Scoreboard
key 2
key 1
Operation: Protocol IPsec CBC / CTR IB

Enqueue Parameters

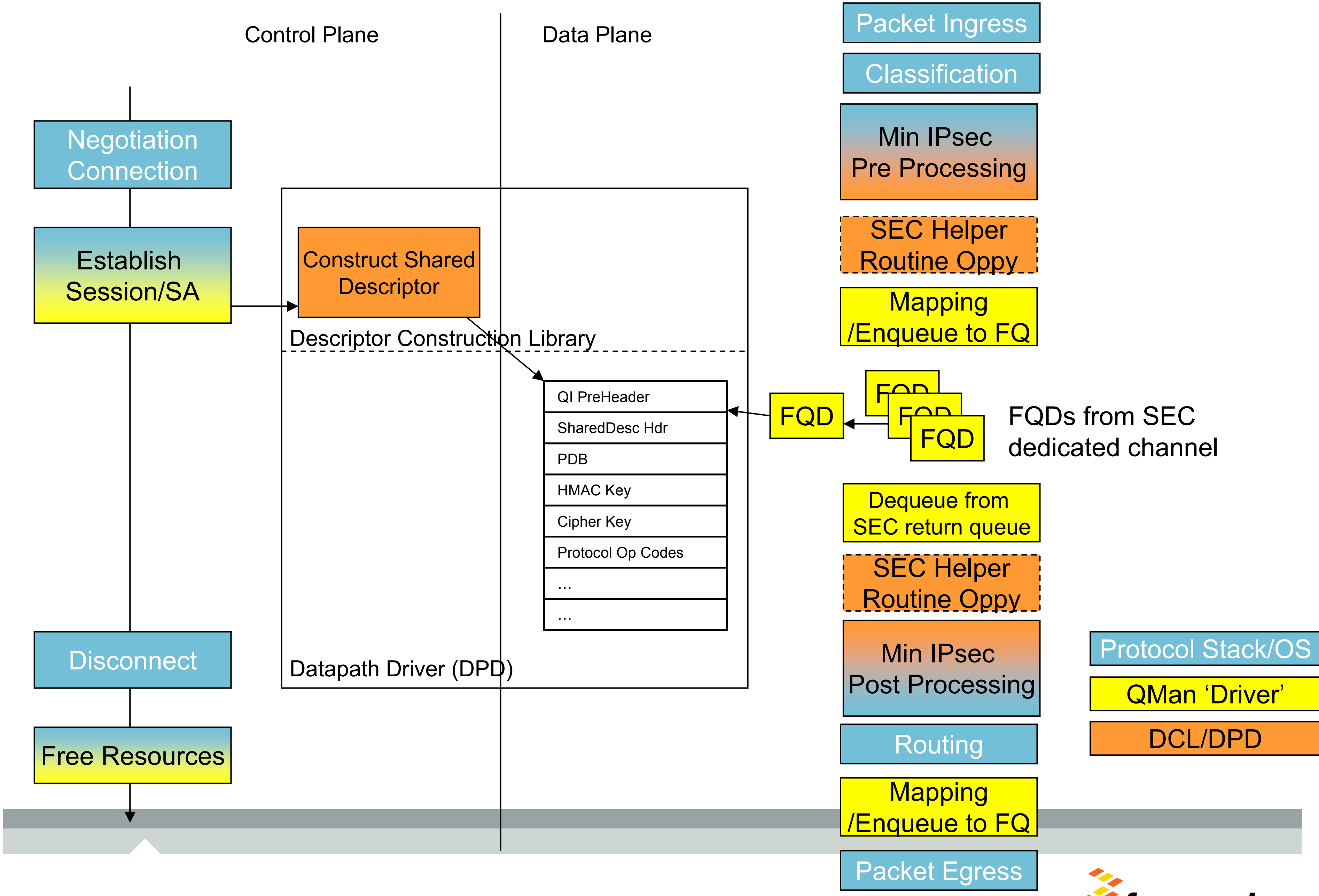
Frame Queue ID
Color
Seq #

Frame Buffer

Res
Packet Header
Payload
Res
Res

Frame Descriptor

Frame Address
Partition ID
Data Length
Data Offset
Status



Control Plane

Data Plane

Negotiation Connection

Establish Session/SA

Disconnect

Free Resources

Construct Shared Descriptor

Descriptor Construction Library

QI PreHeader

SharedDesc Hdr

PDB

HMAC Key

Cipher Key

Protocol Op Codes

…

…

Datapath Driver (DPD)

FQD

FQD

FQD

FQD

FQDs from SEC dedicated channel

Packet Ingress

Classification

Min IPsec Pre Processing

SEC Helper Routine Oppy

Mapping /Enqueue to FQ

Dequeue from SEC return queue

SEC Helper Routine Oppy

Min IPsec Post Processing

Routing

Mapping /Enqueue to FQ

Packet Egress

Protocol Stack/OS

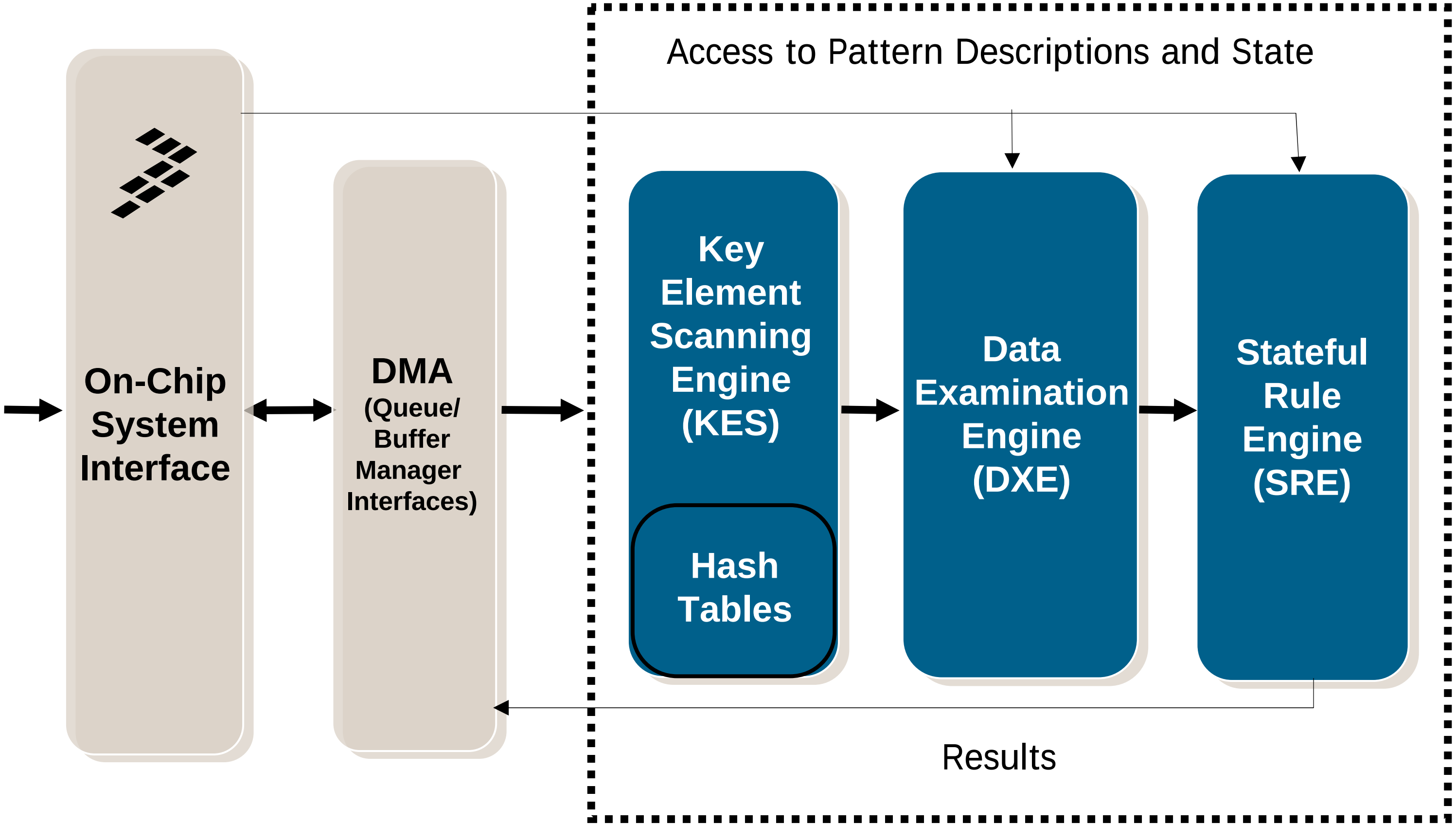
QMan ‘Driver’

DCL/DPD

Pattern Matching Engine (PME) Technology

PME 2.0 Block Diagram

Pattern Matching Engine components


Access to Pattern Descriptions and State
On-Chip System Interface
DMA (Queue/ Buffer Manager Interfaces)
Key Element Scanning Engine (KES)
Hash Tables
Data Examination Engine (DXE)
Stateful Rule Engine (SRE)
Results

Freescale Pattern Matching Engine Advantages

- Hardware-based, full-featured regular expression pattern matching:
  - Supports Perl meta-characters including wildcards, repeats, ranges, anchors, etc.
  - Stateful rules:
    - User-defined hardware instructions react to pattern match events (includes changing state, assignments, bitwise operations, addition, subtraction and comparisons)
    - Can be used to correlate patterns, qualify matches (e.g. contextual match) or to track protocol state changes
    - Delays the need for software post-processing
- Improvements over other pattern matching technologies:
  - No pattern “explosion” to support “wildcarding” or case-insensitivity
  - Fast compilation of pattern database
  - Fast incremental additions, only affected pattern records are downloaded
  - Live pattern database update
  - Patterns stored in on-chip tables and main DDR memory, no need for SRAM, RLDRAM.
- Most work performed solely with on-chip tables (external memory access required only to confirm a match)
- Can match patterns across data “work units” or packet boundaries

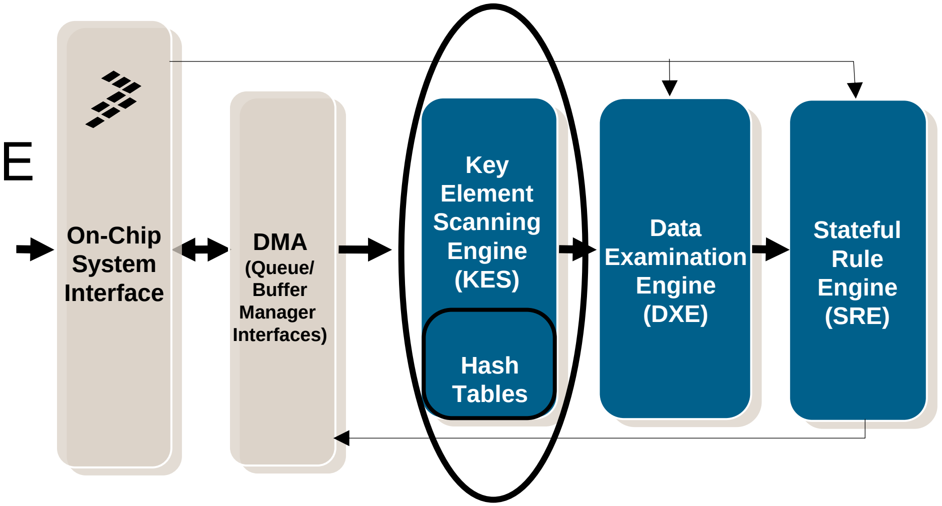
PME 2.0 Summary

- Derived from MPC8572 PME 1.0:
  - KES with internal hash tables for performance
  - DXE with full support for wildcards, repeats, ranges, captures
  - SRE for executing additional instructions following a match (e.g. contextual matching).
- 4x increase in raw performance (2.5 → 10 Gbps peak search performance)
- 4x increase in number of patterns and stateful rules

  16K → 64K total patterns (target)

  8K → 32K stateful rules (target)
- Pattern lengths from 1 to 128 bytes
- 256 sets, each with 16 subsets

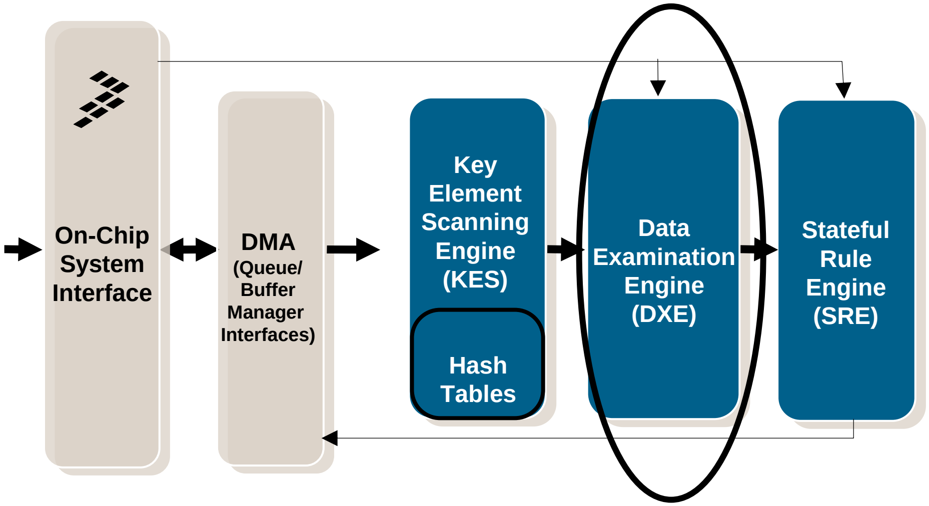
Key Element Scanning


On-Chip System Interface
DMA (Queue/ Buffer Manager Interfaces)
Key Element Scanning Engine (KES)
Hash Tables
Data Examination Engine (DXE)
Stateful Rule Engine (SRE)

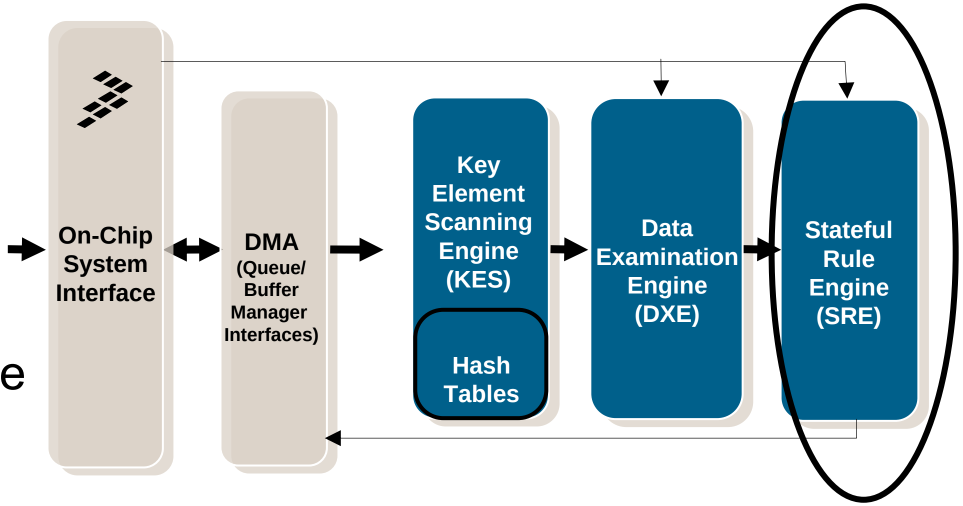
- Scans for possible matches and filters work to be performed by DXE
- All work performed using on-chip hash tables – no external memory access required
- Creates multiple formats of each incoming data byte
  - Original, translated to equivalent, pre-defined category, user-defined category
- Computes a hash for different fingerprint lengths and looks up on-chip hash tables
- A “hit” on one of these hashes results in a second level hash (“confidence” hash) being performed
  - If all levels of hash “hit” then data window is passed to data examination engine
- Scan engine continues as data examination engine checks “possible” matches for a definite match

Data Examination Engine

- Modified NFA invoked only when needed to confirm a match
- Performs complete match for each "possible" match found by KES
- Pattern definitions stored in DRAM
- Incremental updates only affect the changed pattern records
- Implements a significant subset of the regex pattern definition syntax plus many constructs which cannot be expressed in regex
  - Supports Perl meta-characters and Freescale extensions (for capture)
  - Content can be extracted from data for later comparison
  - ASCII-encoded length fields and counts embedded in data can be extracted and converted to numeric form for later use


On-Chip System Interface
DMA (Queue/ Buffer Manager Interfaces)
Key Element Scanning Engine (KES)
Hash Tables
Data Examination Engine (DXE)
Stateful Rule Engine (SRE)

Stateful Rule Engine



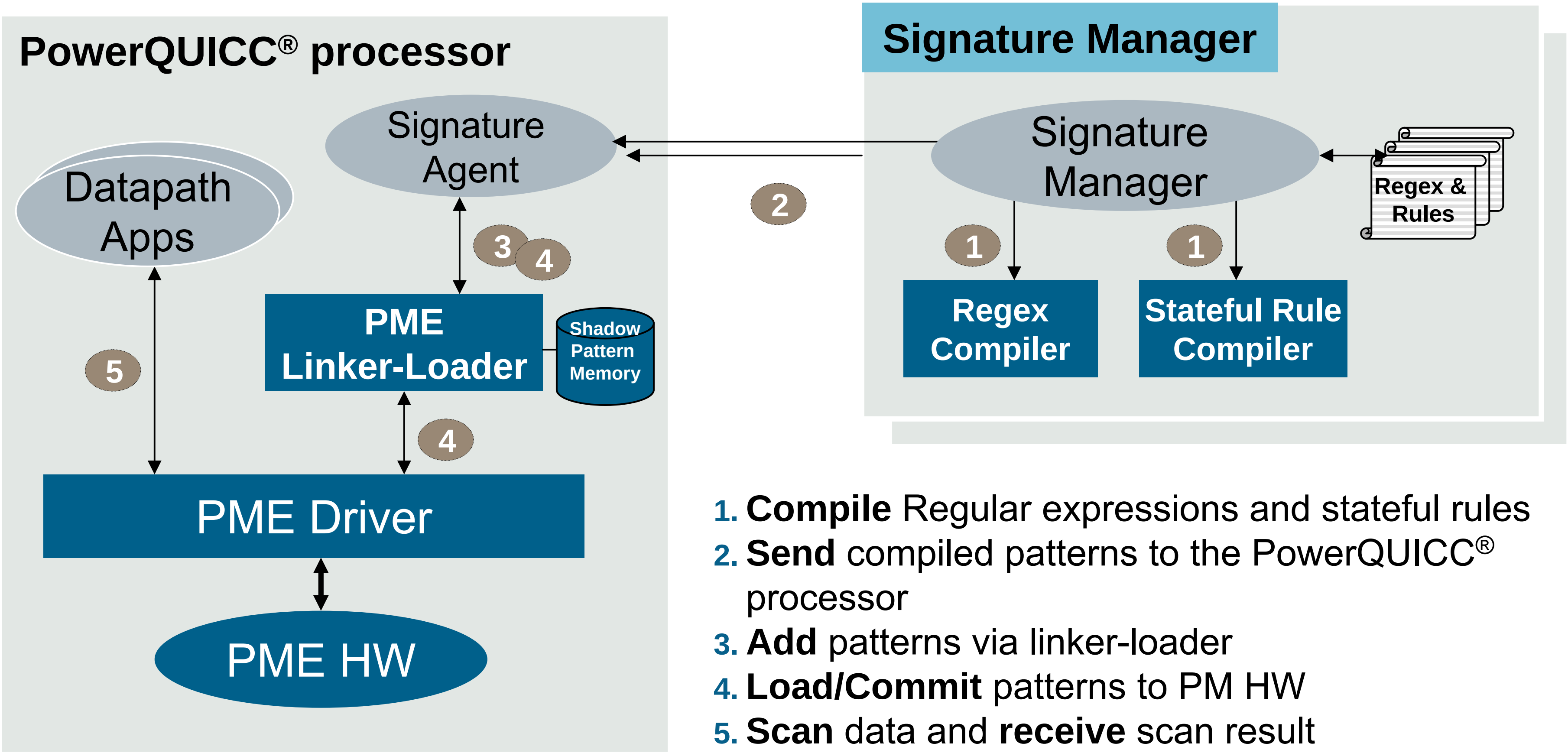
On-Chip System Interface
DMA (Queue/ Buffer Manager Interfaces)
Key Element Scanning Engine (KES)
Hash Tables
Data Examination Engine (DXE)
Stateful Rule Engine (SRE)

- User-defined logic reacts to pattern matches detected by the DXE
- Can be used to further qualify the pattern match. For example, only conclude a positive match:
  - If all patterns making up the signature are found, or
  - If the pattern is matched only within a certain portion of the data (e.g. URL), or
  - If the pattern is matched only within a certain portion of the data whose delineation is specified within the data itself (e.g. within the content portion as specified by the CONTENT_LENGTH field previously extracted from the data)
- Other uses:
  - Protocol state tracking (e.g. track the “normal” transitions of SMTP)
  - Provide support for “greedy” wildcards (e.g. ABC.*DEF == two patterns tied together by a stateful rule)
- State information stored in DRAM

Scanning Data

- Flow-Aware Scanning:
  - Create a (command) frame queue for each new flow to be scanned
  - Specify default scanning attributes for that frame queue (set, subset, session ID, result frame queue ID, etc.)
  - As packets arrive, append packet to appropriate frame queue, override default scanning attributes as required
  - Scan results are appended to result (notification) frame queue
- Flow-Agnostic Scanning:
  - Create one or several frame queues (e.g. one per priority)
  - Specify default scanning attributes for each frame queue
  - As packets arrive, append packet to appropriate frame queue, override default scanning attributes as required
  - Scan results are appended to result (notification) frame queue

PME Software Components

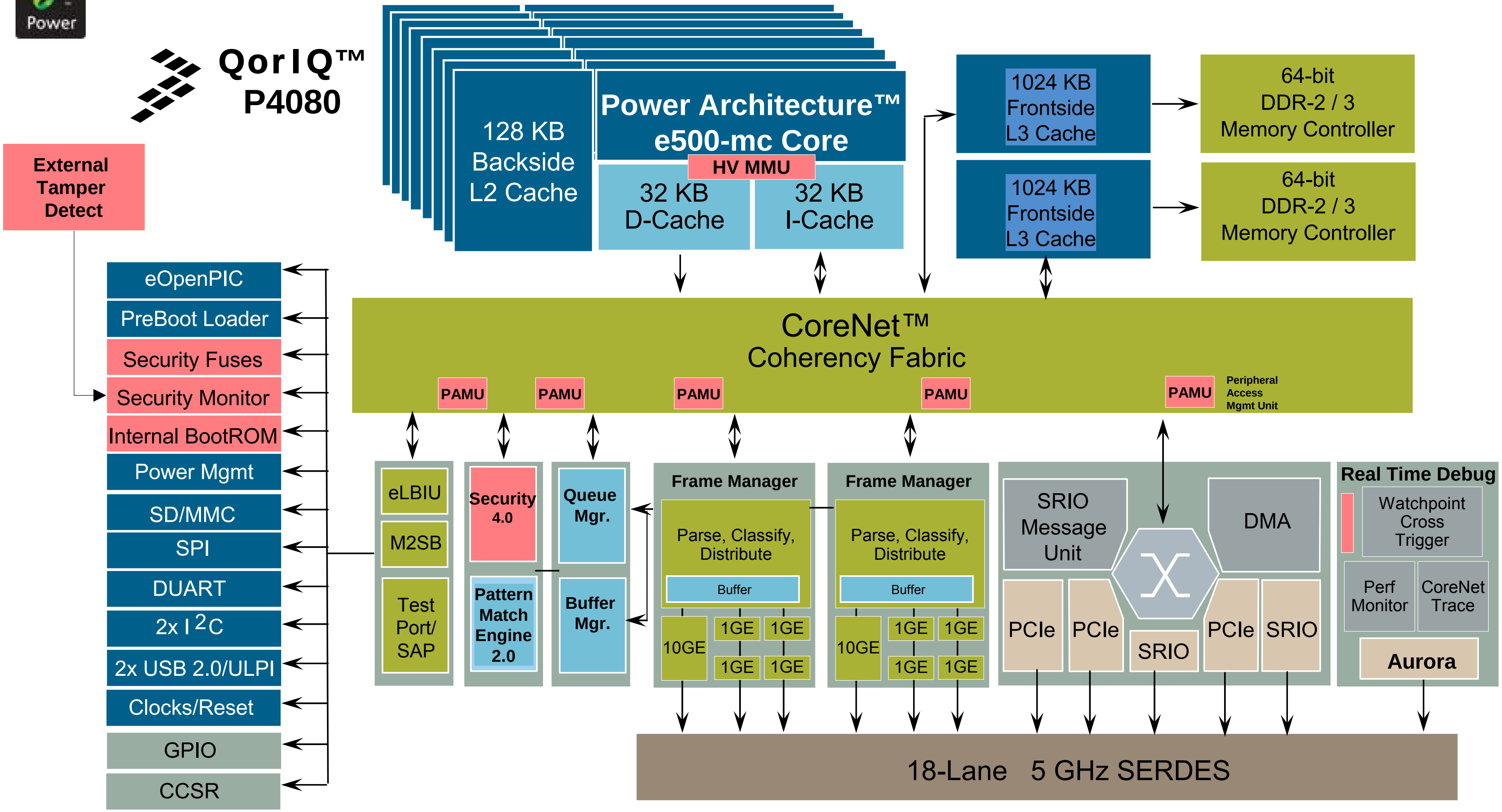


1. **Compile** Regular expressions and stateful rules
2. **Send** compiled patterns to the PowerQUICC® processor
3. **Add** patterns via linker-loader
4. **Load/Commit** patterns to PM HW
5. **Scan** data and **receive** scan result

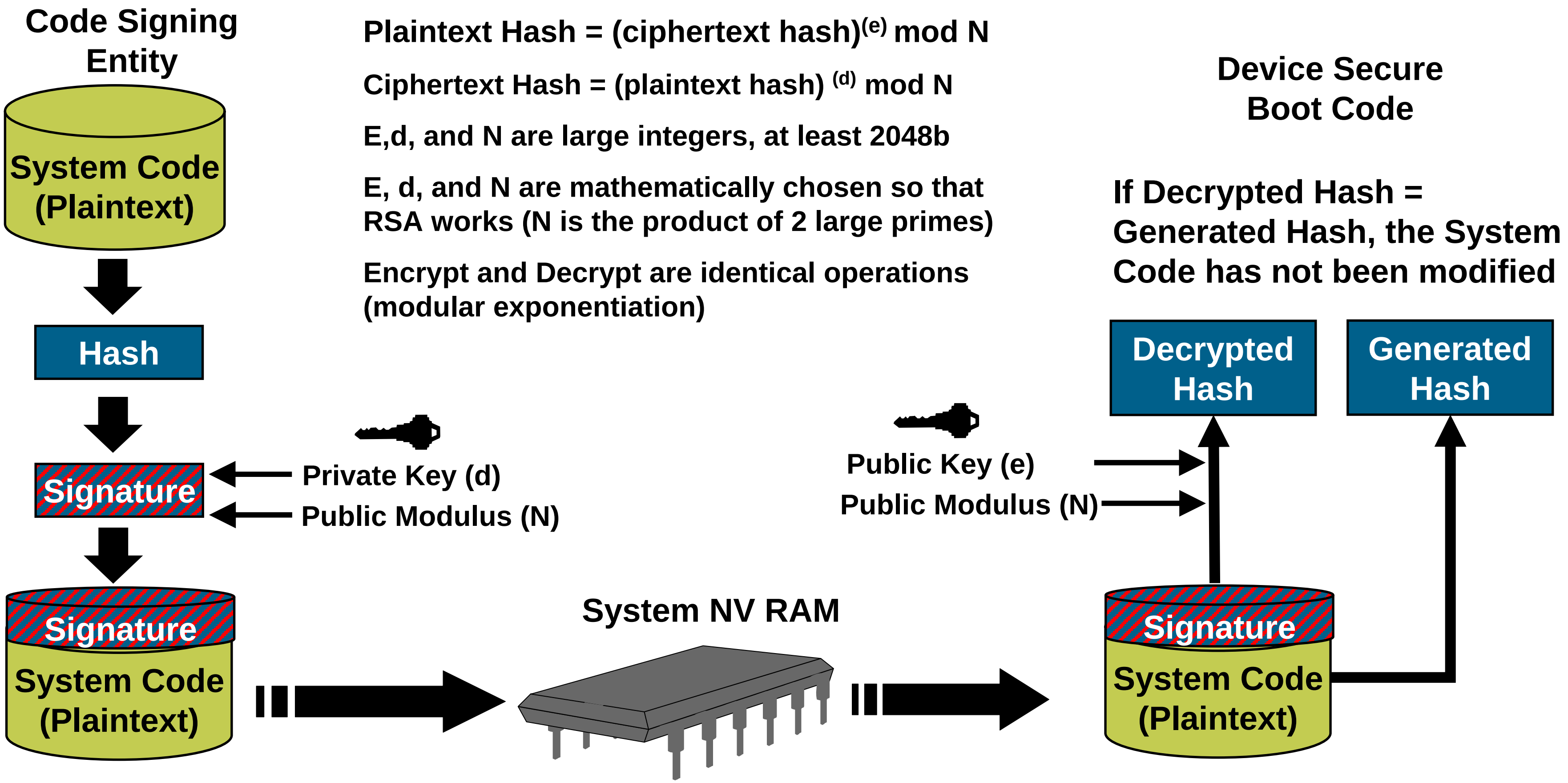
Can **Incrementally** Compile, Send, Add and Load patterns

Platform Trust Technology

QorIQ™ P4080 Block Diagram



Trusted Boot Process



Plaintext Hash = (ciphertext hash)$^{(e)}$ mod N

Ciphertext Hash = (plaintext hash) $^{(d)}$ mod N

E,d, and N are large integers, at least 2048b

E, d, and N are mathematically chosen so that RSA works (N is the product of 2 large primes)

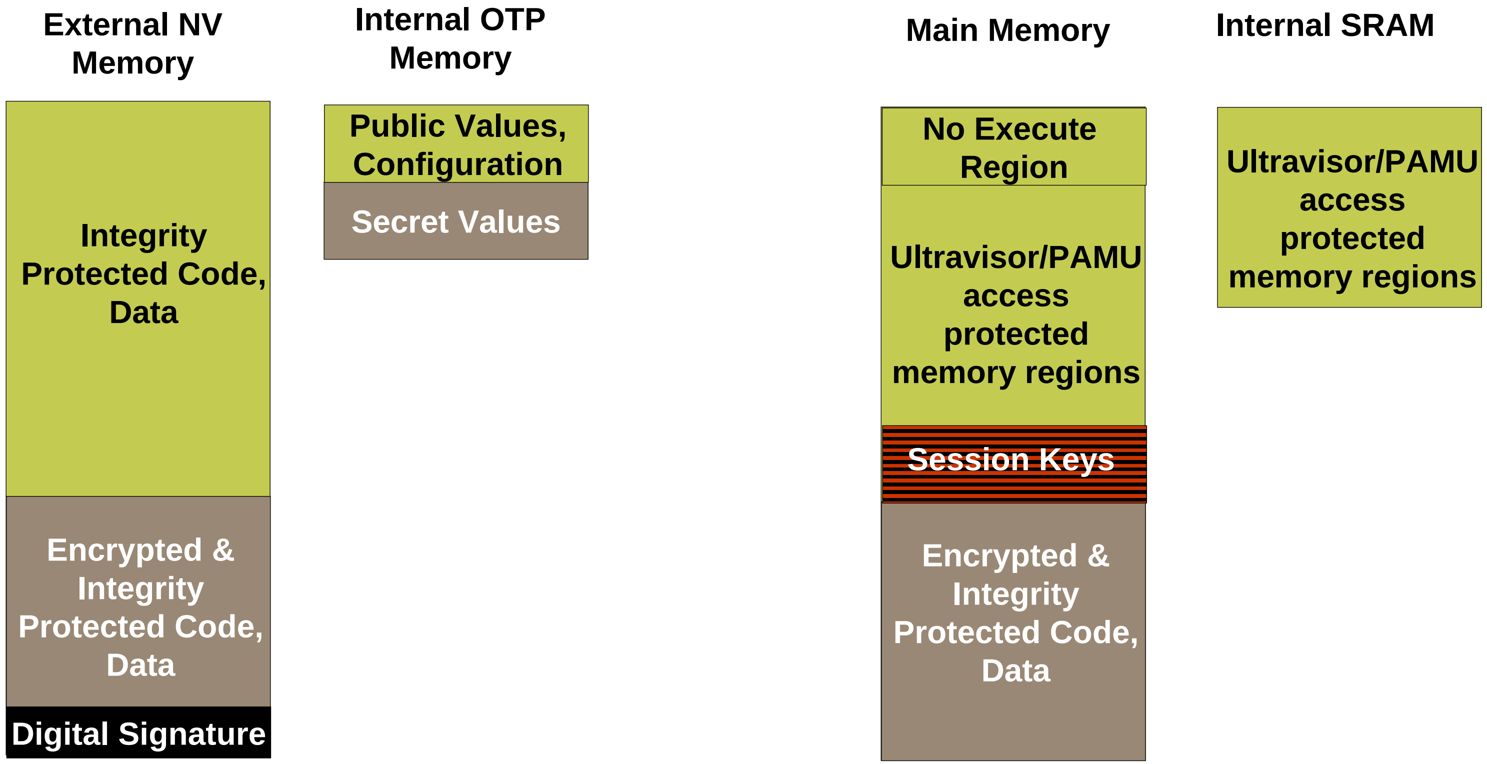
Encrypt and Decrypt are identical operations (modular exponentiation)

If Decrypted Hash = Generated Hash, the System Code has not been modified

Secure Storage

Non-Volatile

Volatile (with zeroization option)



External NV Memory

Integrity Protected Code, Data

Encrypted & Integrity Protected Code, Data

Digital Signature

Internal OTP Memory

Public Values, Configuration

Secret Values

Main Memory

No Execute Region

Ultravisor/PAMU access protected memory regions

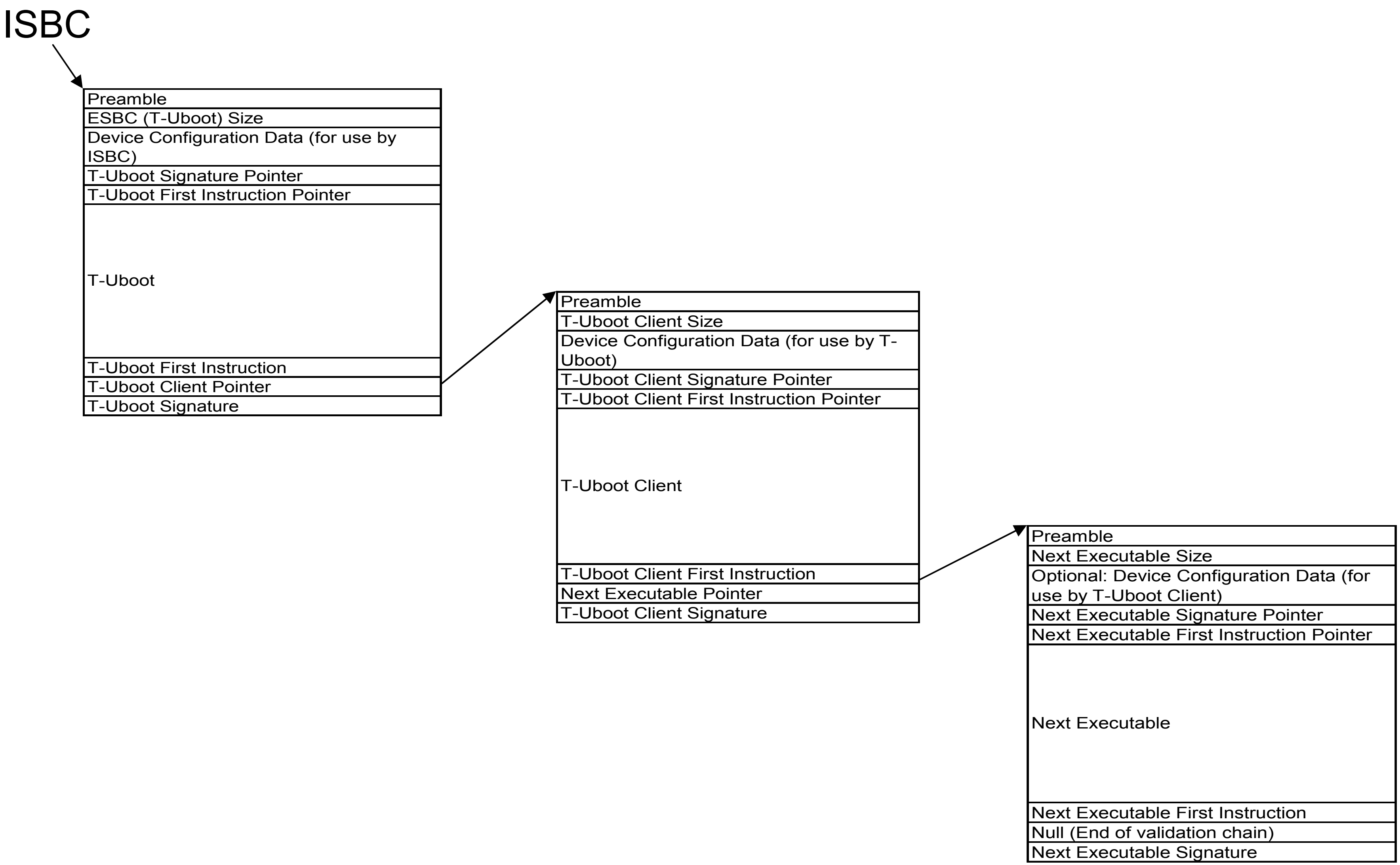
Session Keys

Encrypted & Integrity Protected Code, Data

Internal SRAM

Ultravisor/PAMU access protected memory regions

Secure Boot Chain of Trust



ISBC

Preamble
ESBC (T-Uboot) Size
Device Configuration Data (for use by ISBC)
T-Uboot Signature Pointer
T-Uboot First Instruction Pointer
T-Uboot
T-Uboot First Instruction
T-Uboot Client Pointer
T-Uboot Signature

Preamble
T-Uboot Client Size
Device Configuration Data (for use by T-Uboot)
T-Uboot Client Signature Pointer
T-Uboot Client First Instruction Pointer
T-Uboot Client
T-Uboot Client First Instruction
Next Executable Pointer
T-Uboot Client Signature

Preamble
Next Executable Size
Optional: Device Configuration Data (for use by T-Uboot Client)
Next Executable Signature Pointer
Next Executable First Instruction Pointer
Next Executable
Next Executable First Instruction
Null (End of validation chain)
Next Executable Signature

Related Session Resources

Session Location – Online Literature Library

http://www.freescale.com/webapp/sps/site/homepage.jsp?nodeId=052577903644CB

Sessions

Session ID	*Title*

Demos

Pedestal ID	*Demo Title*

freescale™
semiconductor