Power Matters.

Microsemi

Security Working Group

Cryptographic Extensions Proposal – Current Status

Richard Newell

For the 7th RISC-V Workshop Day 2 (Nov. 29, 2017)
Hosted by Western Digital, Milpitas CA

Security Working Group

- The working group decided to work in two main areas:
  - Trusted Execution / Isolation (and related subjects)
  - Cryptographic Extensions
- Trusted Execution / Isolation status
  - A work in progress
  - Recent changes in direction
  - Not yet ready to present
- Cryptographic Extensions status
  - Also a work in progress – but somewhat further along
  - The subject of this presentation
  - Hope to have a specification by the next workshop

Using RISC-V Vector Extensions for Crypto

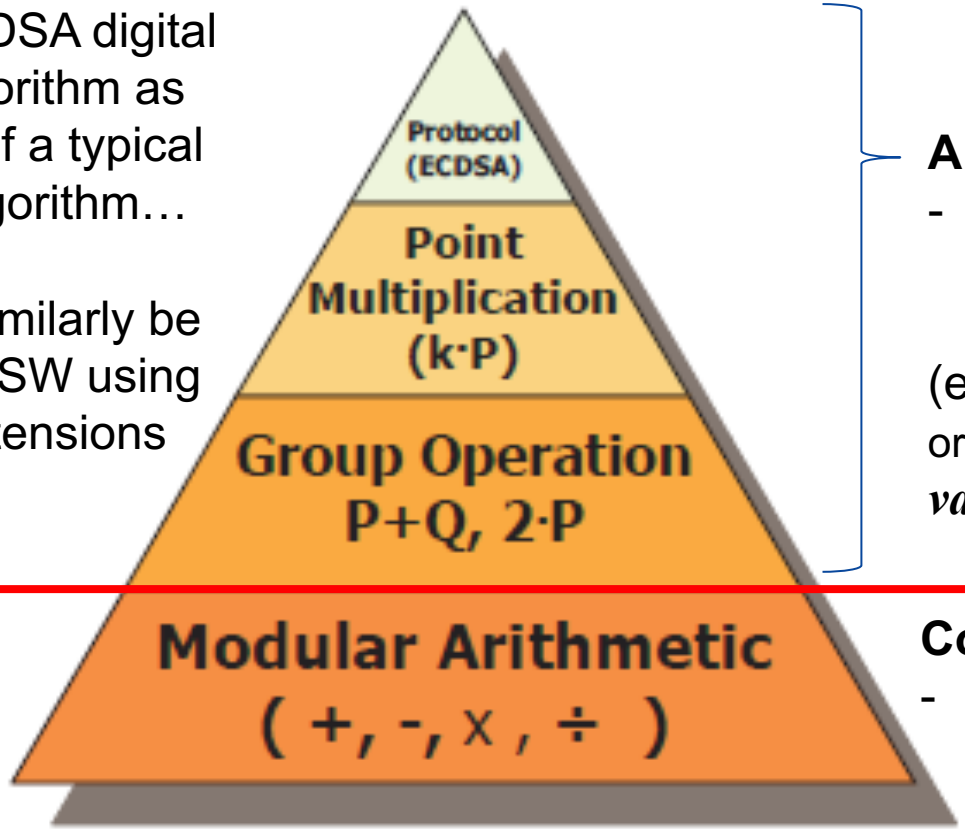
A work in progress – subject to change

- Use vector registers to hold the wide words and data types typically used in crypto:
  - Wide Vector Elements
  - Galois Field and modular data types, bit-strings, keys
- Use vector functional units (VFUs) to perform:
  - Modular and Galois Field arithmetic needed for existing popular and promising post-quantum asymmetric cryptography schemes
  - Also, accelerate symmetric block ciphers and digest algorithms taking advantage of the wide vector registers, using specialized VFUs
    - E.g., AES, SHA-2
  - Implementations may be
    - Small: E.g., based on a single 32x32 multiplier and 1 AES S-box, or
    - Large: highly parallel, pipelined, with multi-lane VFUs
  - Operations may be scalar or vectorized. If strip-mined properly, object code is portable between large and small implementations

RISC-V Asymmetric Cryptography Acceleration

Using an ECDSA digital signature algorithm as an example of a typical public-key algorithm…

RSA would similarly be performed in SW using the vector extensions



Algorithm Specific
- Perform in SW using the RISC-V Vector Extension

(e.g., ***vmul***, ***vadd*** instructions, or with field reduction: ***vmulr***, ***vaddr***)

SW

ISA

HW

Compute Intensive
- Perform arithmetic in HW In Vector Functional Units

All widely-used public key algorithms use one of the following number systems:
- Positive integers $\mathbb{N}$ (typically with long words, e.g., 128 - 8096 bits)
- Galois Field: Prime Field GF_p or a Binary Extension Field GF_{2^n}

Figure from Chapter 9 of *Understanding Cryptography* by Christof Paar and Jan Pelzl

Proposed Vector Types for the Crypto Extension

Possible Vector-Type Meta Data Encoding (5-bit) incl. Cryptographic Types

Index	Abbrev.	Description
0	D	Vector Register is Disabled
1	Un	Unsigned Integer
2	In	Signed Integer
3	Fn	Floating Point
4	GFn	Prime Galois Field – Integer Representation
5	GMn	Prime Galois Field – Montgomery Representation
6	GPn	Binary Extension Galois Field – Polynomial Basis
7	GNn	Binary Extension Galois Field – Normal Basis
8	Bn	Bitstring (e.g., message, digest)
9	BRn	Restricted-access[1] Bitstring (e.g., Secret Key)
10	UNn	UNORM
11	INn	SNORM
12	CIn	Complex Signed Integer
13	CFn	Complex Floating Point
14-23	TBD	Reserved by Foundation for 10 more standard types (future)
24-31	User defined	Available for implementer's custom types (up to 8)

"Base" Vector Extension Types

Proposed Vector Extension Types for Crypto

The Foundation "owns" the first 24 data types

"n" is total size in bits, for example: U2048 (p/o RSA public key), GM384 (NIST ECC P384 Montgomery intermediate), BR192 (192-bit AES key), CF128 ([x, y*i*] F64 x 2)

Use ***vcvt*** instruction to "Montgomerize" and "de-Montgomerize" integers

[1] Definition of "restricted access" TBD

Proposed Vector Element Widths (5-bit encoding)

Index	Width Encoded	Index	Width Encoded
0	ESC0	16	24
1	ESC1	17	32
2	ESC2	18	48
3	1	19	64
4	2	20	96
5	3	21	128
6	4	22	192
7	5	23	256
8	6	24	384
9	7	25	512
10	8	26	768
11	9	27	1024
12	10	28	1536
13	11	29	2048
14	12	30	3072
15	16	31	4096

Wide vector element widths available especially to support cryptography while shorter ones useful for graphics, machine-learning and general computing

All values 1-12, then 1 : 1½ : 2 sequence

Three escape codes point to larger registers to describe up to three wider elements (e.g., 8096) or uncommon values (e.g., 521) at any given time using direct integer encoding

Index	Width
ESC0	3 x 16 bits
ESC1	(Integer encoding)
ESC2	

Cryptographic Extension Proposal

Subject to Change!

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
func4				alg'm ptr			src2					src1					0	m	m	dest						VCRYPT						
src3				func2			src2					src1					1	m	m	dest						VCRYPT						

Index	Function	Index	Function
0	Initialize	8	reserved
1	Encrypt	9	reserved
2	Decrypt	10	reserved
3	Digest	11	reserved
4	Pad	12	custom
5	Look-Up	13	custom
6	reserved	14	custom
7	reserved	15	custom

3 → 8 bits

8x 8-bits

Index	Function
0	Add (w/ reduction)
1	Sub (w/ reduction)
2	Mul (w/ reduction)
3	reserved

Index	Algorithm	Allowed (key/digest) sizes
0	AES	128,192, 256
1	DES	Fixed (56)
2	3DES	56,112,168
3	SHA-1	Fixed (160)
4	SHA-2	128,192,256,384,512
5	SHA-3	128,192,256,384,512
6:n	TBA	Other assigned ciphers & digests
n+1:191	TBD	Reserved by Foundation (future)
192-255	custom	Reserved for non-standard algo's

The Foundation "owns" the list of 192 reserved algorithm names/encodings

Encoding space reserved to prevent collisions – implementation optional except in profiles

Algorithms Used in Some Crypto Suites/Libraries

Not considered secure enough for new applications (or completely broken)

[1] Ciphers that are also used in digests or MACs in these cipher suites (e.g., AES-CCM)

[2] Not all supported algorithms shown for brevity

	CIPHERS																DIGESTS								
	AES [1]	Camilia	ARIA	ChaCha20	DES	3DES	Skipjack	BLOWFISH	GOST (28149) [1]	SEED	CAST	IDEA	KASUMI	SNOW	A5/1,A5/2	RC2	RC4	...[2]	MD5	SHA-1	SHA-2	SHA-3	SHAKE	GOST (R3411)	POLY 1305	RIPEMD-160	...[2]
OpenSSL	✓	✓			✓	✓		✓	✓	✓	✓	✓				✓	✓		✓	✓	✓			✓			
TLS1.3	✓	✓	✓	✓															✓	✓	✓			✓	✓	✓	
TLS1.2	✓	✓	✓			✓			✓	✓							✓		✓	✓							
IPSec	✓					✓				✓										✓	✓						
NIST FIPS	✓					✓	✓													✓	✓	✓	✓				
PKCS#11	✓	✓	✓		✓	✓		✓	✓	✓										✓	✓						
S/MIME3.2	✓					✓										✓			✓	✓	✓						
OpenPGP/GnuPG	✓	✓				✓		✓			✓	✓							✓	✓	✓					✓	
CNSA	✓																				✓						
3G/4G Cellular													✓	✓	✓												
…																											

Profile Concept, as Applied to the Crypto. Ext'n

Exemplary only – not final

	ARITH.[1]				CIPHERS[2]																DIGESTS[3]						
	Large Integers, Mod. (U,GM)	Prime Galois Field (GF,GM)	Ext'n Field, Poly Basis (GP)	Ext'n Field, Normal Basis (GN)	AES	Camilia	ARIA	ChaCha20	DES	3DES	Skipjack	BLOWFISH	GOST (28149)	SEED	CAST	IDEA	KASUMI	SNOW	A5/1	RC2	RC4	MD5	SHA-1	SHA-2	SHA-3	SHAKE	GOST (R3411)	POLY 1305
Internet	✓	✓	✓		✓			✓		✓														✓	✓			✓
Finance	✓	✓			✓					✓														✓				
Cellular	✓	✓			✓												✓	✓						✓				
…																												

✓ Required
✓ Recommended

[1] Set of required and recommended vector element widths will also be specified
[2] Set of required and recommended block and key sizes will also be specified
[3] Set of required and recommended block and digest sizes will also be specified
Deprecated ciphers and digests shown greyed-out

Thank You!

Questions?

Back-up:
Crypto algorithms used in some other libraries

Crypto Libraries – Block Ciphers

Implementation	AES-128	AES-192	AES-256	Camellia	3DES	Blowfish	Twofish	CAST5	IDEA	GOST 28147-89
Botan	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Bouncy Castle	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
CryptoComply	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
cryptlib	Yes	Yes	Yes	No	Yes	Yes				
Crypto++	Yes	Yes	Yes	Yes	Yes	Yes				
Libgcrypt	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
libsodium	No	No	Yes	No	No	No				
Nettle	Yes	Yes	Yes	Yes	Yes	Yes				
wolfCrypt	Yes	Yes	Yes	Yes	Yes	No			Yes	
libtomcrypt	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No

https://en.wikipedia.org/wiki/Comparison_of_cryptography_libraries#Hash_functions

Crypto Libraries – Stream Ciphers

Implementation	RC4	HC-256	Rabbit	Salsa20	ChaCha	SEAL	Panama	WAKE	Grain	VMPC	ISAAC
Botan	Yes	No	No	Yes	Yes	No	No	No	No	No	No
Bouncy Castle	Yes	Yes	No	Yes	Yes	No	No	No	Yes	Yes	Yes
CryptoComply	Yes	Yes	No	Yes	Yes	No	No	No	Yes	Yes	Yes
cryptlib	Yes	No	No	No	No	No	No	No	No	No	No
Crypto++	Yes	No	No	Yes	No	Yes	Yes	Yes	No	No	No
Libgcrypt	Yes	No	No	Yes	Yes	No	No	No	No	No	No
libsodium	No	No	No	Yes	Yes	No	No	No	No	No	No
Nettle	Yes	No	No	Yes	Yes	No	No	No	No	No	No
wolfCrypt	Yes	Yes	Yes	Yes	Yes	No	No	No	No	No	No
libtomcrypt	Yes	No	No	No	Yes	No	No	No	No	No	No

https://en.wikipedia.org/wiki/Comparison_of_cryptography_libraries#Hash_functions

Crypto Libraries – Digests

Implementation	MD5	SHA-1	SHA-2	SHA-3	RIPEMD-160	Tiger	Whirlpool	GOST	Stribog	BLAKE2
Botan	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Bouncy Castle	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes
CryptoComply	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
cryptlib	Yes	Yes	Yes	Yes	Yes	No	Yes	No	No	No
Crypto++	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes
Libgcrypt	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
libsodium	No	No	Yes	No	No	No	No	No	No	Yes
Nettle	Yes	Yes	Yes	Yes	Yes	No	No	Yes		No
OpenSSL	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes		Yes
wolfCrypt	Yes	Yes	Yes	Yes	Yes	No	No	No		Yes
libtomcrypt	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No	Yes

https://en.wikipedia.org/wiki/Comparison_of_cryptography_libraries#Hash_functions

Crypto Libraries – MACs

Implementation	HMAC-MD5	HMAC-SHA1	HMAC-SHA2	Poly1305-AES	BLAKE2-MAC
Botan	Yes	Yes	Yes	Yes	Yes
Bouncy Castle	Yes	Yes	Yes	Yes	Yes
CryptoComply	Yes	Yes	Yes	Yes	Yes
cryptlib	Yes	Yes	Yes	No	No
Crypto++	Yes	Yes	Yes	No	Yes
Libgcrypt	Yes	Yes	Yes	Yes	Yes
libsodium	No	No	Yes	Yes	Yes
Nettle	Yes	Yes	Yes	Yes	No
wolfCrypt	Yes	Yes	Yes	Yes	Yes
libtomcrypt	Yes	Yes	Yes	Yes	Yes

https://en.wikipedia.org/wiki/Comparison_of_cryptography_libraries#Hash_functions