A Galois Field Arithmetic Library

Pakize ŞANAL, MSc Candidate

Supervisor: Asst. Prof. Hüseyin HIŞIL

Yasar University
Faculty of Engineering
Department of Computer Engineering

June 5, 2017

Outline

Content of the bachelor thesis

Studied assembly optimizations

Test results

Content of the bachelor thesis

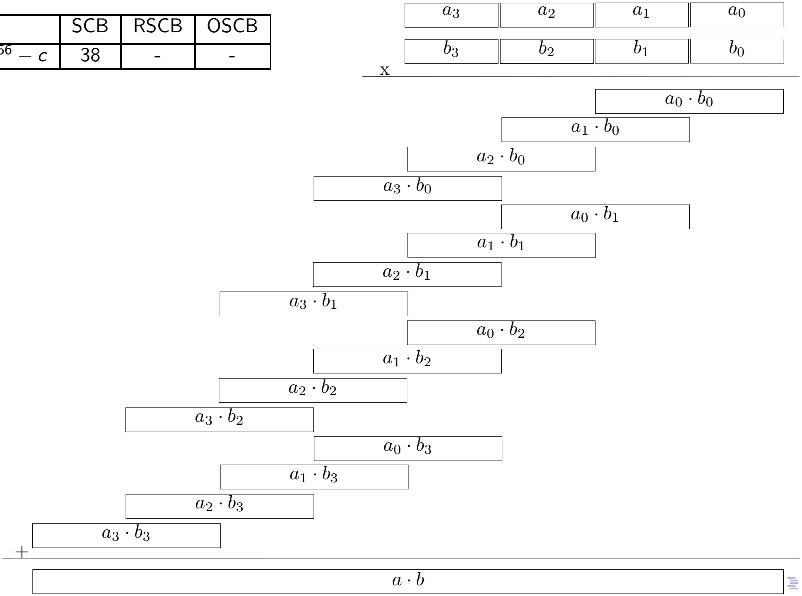
A Galois Field Arithmetic Library

- $+, -, *$.
- $GF(2^w - c)$ where $w = 127, 128, 255, 256$ and $GF(2^{127} - 1)$.
- Constant time `AMD64` Assembly.
- Extensive validation and performance tests.

1. By scheduling of the operations

Four digits schoolbook vs. one level recursive schoolbook multiplication vs. . . .

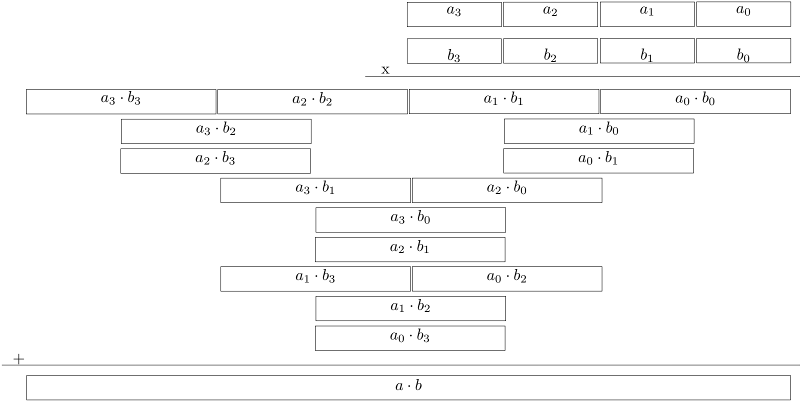
	SCB	RSCB	OSCB
$2^{256} - c$	38	-	-



1. By scheduling of the operations

Four digits schoolbook vs. one level recursive schoolbook multiplication vs. . . .

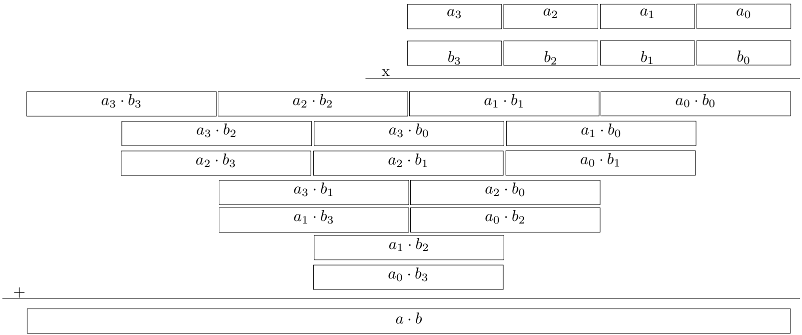
	SCB	RSCB	OSCB
$2^{256} - c$	38	35	-



1. By scheduling of the operations

Four digits schoolbook vs. one level recursive schoolbook multiplication vs. ...

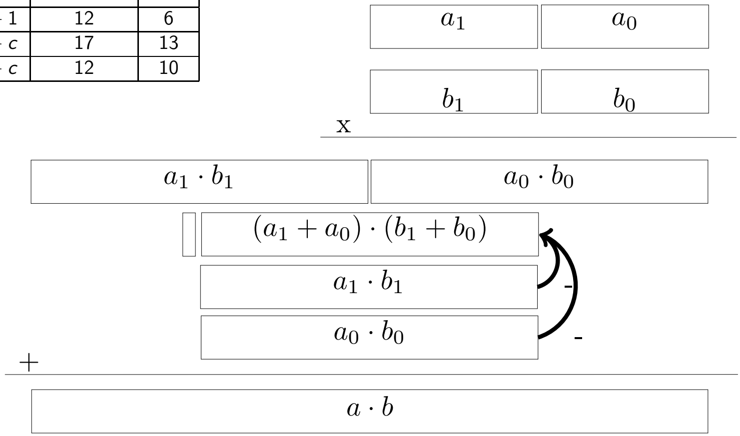
	SCB	RSCB	OSCB
$2^{256} - c$	38	35	37



1. By scheduling of the operations

One level Karatsuba multiplication vs. one level schoolbook multiplication

	Karatsuba	SCB
$2^{127}-1$	12	6
$2^{127}-c$	17	13
$2^{128}-c$	12	10

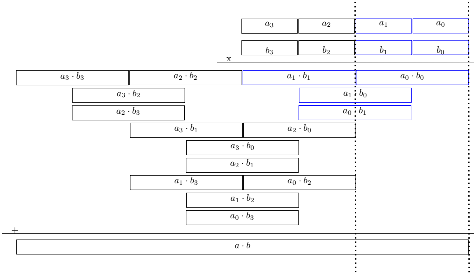


2. By making optimization

Register optimization

```
    //...
    movq 8*0(%r8), %rax
    mulq 8*0(%r9)
    movq %rax, %rbx
    movq %rdx, %rsi
    movq 8*1(%r8), %rax
    mulq 8*1(%r9)
    movq %rax, %r10
    movq %rdx, %r11
    movq 8*1(%r8), %rax
    mulq 8*0(%r9)
    addq %rax, %rsi
    adcq %rdx, %r10
    adcq $0, %r11
    movq 8*0(%r8), %rax
    mulq 8*1(%r9)
    addq %rax, %rsi
    adcq %rdx, %r10
    adcq $0, %r11
    movq %rbx, 8*0(%rdi)
    movq %rsi, 8*1(%rdi)
    //...
```

Listing 1 : $< GF(2^{255} - c), \ * >$



3. By using special instructions

The instruction `cmovxx`

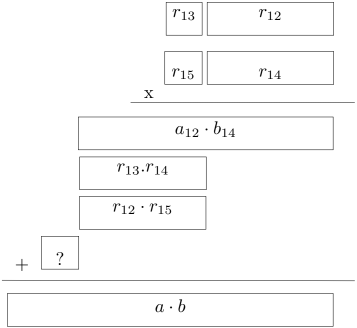
Conditional Move

```
    //...
    movq %r12, %rax
    mulq %r14
    movq $0, %rbp
    cmp $0, %r13
    cmovz %rbp, %r14
    cmp $0, %r15
    cmovz %rbp, %r12
    andq %r13, %r15
    addq %r12, %rdx
    adcq $0, %rbp
    addq %r14, %rdx
    adcq %r15, %rbp
    //...
```

Listing 2 : $< GF(2^{128} - c), \ * >$

if $r_{13} = 0$ **then**
  Return 0.
else
  Return r_{14}.
end

if $r_{12} = 0$ **then**
  Return 0.
else
  Return r_{15}.
end



3. By using special instructions

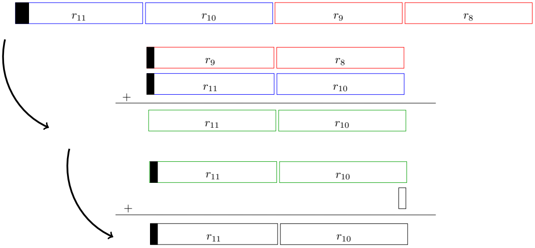
The instruction `btxx`

Bit Test and Reset

```
    //...
    /*r11, r10, r9, r8*/
    shlq $1, %r11
    btrq $63, %r10
    adcq $0, %r11
    shlq $1, %r10
    btrq $63, %r9
    adcq $0, %r10

    addq %r8, %r10
    adcq %r9, %r11

    btrq $63, %r11
    adcq $0, %r10
    adcq $0, %r11
    //...
```



Listing 3 :
$< GF(2^{127}-1), \; * >$

Faster compact Diffie-Hellman: Endomorphisms on the x−line
C. Costello, H. Hisil, and B. Smith

3. By using special instructions

Comparing with the MPFQ library $< GF(2^{127}-1), * >$

33 instructions, 6 clock cyles

```
    //...
    /*r11, r10, r9, r8*/
    shlq $1, %r11
    btrq $63, %r10
    adcq $0, %r11
    shlq $1, %r10
    btrq $63, %r9
    adcq $0, %r10

    addq %r8, %r10
    adcq %r9, %r11

    btrq $63, %r11
    adcq $0, %r10
    adcq $0, %r11
    //...
```

Listing 4 : `My schoolbook's code reduction part`

45 instructions, 9 clock cycles

```
    //... /*r11, r10, r9, r8*/
    movq $9223372036854775807, %rax
    movq %r9, %r12
    andq %rax, %r9
    shrq $63, %r12
    movq %r10, %rdx
    shlq $1, %r10
    orq %r10, %r12
    shlq $1, %r11
    shrq $63, %rdx
    orq %r11, %rdx
    addq %r12, %r8
    adcq %rdx, %r9
    movq %r9, %r12
    andq %rax, %r9
    shlq $1, %r12
    adcq $0, %r8
    adcq $0, %r9
    //...
```

Listing 5 : `MPFQ schoolbook's code reduction part`

`https://www.imsc.res.in/~ecc14/slides/hisil.pdf`

Test Results

Timing benchmarks were taken on an Intel Core i7-6500U processor running Ubuntu 14.04.5 LTS with TurboBoost disabled and all cores but one are switched-off (i.e. hyperthreading is disabled). To obtain the executables, we used GNU-gcc version 4.8.4 with the -O2 flag set and GNU assembler version 2.24.

	Karatsuba	Schoolbook (SCB)	Recursive SCB
$2^{127} - 1$	12	6	-
$2^{127} - c$	17	13	-
$2^{128} - c$	12	10	-
$2^{255} - c$	-	46	40
$2^{256} - c$	-	38	34

```c
/*libraries*/
#define TRIAL 100000000000
int main() {
  long long st, fn;
  st = cpucycles();
  unsigned long an[2], bn[2], cn[2];
  an[0] = (unsigned long) rand() * (unsigned long) rand();
  an[1] = (unsigned long) rand() * (unsigned long) rand();
  bn[0] = (unsigned long) rand() * (unsigned long) rand();
  bn[1] = (unsigned long) rand() * (unsigned long) rand();
  cn[0] = (unsigned long) rand() * (unsigned long) rand();
  cn[1] = (unsigned long) rand() * (unsigned long) rand();
  unsigned long int i;
  for (i = 0; i < TRIAL; i++) {
    mul127_scb_v01(an, bn, cn);
    an[0] = bn[1];
    an[1] = cn[0];
    bn[0] = an[1];
    bn[1] = cn[1];
    cn[0] = an[1];
    cn[1] = bn[0];
  }
  fn = cpucycles();
  double first = ((double) fn - st) / TRIAL;
  st = cpucycles();
  for (i = 0; i < TRIAL; i++) {
    mul127_scb_test(an, bn, cn);
    an[0] = bn[1];
    an[1] = cn[0];
    bn[0] = an[1];
    bn[1] = cn[1];
    cn[0] = an[1];
    cn[1] = bn[0];
  }
  fn = cpucycles();
  double second = ((double) fn - st) / TRIAL;
  printf("net clock cycle : %lf\n\n", first - second);
  return 1;
}
```

Listing 6 : A performance test