GF 256

Martin Kozeny
MATH 4530: Intro Cryptography
Spring 2011 University of New Orleans

March 13, 2011

1 Implementation

Details of implementation are written in comments in attached source code. **Adding** is implemented as simple XOR, **multiplying** as multiplying members of the first polynom with members of the second polynom divided by modulo polynom (dividing was used instead of rewriting rules, beacause it is also used in inverse finding). Algorithm of **inverse finding** is the same as in previous homework - Extended Euclidean Algorithm - , but in implementation are used polynoms instead of integers.

2 Testing

These tests were verified manually and also by [1].

2.1 Adding

For adding, I tried to evaluate these examples with shown results in table 1.

Table 1: Adding in GF 256

Left operand	Right operand	Expected result	Result
A1	D7	76	76
57	83	D4	D4
4B	C8	83	83
B2	E5	57	57
F1	10	E1	E1
D7	12	C5	C5
07	B9	BE	BE
45	2D	68	68
CC	EE	22	22
AB	CD	66	66
3F	7B	44	44
62	33	51	51
0E	9D	93	93

2.2 Multiplying

For multiplying, I tried to evaluate these examples with shown results in table 2.

Table 2: Multiplying in GF 256

Left operand	Right operand	Expected result	Result
A1	D7	D0	D0
57	83	C1	C1
4B	C8	89	89
B2	E5	76	76
F1	10	89	89
D7	12	6A	6A
07	B9	2	2
45	2D	2C	2C
CC	EE	A	A
AB	CD	BB	BB
3F	7B	87	87
62	33	28	28
0E	9D	E7	E7

2.3 Finding inverses

For finding inverses, I tried to evaluate these examples with shown results in table 3.

Table 3: Finding inverses in GF 256

Operand	Expected result	Result
7E	81	81
4B	13	13
C6	E4	E4
B2	1F	1F
F1	23	23
D7	EA	EA
07	D1	D1
45	31	31
CC	1B	1B
AB	4A	4A
3F	19	19
62	AF	AF
0E	E5	E5

3 Conclusion

Provided tests show, that implemented algorithms work correctly.

References

[1] GF256 calculator, `http://computing.unn.ac.uk/staff/cgmb3/teaching/cryptography/GF256Applet/GF256Applet.html`.