LAMPIRAN A

Tabel *Galois Field Multiplication*

M2	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0x00	0x02	0x04	0x06	0x08	0x0a	0x0c	0x0e	0x10	0x12	0x14	0x16	0x18	0x1a	0x1c	0x1e
1	0x20	0x22	0x24	0x26	0x28	0x2a	0x2c	0x2e	0x30	0x32	0x34	0x36	0x38	0x3a	0x3c	0x3e
2	0x40	0x42	0x44	0x46	0x48	0x4a	0x4c	0x4e	0x50	0x52	0x54	0x56	0x58	0x5a	0x5c	0x5e
3	0x60	0x62	0x64	0x66	0x68	0x6a	0x6c	0x6e	0x70	0x72	0x74	0x76	0x78	0x7a	0x7c	0x7e
4	0x80	0x82	0x84	0x86	0x88	0x8a	0x8c	0x8e	0x90	0x92	0x94	0x96	0x98	0x9a	0x9c	0x9e
5	0xa0	0xa2	0xa4	0xa6	0xa8	0xaa	0xac	0xae	0xb0	0xb2	0xb4	0xb6	0xb8	0xba	0xbc	0xbe
6	0xc0	0xc2	0xc4	0xc6	0xc8	0xca	0xcc	0xce	0xd0	0xd2	0xd4	0xd6	0xd8	0xda	0xdc	0xde
7	0xe0	0xe2	0xe4	0xe6	0xe8	0xea	0xec	0xee	0xf0	0xf2	0xf4	0xf6	0xf8	0xfa	0xfc	0xfe
8	0x1b	0x19	0x1f	0x1d	0x13	0x11	0x17	0x15	0x0b	0x09	0x0f	0x0d	0x03	0x01	0x07	0x05
9	0x3b	0x39	0x3f	0x3d	0x33	0x31	0x37	0x35	0x2b	0x29	0x2f	0x2d	0x23	0x21	0x27	0x25
A	0x5b	0x59	0x5f	0x5d	0x53	0x51	0x57	0x55	0x4b	0x49	0x4f	0x4d	0x43	0x41	0x47	0x45
B	0x7b	0x79	0x7f	0x7d	0x73	0x71	0x77	0x75	0x6b	0x69	0x6f	0x6d	0x63	0x61	0x67	0x65
C	0x9b	0x99	0x9f	0x9d	0x93	0x91	0x97	0x95	0x8b	0x89	0x8f	0x8d	0x83	0x81	0x87	0x85
D	0xbb	0xb9	0xbf	0xbd	0xb3	0xb1	0xb7	0xb5	0xab	0xa9	0xaf	0xad	0xa3	0xa1	0xa7	0xa5
E	0xdb	0xd9	0xdf	0xdd	0xd3	0xd1	0xd7	0xd5	0xcb	0xc9	0xcf	0xcd	0xc3	0xc1	0xc7	0xc5
F	0xfb	0xf9	0xff	0xfd	0xf3	0xf1	0xf7	0xf5	0xeb	0xe9	0xef	0xed	0xe3	0xe1	0xe7	0xe5

Gambar 5.1: Tabel *Galois Multiplication* dengan pengali 2

M3	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0x00	0x03	0x06	0x05	0x0c	0x0f	0x0a	0x09	0x18	0x1b	0x1e	0x1d	0x14	0x17	0x12	0x11
1	0x30	0x33	0x36	0x35	0x3c	0x3f	0x3a	0x39	0x28	0x2b	0x2e	0x2d	0x24	0x27	0x22	0x21
2	0x60	0x63	0x66	0x65	0x6c	0x6f	0x6a	0x69	0x78	0x7b	0x7e	0x7d	0x74	0x77	0x72	0x71
3	0x50	0x53	0x56	0x55	0x5c	0x5f	0x5a	0x59	0x48	0x4b	0x4e	0x4d	0x44	0x47	0x42	0x41
4	0xc0	0xc3	0xc6	0xc5	0xcc	0xcf	0xca	0xc9	0xd8	0xdb	0xde	0xdd	0xd4	0xd7	0xd2	0xd1
5	0xf0	0xf3	0xf6	0xf5	0xfc	0xff	0xfa	0xf9	0xe8	0xeb	0xee	0xed	0xe4	0xe7	0xe2	0xe1
6	0xa0	0xa3	0xa6	0xa5	0xac	0xaf	0xaa	0xa9	0xb8	0xbb	0xbe	0xbd	0xb4	0xb7	0xb2	0xb1
7	0x90	0x93	0x96	0x95	0x9c	0x9f	0x9a	0x99	0x88	0x8b	0x8e	0x8d	0x84	0x87	0x82	0x81
8	0x9b	0x98	0x9d	0x9e	0x97	0x94	0x91	0x92	0x83	0x80	0x85	0x86	0x8f	0x8c	0x89	0x8a
9	0xab	0xa8	0xad	0xae	0xa7	0xa4	0xa1	0xa2	0xb3	0xb0	0xb5	0xb6	0xbf	0xbc	0xb9	0xba
A	0xfb	0xf8	0xfd	0xfe	0xf7	0xf4	0xf1	0xf2	0xe3	0xe0	0xe5	0xe6	0xef	0xec	0xe9	0xea
B	0xcb	0xc8	0xcd	0xce	0xc7	0xc4	0xc1	0xc2	0xd3	0xd0	0xd5	0xd6	0xdf	0xdc	0xd9	0xda
C	0x5b	0x58	0x5d	0x5e	0x57	0x54	0x51	0x52	0x43	0x40	0x45	0x46	0x4f	0x4c	0x49	0x4a
D	0x6b	0x68	0x6d	0x6e	0x67	0x64	0x61	0x62	0x73	0x70	0x75	0x76	0x7f	0x7c	0x79	0x7a
E	0x3b	0x38	0x3d	0x3e	0x37	0x34	0x31	0x32	0x23	0x20	0x25	0x26	0x2f	0x2c	0x29	0x2a
F	0x0b	0x08	0x0d	0x0e	0x07	0x04	0x01	0x02	0x13	0x10	0x15	0x16	0x1f	0x1c	0x19	0x1a

Gambar 5.2: Tabel *Galois Multiplication* dengan pengali 3

M9	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0x00	0x09	0x12	0x1b	0x24	0x2d	0x36	0x3f	0x48	0x41	0x5a	0x53	0x6c	0x65	0x7e	0x77
1	0x90	0x99	0x82	0x8b	0xb4	0xbd	0xa6	0xaf	0xd8	0xd1	0xca	0xc3	0xfc	0xf5	0xee	0xe7
2	0x3b	0x32	0x29	0x20	0x1f	0x16	0x0d	0x04	0x73	0x7a	0x61	0x68	0x57	0x5e	0x45	0x4c
3	0xab	0xa2	0xb9	0xb0	0x8f	0x86	0x9d	0x94	0xe3	0xea	0xf1	0xf8	0xc7	0xce	0xd5	0xdc
4	0x76	0x7f	0x64	0x6d	0x52	0x5b	0x40	0x49	0x3e	0x37	0x2c	0x25	0x1a	0x13	0x08	0x01
5	0xe6	0xef	0xf4	0xfd	0xc2	0xcb	0xd0	0xd9	0xae	0xa7	0xbc	0xb5	0x8a	0x83	0x98	0x91
6	0x4d	0x44	0x5f	0x56	0x69	0x60	0x7b	0x72	0x05	0x0c	0x17	0x1e	0x21	0x28	0x33	0x3a
7	0xdd	0xd4	0xcf	0xc6	0xf9	0xf0	0xeb	0xe2	0x95	0x9c	0x87	0x8e	0xb1	0xb8	0xa3	0xaa
8	0xec	0xe5	0xfe	0xf7	0xc8	0xc1	0xda	0xd3	0xa4	0xad	0xb6	0xbf	0x80	0x89	0x92	0x9b
9	0x7c	0x75	0x6e	0x67	0x58	0x51	0x4a	0x43	0x34	0x3d	0x26	0x2f	0x10	0x19	0x02	0x0b
A	0xd7	0xde	0xc5	0xcc	0xf3	0xfa	0xe1	0xe8	0x9f	0x96	0x8d	0x84	0xbb	0xb2	0xa9	0xa0
B	0x47	0x4e	0x55	0x5c	0x63	0x6a	0x71	0x78	0x0f	0x06	0x1d	0x14	0x2b	0x22	0x39	0x30
C	0x9a	0x93	0x88	0x81	0xbe	0xb7	0xac	0xa5	0xd2	0xdb	0xc0	0xc9	0xf6	0xff	0xe4	0xed
D	0x0a	0x03	0x18	0x11	0x2e	0x27	0x3c	0x35	0x42	0x4b	0x50	0x59	0x66	0x6f	0x74	0x7d
E	0xa1	0xa8	0xb3	0xba	0x85	0x8c	0x97	0x9e	0xe9	0xe0	0xfb	0xf2	0xcd	0xc4	0xdf	0xd6
F	0x31	0x38	0x23	0x2a	0x15	0x1c	0x07	0x0e	0x79	0x70	0x6b	0x62	0x5d	0x54	0x4f	0x46

Gambar 5.3: Tabel *Galois Multiplication* dengan pengali 9

M11	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0x00	0x0b	0x16	0x1d	0x2c	0x27	0x3a	0x31	0x58	0x53	0x4e	0x45	0x74	0x7f	0x62	0x69
1	0xb0	0xbb	0xa6	0xad	0x9c	0x97	0x8a	0x81	0xe8	0xe3	0xfe	0xf5	0xc4	0xcf	0xd2	0xd9
2	0x7b	0x70	0x6d	0x66	0x57	0x5c	0x41	0x4a	0x23	0x28	0x35	0x3e	0x0f	0x04	0x19	0x12
3	0xcb	0xc0	0xdd	0xd6	0xe7	0xec	0xf1	0xfa	0x93	0x98	0x85	0x8e	0xbf	0xb4	0xa9	0xa2
4	0xf6	0xfd	0xe0	0xeb	0xda	0xd1	0xcc	0xc7	0xae	0xa5	0xb8	0xb3	0x82	0x89	0x94	0x9f
5	0x46	0x4d	0x50	0x5b	0x6a	0x61	0x7c	0x77	0x1e	0x15	0x08	0x03	0x32	0x39	0x24	0x2f
6	0x8d	0x86	0x9b	0x90	0xa1	0xaa	0xb7	0xbc	0xd5	0xde	0xc3	0xc8	0xf9	0xf2	0xef	0xe4
7	0x3d	0x36	0x2b	0x20	0x11	0x1a	0x07	0x0c	0x65	0x6e	0x73	0x78	0x49	0x42	0x5f	0x54
8	0xf7	0xfc	0xe1	0xea	0xdb	0xd0	0xcd	0xc6	0xaf	0xa4	0xb9	0xb2	0x83	0x88	0x95	0x9e
9	0x47	0x4c	0x51	0x5a	0x6b	0x60	0x7d	0x76	0x1f	0x14	0x09	0x02	0x33	0x38	0x25	0x2e
A	0x8c	0x87	0x9a	0x91	0xa0	0xab	0xb6	0xbd	0xd4	0xdf	0xc2	0xc9	0xf8	0xf3	0xee	0xe5
B	0x3c	0x37	0x2a	0x21	0x10	0x1b	0x06	0x0d	0x64	0x6f	0x72	0x79	0x48	0x43	0x5e	0x55
C	0x01	0x0a	0x17	0x1c	0x2d	0x26	0x3b	0x30	0x59	0x52	0x4f	0x44	0x75	0x7e	0x63	0x68
D	0xb1	0xba	0xa7	0xac	0x9d	0x96	0x8b	0x80	0xe9	0xe2	0xff	0xf4	0xc5	0xce	0xd3	0xd8
E	0x7a	0x71	0x6c	0x67	0x56	0x5d	0x40	0x4b	0x22	0x29	0x34	0x3f	0x0e	0x05	0x18	0x13
F	0xca	0xc1	0xdc	0xd7	0xe6	0xed	0xf0	0xfb	0x92	0x99	0x84	0x8f	0xbe	0xb5	0xa8	0xa3

Gambar 5.4: Tabel *Galois Multiplication* dengan pengali 11

M13	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0x00	0x0d	0x1a	0x17	0x34	0x39	0x2e	0x23	0x68	0x65	0x72	0x7f	0x5c	0x51	0x46	0x4b
1	0xd0	0xdd	0xca	0xc7	0xe4	0xe9	0xfe	0xf3	0xb8	0xb5	0xa2	0xaf	0x8c	0x81	0x96	0x9b
2	0xbb	0xb6	0xa1	0xac	0x8f	0x82	0x95	0x98	0xd3	0xde	0xc9	0xc4	0xe7	0xea	0xfd	0xf0
3	0x6b	0x66	0x71	0x7c	0x5f	0x52	0x45	0x48	0x03	0x0e	0x19	0x14	0x37	0x3a	0x2d	0x20
4	0x6d	0x60	0x77	0x7a	0x59	0x54	0x43	0x4e	0x05	0x08	0x1f	0x12	0x31	0x3c	0x2b	0x26
5	0xbd	0xb0	0xa7	0xaa	0x89	0x84	0x93	0x9e	0xd5	0xd8	0xcf	0xc2	0xe1	0xec	0xfb	0xf6
6	0xd6	0xdb	0xcc	0xc1	0xe2	0xef	0xf8	0xf5	0xbe	0xb3	0xa4	0xa9	0x8a	0x87	0x90	0x9d
7	0x06	0x0b	0x1c	0x11	0x32	0x3f	0x28	0x25	0x6e	0x63	0x74	0x79	0x5a	0x57	0x40	0x4d
8	0xda	0xd7	0xc0	0xcd	0xee	0xe3	0xf4	0xf9	0xb2	0xbf	0xa8	0xa5	0x86	0x8b	0x9c	0x91
9	0x0a	0x07	0x10	0x1d	0x3e	0x33	0x24	0x29	0x62	0x6f	0x78	0x75	0x56	0x5b	0x4c	0x41
A	0x61	0x6c	0x7b	0x76	0x55	0x58	0x4f	0x42	0x09	0x04	0x13	0x1e	0x3d	0x30	0x27	0x2a
B	0xb1	0xbc	0xab	0xa6	0x85	0x88	0x9f	0x92	0xd9	0xd4	0xc3	0xce	0xed	0xe0	0xf7	0xfa
C	0xb7	0xba	0xad	0xa0	0x83	0x8e	0x99	0x94	0xdf	0xd2	0xc5	0xc8	0xeb	0xe6	0xf1	0xfc
D	0x67	0x6a	0x7d	0x70	0x53	0x5e	0x49	0x44	0x0f	0x02	0x15	0x18	0x3b	0x36	0x21	0x2c
E	0x0c	0x01	0x16	0x1b	0x38	0x35	0x22	0x2f	0x64	0x69	0x7e	0x73	0x50	0x5d	0x4a	0x47
F	0xdc	0xd1	0xc6	0xcb	0xe8	0xe5	0xf2	0xff	0xb4	0xb9	0xae	0xa3	0x80	0x8d	0x9a	0x97

Gambar 5.5: Tabel *Galois Multiplication* dengan pengali 13

M14	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0x00	0x0e	0x1c	0x12	0x38	0x36	0x24	0x2a	0x70	0x7e	0x6c	0x62	0x48	0x46	0x54	0x5a
1	0xe0	0xee	0xfc	0xf2	0xd8	0xd6	0xc4	0xca	0x90	0x9e	0x8c	0x82	0xa8	0xa6	0xb4	0xba
2	0xdb	0xd5	0xc7	0xc9	0xe3	0xed	0xff	0xf1	0xab	0xa5	0xb7	0xb9	0x93	0x9d	0x8f	0x81
3	0x3b	0x35	0x27	0x29	0x03	0x0d	0x1f	0x11	0x4b	0x45	0x57	0x59	0x73	0x7d	0x6f	0x61
4	0xad	0xa3	0xb1	0xbf	0x95	0x9b	0x89	0x87	0xdd	0xd3	0xc1	0xcf	0xe5	0xeb	0xf9	0xf7
5	0x4d	0x43	0x51	0x5f	0x75	0x7b	0x69	0x67	0x3d	0x33	0x21	0x2f	0x05	0x0b	0x19	0x17
6	0x76	0x78	0x6a	0x64	0x4e	0x40	0x52	0x5c	0x06	0x08	0x1a	0x14	0x3e	0x30	0x22	0x2c
7	0x96	0x98	0x8a	0x84	0xae	0xa0	0xb2	0xbc	0xe6	0xe8	0xfa	0xf4	0xde	0xd0	0xc2	0xcc
8	0x41	0x4f	0x5d	0x53	0x79	0x77	0x65	0x6b	0x31	0x3f	0x2d	0x23	0x09	0x07	0x15	0x1b
9	0xa1	0xaf	0xbd	0xb3	0x99	0x97	0x85	0x8b	0xd1	0xdf	0xcd	0xc3	0xe9	0xe7	0xf5	0xfb
A	0x9a	0x94	0x86	0x88	0xa2	0xac	0xbe	0xb0	0xea	0xe4	0xf6	0xf8	0xd2	0xdc	0xce	0xc0
B	0x7a	0x74	0x66	0x68	0x42	0x4c	0x5e	0x50	0x0a	0x04	0x16	0x18	0x32	0x3c	0x2e	0x20
C	0xec	0xe2	0xf0	0xfe	0xd4	0xda	0xc8	0xc6	0x9c	0x92	0x80	0x8e	0xa4	0xaa	0xb8	0xb6
D	0x0c	0x02	0x10	0x1e	0x34	0x3a	0x28	0x26	0x7c	0x72	0x60	0x6e	0x44	0x4a	0x58	0x56
E	0x37	0x39	0x2b	0x25	0x0f	0x01	0x13	0x1d	0x47	0x49	0x5b	0x55	0x7f	0x71	0x63	0x6d
F	0xd7	0xd9	0xcb	0xc5	0xef	0xe1	0xf3	0xfd	0xa7	0xa9	0xbb	0xb5	0x9f	0x91	0x83	0x8d

Gambar 5.6: Tabel *Galois Multiplication* dengan pengali 14

LAMPIRAN B

Source Code

Algoritma 2.1 Algoritma Enkripsi AES-128

```
#define LENGTH 16
#define NROWS 4
#define NCOLS 4
#define ROUNDS 10
typedef unsigned char byte;

rijndael (byte plaintext[LENGTH], byte ciphertext[LENGTH], byte
   key[LENGTH]) {
int r;
byte state[NROWS][NCOLS];
struct{byte k[NROWS][NCOLS];} rk[ROUNDS + 1];

KeyExpansion(key, rk);
CopyPlaintextToState(state, plaintext);

AddRoundKey(state, rk[0]);

for (r = 1; r <= ROUNDS - 1; r++) {
SubBytes(state);
ShiftRows(state);
MixColumns(state);
AddRoundKey(state, rk[r]);
}
SubBytes(state);
ShiftRows(state);
```

```
AddRoundKey(state, rk[ROUNDS]);

CopyStateToCiphertext(ciphertext, state);
}
```

Algoritma 2.2 Algoritma Dekripsi AES-128

```
#define LENGTH 16
#define NROWS 4
#define NCOLS 4
#define ROUNDS 10
typedef unsigned char byte;

invrijndael (byte ciphertext[LENGTH], byte plaintext[LENGTH],
   byte key[LENGTH]) {
int r;
byte state[NROWS][NCOLS];
struct{byte k[NROWS][NCOLS];} rk[ROUNDS + 1];

KeyExpansion(key, rk);
CopyCiphertextToState(state, ciphertext);

AddRoundKey(state, rk[ROUNDS]);

for (r = 1; r <= ROUNDS - 1; r++) {
invShiftRows(state);
invSubBytes(state);
AddRoundKey(state, rk[ROUNDS-i]);
invMixColumns(state);
}
invShiftRows(state);
```

```
invSubBytes(state);
AddRoundKey(state, rk[0]);

CopyStateToPlaintext(plaintext, state);
}
```

Algoritma 2.3 *Source Code* SIM *writer*

```
#include "SerialCommand.h"
#include <SPI.h>
#include <MFRC522.h>
#include <AESLib.h>
#define RST_PIN         9
#define SS_PIN          10

MFRC522 mfrc522(SS_PIN, RST_PIN);
MFRC522::MIFARE_Key key;

MFRC522::StatusCode status;

const uint8_t pkey[] = {0x66, 0x61, 0x73, 0x69, 0x6C, 0x6B, 0x6F
    , 0x6D, 0x75, 0x6E, 0x6A, 0x5F, 0x32, 0x30, 0x31, 0x33};

uint8_t secret[16];
char *tmpSecret;

SerialCommand sCmd;

void loop() {
  if (!mfrc522.PICC_IsNewCardPresent()) {
```

```
    return;
  }
  if (!mfrc522.PICC_ReadCardSerial()) {
    return;
  }
    sCmd.readSerial();
  }

void setKey() {
  byte enc_secret[16], simTypeBuffer[16];
  char *jenis;
  tmpSecret = sCmd.next();
  generateKey(tmpSecret, secret);
  uint8tobyte(enc_secret, secret);
  Serial.println("Random key:");

  dump_byte_array(enc_secret, 16);
  aes128_enc_single(pkey, (char *) enc_secret);
  writeData(enc_secret, 28);

  Serial.println("Encrypted random key:");
  dump_byte_array(enc_secret, 16);

  jenis = sCmd.next();
  fillBuffer(jenis, simTypeBuffer);
  aes128_enc_single(secret, (char *) simTypeBuffer);

  writeData(simTypeBuffer, 10);

  Serial.println(" ");
  mfrc522.PICC_HaltA(); // Halt PICC
```

```
  mfrc522.PCD_StopCrypto1();  // Stop encryption on PCD
  Serial.println();
  delay(10);
}

void saveData() {

  if(secret == NULL) {
    Serial.println("Pass Not Set");
    return;
  }

  readData(28, datablock, size);
  copydatablock(dec, datablock);
  aes128_dec_single(pkey, (char *) dec);
  bytetouint8(secret, dec);

  nosim = sCmd.next();
  fillBuffer(nosim, noBuffer);
  aes128_enc_single(secret, (char *) noBuffer);
  Serial.print("Nomor SIM\t:  ");Serial.println(nosim);
  dump_byte_array(noBuffer, 18);

  fname = sCmd.next();
  fillBuffer(fname, fnameBuffer);
  aes128_enc_single(secret, (char *) fnameBuffer);

  lname = sCmd.next();
  fillBuffer(lname, lnameBuffer);
  aes128_enc_single(secret, (char *) lnameBuffer);
```

```
  alamat = sCmd.next();
  fillBuffer(alamat, alamatBuffer);
  aes128_enc_single(secret, (char *) alamatBuffer);

  point = sCmd.next();
  fillBuffer(point, pointBuffer);
  aes128_enc_single(secret, (char *) pointBuffer);

  writeData(noBuffer, 4);
  writeData(fnameBuffer, 5);
  writeData(lnameBuffer, 6);
  writeData(alamatBuffer, 8);
  writeData(pointBuffer, 9);

  mfrc522.PICC_HaltA(); // Halt PICC
  mfrc522.PCD_StopCrypto1();  // Stop encryption on PCD
  Serial.println();
  delay(10);
}

void fillBuffer(char *source, byte *dest) {
  for (int i = 0; i < 16; i++) {
  dest[i] = source[i] ;
  }
  Serial.println((char *) dest);
}

void generateKey(char *buff, uint8_t *secret) {
  int len =  getLength(buff);
  for (int i = 0; i < 16; i++) {
  secret[i] = (buff[random(0,len)] + random(255))%255;
```

```
  }
}

void writeData(byte *buffer, int block) {
  MFRC522::StatusCode status;
  Serial.println(F("Authenticating using key A..."));
  status = mfrc522.PCD_Authenticate(MFRC522::
     PICC_CMD_MF_AUTH_KEY_A, block, &key, &(mfrc522.uid));
  if (status != MFRC522::STATUS_OK) {
    Serial.print(F("PCD_Authenticate() failed: "));
    Serial.println(mfrc522.GetStatusCodeName(status));
    return;
  }
  else Serial.println(F("PCD_Authenticate() success: "));
    status = mfrc522.MIFARE_Write(block, buffer, 16);
    if (status != MFRC522::STATUS_OK) {
    Serial.print(F("Gagal menyimpan data: "));
    Serial.println(mfrc522.GetStatusCodeName(status));
    return;
  }
  else Serial.println(F("Berhasil menyimpan data: "));
}

void read_data(byte sector, byte block, byte *datablock, byte
   size){
  mfrc522.PICC_DumpMifareClassicSectorToSerial(&(mfrc522.uid), &
     key, sector);
  Serial.println();
  Serial.print(F("Reading data from block ")); Serial.print(
     block);
  Serial.println(F(" ..."));
```

```
  status = (MFRC522::StatusCode) mfrc522.MIFARE_Read(block,
     datablock, &size);
  if (status != MFRC522::STATUS_OK) {
    Serial.print(F("MIFARE_Read() failed: "));
    Serial.println(mfrc522.GetStatusCodeName(status));
  }
  Serial.print(F("Data pada block")); Serial.print(block);
     Serial.println(F(" dalam desimal:"));
  dump_byte_array(datablock, 18); Serial.println();
  Serial.println();
}

void readInformation() {
  byte block[] = {4, 5, 6, 8, 9, 10, 28, 29};
  char *info[] = {"Nomor SIM\t: ", "Nama\t\t: ", " " , "Region\t
     \t: ", "Point\t\t: ",  "Jenis SIM\t\t: "};
  byte dec[16], embedded[16];
  byte datablock[36];
  byte size = sizeof(datablock);
  uint8_t secret[16];

  Serial.println("Embedded Key:");
  uint8tobyte(embedded, pkey);
  dump_byte_array(embedded, 16);
  Serial.println("");
  readData(block[6], datablock, size);
  copydatablock(dec, datablock);
  Serial.println("Encrypted Random Key:");
  dump_byte_array(dec, 16);
  aes128_dec_single(pkey, (char *) dec);
  Serial.println("Decrypted Random Key:");
```

```
dump_byte_array(dec, 16);
bytetouint8(secret, dec);
Serial.println();
Serial.println("************************ Data SIM
    ************************");

for (int i = 0; i < 6; i++) {
  readData(block[i], datablock, size);
  copydatablock(dec, datablock);
  aes128_dec_single(secret, (char *) dec);
  Serial.print(info[i]);
  if(i == 1) {
    Serial.print((char *) dec);
  } else {
    Serial.println((char *) dec);
  }
}

Serial.println
    ("*************************************************************")
    ;
Serial.println();
Serial.println("Data dalam blok\n");
for (int i = 0; i < 6; i++) {
  Serial.print("Blok " );Serial.print(block[i]);Serial.println
      (": ");
  readData(block[i], datablock, size);
  copydatablock(dec, datablock);
  Serial.println("Encrypted:");
  dump_byte_array(dec, 16);
  aes128_dec_single(secret, (char *) dec);
```

```
    Serial.println("Decrypted:");
    dump_byte_array(dec, 16);
    Serial.println("");
  }

  Serial.println(" ");
  mfrc522.PICC_HaltA(); // Halt PICC
  mfrc522.PCD_StopCrypto1();  // Stop encryption on PCD
  Serial.println();
  delay(10);
}

void readData(byte block, byte *datablock, byte size) {
  status = mfrc522.PCD_Authenticate(MFRC522::
     PICC_CMD_MF_AUTH_KEY_A, block, &key, &(mfrc522.uid));
  mfrc522.MIFARE_Read(block, datablock, &size);
  if (status != MFRC522::STATUS_OK) {
    Serial.print(F("MIFARE_Read() failed: "));
    Serial.println(mfrc522.GetStatusCodeName(status));
  }
}

void fetchData() {

  readData(block[6], datablock, size);
  copydatablock(dec, datablock);
  aes128_dec_single(pkey, (char *) dec);
  bytetouint8(secret, dec);

  for (int i = 0; i < 6; i++) {
    readData(block[i], datablock, size);
```

```
    copydatablock(dec, datablock);
    aes128_dec_single(secret, (char *) dec);
    Serial.print(info[i]);
    Serial.print((char *) dec);
  }

  Serial.println(" ");
  mfrc522.PICC_HaltA();
  mfrc522.PCD_StopCrypto1();
  Serial.println();
  delay(10);
}
```

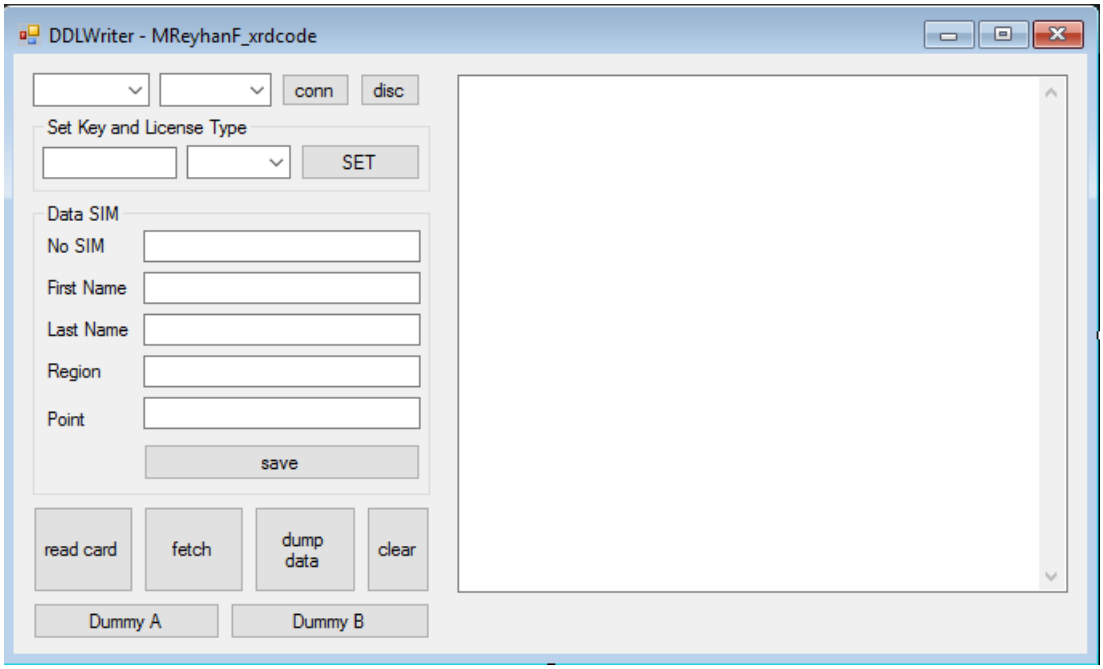
Algoritma 2.4 *Source Code* SIM *reader*

```
#include <SPI.h>
#include <MFRC522.h>
#include <AESLib.h>

#define RELAY1  4
#define RST_PIN         9
#define SS_PIN          10

MFRC522 mfrc522(SS_PIN, RST_PIN);
MFRC522::MIFARE_Key key;

byte datablock[36];
byte buffer1[16];
byte trailerBlock = 7;
MFRC522::StatusCode status;
```

```
const uint8_t pkey[] = {0x66, 0x61, 0x73, 0x69, 0x6C, 0x6B, 0x6F
   , 0x6D, 0x75, 0x6E, 0x6A, 0x5F, 0x32, 0x30, 0x31, 0x33};

const int LEDr = 7;
const int LEDg = 6;
const int button = 2;

int timeOut = 6;

void loop() {
  byte block[] = {4, 5, 6, 8, 9, 10, 28, 29};
  byte sector = 0;
  byte dec[16];
  byte typedec[16];
  byte size = sizeof(datablock);
  uint8_t secret[16];

  if (!mfrc522.PICC_IsNewCardPresent()) {
    timeOut = timeOut-1;
    delay(1000);
    if (timeOut != 0) {
      return;
    } else {
      digitalWrite(LEDr, HIGH);
      digitalWrite(LEDg, LOW);
      digitalWrite(RELAY1,0);
    }
  }

  timeOut = 2;
```

```
  if (!mfrc522.PICC_ReadCardSerial()) return;

  read_data(block[6],datablock,size);
  copydatablock(dec, datablock);
  aes128_dec_single(pkey, (char *) dec);
  bytetouint8(secret, dec);

  read_data(block[0],datablock,size);
  copydatablock(dec, datablock);
  aes128_dec_single(secret, (char *) dec);

  read_data(block[5],datablock,size);
  copydatablock(typedec, datablock);
  aes128_dec_single(secret, (char *) typedec);

  if (check(dec) && typedec[0] == 0x41) {
    digitalWrite(LEDg, HIGH);
    digitalWrite(LEDr, LOW);
    digitalWrite(RELAY1,1);
  } else {
    Serial.println("Gagal");
    digitalWrite(RELAY1,0);
    digitalWrite(LEDg, LOW);
    digitalWrite(LEDr, HIGH);
  }

  mfrc522.PCD_StopCrypto1();
}

void read_data(byte block, byte *datablock, byte size) {
```

```
  status = mfrc522.PCD_Authenticate(MFRC522::
     PICC_CMD_MF_AUTH_KEY_A, block, &key, &(mfrc522.uid));
  mfrc522.MIFARE_Read(block, datablock, &size);
  if (status != MFRC522::STATUS_OK) {
    Serial.print(F("MIFARE_Read() failed: "));
    Serial.println(mfrc522.GetStatusCodeName(status));
  }
}

boolean check(byte *dec) {
  for (int i = 0; i < 16; i++) {
    if((dec[i] < 0x30 || dec[i] > 0x7A) && dec[i] != 0x20 && dec
       [i] != 0x00 ) {
    return false;
    }
  }
  return true;
}
```



Gambar 5.7: Desain *windows form software* SIM *writer*

Algoritma 2.5 *Source Code Software* SIM *writer* - Home.cs

```
using System;
using System.IO.Ports;
using System.Windows.Forms;

namespace DDLWriter
{
  public partial class Home : Form
  {
    MyPorts myPort;
    bool fetchClicked = false;
    int count = 0;

    public Home()
    {
```

```
  InitializeComponent();
}

private void Home_Load(object sender, EventArgs e)
{
  getPort();
  getBaudRate();
  getSIM();
  myPort = new MyPorts(serialPort1);
  myPort.setTexbox(textOut);

  timer1.Tick += new EventHandler(timer1_Tick);
  timer1.Interval = 100;
}

private void getPort()
{
  string[] ports = SerialPort.GetPortNames();
  cmbPorts.DataSource = ports;
}

private void getSIM()
{
  string[] sim = { "A", "C"};
  cmbSIM.DataSource = sim;
}

private void getBaudRate()
{
  int[] baud = {9600};
  cmbBaudRate.DataSource = baud;
```

```
}

private void button1_Click(object sender, EventArgs e)
{
  myPort.sendCommand("sk " + txtKey.Text + " " + cmbSIM.Text
      );
}

private void timer1_Tick(object sender, EventArgs e)
{
  string existing = serialPort1.ReadExisting();
  if(existing != "")
  {
    if(existing.Contains("fetch"))
    {
      getFetchData(existing);
    }
    textOut.AppendText(existing);
    Console.WriteLine(existing);
    count++;
  }
}

private void getFetchData(string txt)
{
  string[] data = txt.Split(',');
  textNosim.Text = data[1];
  textNama.Text = data[2];
  cmbSIM.SelectedValue = data[3];
  textReg.Text = data[4];
  textPoint.Text = data[5];
```

```
}

private void btn_connect_Click(object sender, EventArgs e)
{
  textOut.Text = "";
  myPort.setBaudrate(int.Parse(cmbBaudRate.Text));
  myPort.setPorts(cmbPorts.Text);
  myPort.establishConnection();
  timer1.Start();
}

private void btn_close_Click(object sender, EventArgs e)
{
  myPort.closeConnection();
  timer1.Enabled = false;
}

private void btn_read_Click(object sender, EventArgs e)
{
  myPort.sendCommand("r");
}

private void button2_Click(object sender, EventArgs e)
{
  string nosim = textNosim.Text;
  string nama = textNama.Text;
  string lname = textLast.Text;
  string region = textReg.Text;
  string point = textPoint.Text;
  myPort.sendCommand("save " + nosim + " " + nama + " " +
      lname + " " + region + " " + point);
```

```
      MessageBox.Show("Letakkan kartu pada writer");
    }

    private void btnClear_Click(object sender, EventArgs e)
    {
      textOut.Text = "";
    }

    private void btnDump_Click(object sender, EventArgs e)
    {
      myPort.sendCommand("dump");
    }

    private void button3_Click(object sender, EventArgs e)
    {
      if(!fetchClicked)
      {
        myPort.sendCommand("fetch");
        MessageBox.Show("Now, Please insert the card");
        fetchClicked = true;
      } else
      {
        textOut.Text = "";
        fetchClicked = false;
      }

    }
  }
}
```

Algoritma 2.6 *Source Code Software* SIM *writer* - MyPorts.cs

```
using System;

namespace DDLWriter
{
  class MyPorts
  {
    private System.IO.Ports.SerialPort serialPort = null;
    private string port = null;
    private int baudrate = -1;
    private System.Windows.Forms.TextBox textB;

    public MyPorts(System.IO.Ports.SerialPort serialPort)
    {
      this.serialPort = serialPort;
    }

    public MyPorts(string port, int baudrate, System.Windows.
       Forms.TextBox textB)
    {
      this.baudrate = baudrate;
      this.port = port;
      this.textB = textB;
    }

    public void setBaudrate(int baudrate)
    {
      this.baudrate = baudrate;
    }

    public void setPorts(string port)
    {
```

```
  this.port = port;
}

public void setTexbox(System.Windows.Forms.TextBox textB)
{
  this.textB = textB;
}

public bool establishConnection()
{
  try
  {
    if (baudrate != -1 && port != null && textB != null)
    {
      serialPort.PortName = port;
      serialPort.BaudRate = baudrate;
      serialPort.Open();
      textB.AppendText("Connected" + Environment.NewLine);
      return true;
    }
    else
    {
      return false;
    }
  } catch (Exception ex)  {
    textB.Text += "Connecting Fail : " + ex.Message +
       Environment.NewLine;
    return false;
  }
}
```

```
    public void sendCommand(string command)
    {
      try
      {
        serialPort.WriteLine(command);
      } catch (Exception ex)  {
        textB.AppendText(Environment.NewLine + "Error :" + ex.
            Message);
      }
    }

    public void closeConnection()
    {
      try
      {
        serialPort.Close();
        textB.AppendText("Disconnected" + Environment.NewLine);
      } catch (Exception ex)  {
        textB.AppendText(ex.Message + Environment.NewLine);
      }
    }
  }
}
```