Tugas 6 (Individu)

Decipher AES Rijndael

Nama : Ramadhan Kalih Sewu

NPM: 1806148826

A. Proses Input

Teks dan key diambil dari permintaan tugas. Sebelum masuk ke fungsi AES, input akan diubah kedalam bentuk decimal. Program dilengkapi dengan validasi output untuk mengecek kebenaran decipher.

```
text = {'19', 'a0', '9a', 'e9'; '3d', 'f4','c6','f8';'e3','e2','8d','48';'be','2b','2a','08'};
key = {'a0', '88', '23', '2a'; 'fa', '54','a3','6c';'fe','2c','39','76';'17','b1','39','05'};

text = reshape(hex2dec(text), size(text));
key = reshape(hex2dec(key), size(key));

cipher = AES(text, key)
decipher = decryptAES(cipher, key)

if (decipher == text) display('Decipher Valid!');
else                  display('Invalid Output');
end
```

B. Dekripsi AES

Pada dekripsi AES, **urutan** dari **langkah** subfungsi akan berubah dengan fungsi enkripsi. Walaupun begitu, kunci yang dipakai untuk tiap ronde akan sama dengan kunci pada enkripsi. Berikut adalah penjelasan proses dekripsi:

- Meminta KeySchedule untuk meminta kunci tiap ronde
- Initial round berupa bitxor cipher terhadap key ronde 11
- Lakukan rutin subfungsi AES selama 10 ronde dimana pada ronde terakhir tidak perlu Inverse Mix Column

Kunci yang digunakan pada tiap ronde untuk subfungsi bitxor dimulai dari kunci ronde 11 – 10 – 9 - ... – 1. Penggunaan kunci dalam perintah bitxor ini berkebalikan dengan encipher.

```
function [state] = decryptAES(cipher, key)
    % get every round key
    keyRnd = KeySchedule(key);
    % initial round
    state = bitxor(cipher, cell2mat(keyRnd(:,11)));
    for round = 10:-1:1
        % Shift Rows
        state = InvShiftRows(state);
        % Byte Substitution
        state = InvSubBytes(state);
        % XOR with Round Key
        state = bitxor(state, cell2mat(keyRnd(:,round)));
        % Mix Columns
        if (round ~= 1)
            state = InvMixColumns(state);
        end
    end
end
```

Subfungsi Shift Rows decipher akan melakukan hal berkebalikan dengan encipher, berikut rutinitas yang dilakukan:

- Row 1 – Tidak di rotate
- Row 2 – Rotate 1 byte ke kanan
- Row 3 – Rotate 2 byte ke kanan
- Row 4 – Rotate 3 byte ke kanan

```
function [state] = InvShiftRows(state)
    for i = 2:4
        state(i, :) = circshift(state(i, :), [0, i-1]);
    end
end
```

Substitusi Bytes pada decipher melakukan hal yang sama seperi pada encipher. Yang membedakan hanyalah matrix substitusi yang digunakan. Pada decipher kita menggunakan matrix sbox yang telah di-inverse. Pada dasarnya, fungsi ini akan mengolah tiap byte pada state. Berikut yang dilakukan:

- Mengambil 4-bit MSB atau 1-hexadecimal MSB sebagai Indeks Baris
- Mengambil 4-bit LSB atau 1-hexadecimal LKB sebagai Indeks Kolom
- Mensubstitusi byte pada state dengan elemen pada Inverse S Box (menggunakan indeks yang didapatkan dari langkah diatas).

Referensi matrix: https://en.wikipedia.org/wiki/Rijndael_S-box

```
function [res] = InvSubBytes(state)
    invsbox = { '52' '09' '6a' 'd5' '30' '36' 'a5' '38' 'bf' '40' 'a3' '9e' '81' 'f3' 'd7' 'fb';
                '7c' 'e3' '39' '82' '9b' '2f' 'ff' '87' '34' '8e' '43' '44' 'c4' 'de' 'e9' 'cb';
                '54' '7b' '94' '32' 'a6' 'c2' '23' '3d' 'ee' '4c' '95' '0b' '42' 'fa' 'c3' '4e';
                '08' '2e' 'a1' '66' '28' 'd9' '24' 'b2' '76' '5b' 'a2' '49' '6d' '8b' 'd1' '25';
                '72' 'f8' 'f6' '64' '86' '68' '98' '16' 'd4' 'a4' '5c' 'cc' '5d' '65' 'b6' '92';
                '6c' '70' '48' '50' 'fd' 'ed' 'b9' 'da' '5e' '15' '46' '57' 'a7' '8d' '9d' '84';
                '90' 'd8' 'ab' '00' '8c' 'bc' 'd3' '0a' 'f7' 'e4' '58' '05' 'b8' 'b3' '45' '06';
                'd0' '2c' '1e' '8f' 'ca' '3f' '0f' '02' 'c1' 'af' 'bd' '03' '01' '13' '8a' '6b';
                '3a' '91' '11' '41' '4f' '67' 'dc' 'ea' '97' 'f2' 'cf' 'ce' 'f0' 'b4' 'e6' '73';
                '96' 'ac' '74' '22' 'e7' 'ad' '35' '85' 'e2' 'f9' '37' 'e8' '1c' '75' 'df' '6e';
                '47' 'f1' '1a' '71' '1d' '29' 'c5' '89' '6f' 'b7' '62' '0e' 'aa' '18' 'be' '1b';
                'fc' '56' '3e' '4b' 'c6' 'd2' '79' '20' '9a' 'db' 'c0' 'fe' '78' 'cd' '5a' 'f4';
                '1f' 'dd' 'a8' '33' '88' '07' 'c7' '31' 'b1' '12' '10' '59' '27' '80' 'ec' '5f';
                '60' '51' '7f' 'a9' '19' 'b5' '4a' '0d' '2d' 'e5' '7a' '9f' '93' 'c9' '9c' 'ef';
                'a0' 'e0' '3b' '4d' 'ae' '2a' 'f5' 'b0' 'c8' 'eb' 'bb' '3c' '83' '53' '99' '61';
                '17' '2b' '04' '7e' 'ba' '77' 'd6' '26' 'e1' '69' '14' '63' '55' '21' '0c' '7d'};
    res = arrayfun( @(x) invsbox(floor(x/16) + 1, mod(x, 16) + 1),state );
    res = reshape(hex2dec(res), size(state));
end
```

Mix Column pada decipher akan menggunakan Rijndael Galois Multiplication dengan Field yang telah ditentukan. Field nya berbeda dengan yang digunakan pada encipher. Saya menggunakan fungsi bawaan matlab untuk membentuk sebuah galois field yang sesuai dengan AES rijndael dengan jumlah m = 8 dan polinomial: $x^8 + x^4 + x^3 + x + 1$

```
function [state] = InvMixColumns(state)
    warning('off');
    % Rijndael Galois Field for AES Decryption
    RGF = [14 11 13 9; 9 14 11 13; 13 9 14 11; 11 13 9 14];
    % Galois Field 2^8-1 Polynomial: X^8 + X^4 + X^3 + X + 1
    gfA = gf(RGF,8,'D8+D4+D3+D+1');
    gfH = gf(state,8,'D8+D4+D3+D+1');
    % Rijndael Galois Multiplication
    state = gfA * gfH;
    state = double(state.x);
end
```

C. Output

Output decipher maupun cipher dari fungsi AES yang saya buat akan menghasilkan sebuah nilai decimal dengan matrix ukuran 4x4. Di akhir jalannya program, hasil decipher akan dicocokkan dengan teks pada proses input, kode ditunjukkan pada gambar di bagian A. Fungsi decipher berhasil mengembalikan ke bentuk semula.