PEMANFAATAN MIKROKONTROLLER STM32F407VG SEBAGAI *CRYPTOPROCESSOR* PADA SISTEM KOMUNIKASI BERBASIS SUARA

Novita Angraini

Lembaga Sandi Negara
Jl. Harsono RM no 70 Ragunan Pasarminggu Jakarta Selatan 12550
Email : novita.angraini@bssn.go.id

Abstrak

Ancaman terhadap teknologi komunikasi secara umum berkembang lebih cepat jika dibandingkan oleh metode keamanan untuk mengantisipasi ancaman tersebut. Untuk mengantisipasi hal-hal tersebut dapat menggunakan peralatan komunikasi yang menerapkan kriptografi, yang bisa disebut sebagai modul kriptografi. Modul kriptografi harus memiliki spesifikasi yang sesuai dengan kriteria keamanan. FIPS PUB 140-2 yang saat ini diberlakukan, membedakan security level modul kriptografi menjadi 4 (empat) tingkatan, yaitu security level 1, 2, 3 dan 4. Salah satu syarat untuk dapat memenuhi security level 3 dan level 4 pada modul kriptografi adalah entry atau output CSP (critical security parameter) dapat dilakukan dengan menggunakan port yang secara fisik terpisah dari port lain.

Oleh karena itu, pada penelitian kali ini akan dirancang cryptoprocessor sebagai media penyimpanan CSP pada modul kriptografi yang terpisah dari prosesor utama pada sistem komunikasi berbasis suara. Cryptoprocessor yang akan dirancang menggunakan mikrokontroller STM32F407VG dengan menerapkan algoritma enkripsi AES-128.

Kata kunci: *mikrokontroller STM32F407VG, AES-128, Cryptoprocessor*

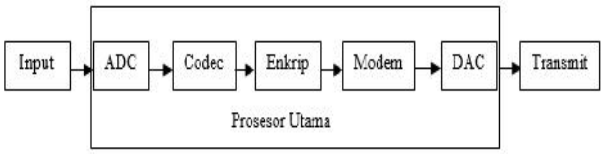
1. Pendahuluan

Teknologi komunikasi merupakan teknologi yang mendukung dalam pertukaran informasi. Komunikasi itu sendiri bisa diartikan sebagai proses pertukaran informasi dan data antara yang satu dengan lainnya. Informasi merupakan aset penting bagi pertumbuhan dan keberlangsungan suatu organisasi. Oleh karena itu, informasi harus dijaga keutuhannya dari modifikasi pihak yang tidak sah dan harus tersedia saat dibutuhkan. Selain itu, perlu dipahami bahwa tidak semua informasi boleh diketahui oleh semua pihak. Ada beberapa jenis informasi yang jika diketahui oleh pihak yang tidak berhak mengetahuinya dapat menimbulkan kerawanan bagi organisasi. Untuk mencegah hal-hal tersebut dapat diatasi dengan menggunakan peralatan komunikasi yang menerapkan kriptografi, yang bisa disebut sebagai modul kriptografi.

Kriptografi merupakan studi matematika yang berhubungan dengan aspek pengamanan informasi seperti kerahasiaan (*confidentiality*), integritas data (*data integrity*), otentikasi entitas (*entity authentication*), dan otentikasi keaslian data (*data origin authentication*). Sistem kriptografi dibagi menjadi tiga bagian, yaitu: sistem simetrik, asimetrik, dan fungsi hash. Sistem simetrik sendiri terbagi menjadi dua, yaitu *stream cipher* dan *block cipher*. *Block cipher* adalah suatu fungsi yang memetakan *n*-bit blok teks terang ke *n*-bit blok teks sandi dengan *n* merupakan panjang blok[2].

Mengacu pada FIPS PUB 140, definisi modul kriptografi adalah sekumpulan *software*, *hardware* dan/atau *firmware* yang mengimplementasikan fungsi-fungsi keamanan (misalnya, algoritma kriptografi dan pembangkit kunci) dan yang termasuk di dalam lingkup kriptografi [4]. Modul kriptografi dapat digunakan untuk memenuhi kebutuhan konfidensialitas atau kerahasiaan, integritas maupun ketersediaan dari informasi.

Pada umumnya skema sistem komunikasi berbasis suara yang sudah menggunakan proses enkripsi didalamnya adalah sebagai berikut:



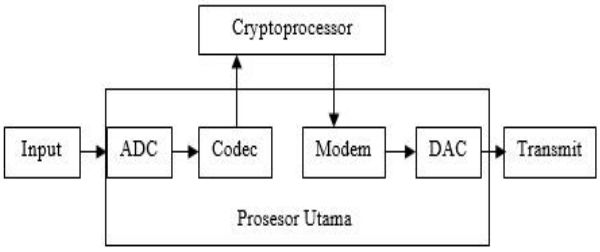
Gambar 1. *Skema Sistem Komunikasi Berbasis Suara*

Pada gambar 1 dapat dilihat bahwa sistem komunikasi berbasis suara yang pada umumnya menjalankan semua prosesnya pada prosesor utama. Untuk mencegah pengungkapan dan modifikasi terhadap informasi yang tersimpan di dalam modul kriptografi, maka modul kriptografi harus memiliki spesifikasi yang sesuai dengan kriteria keamanan. Hal ini penting karena suatu kesalahan yang tidak terdeteksi dalam mendesain modul kriptografi dapat berpotensi mempengaruhi kinerja fungsi kriptografi. Penggunaan modul kriptografi yang memiliki spesifikasi sesuai dengan kriteria keamanan dapat menjamin keamanan dari informasi yang disimpan atau diprosesnya. FIPS PUB 140-2 yang saat ini diberlakukan, membedakan *security level* modul kriptografi menjadi 4 (empat) tingkatan, yaitu *security*

level 1, 2, 3 dan 4. Setiap *security level* memiliki beberapa persyaratan keamanan (*security requirement*) yang harus dipenuhi[4]. Persyaratan keamanan yang tertuang di dalam *security level* berkaitan dengan tujuan desain dan implementasi modul kriptografi yang aman.

Salah satu syarat untuk dapat memenuhi security level 3 dan level 4 pada modul kriptografi adalah *entry* atau *output* CSP (critical security parameter) dapat dilakukan dengan menggunakan *port* yang secara fisik terpisah dari *port* lain, atau *interface* yang secara logik terpisah menggunakan *trusted channel* dari *interface* lainnya[4].

CSP (critical security parameter) adalah parameter yang berkaitan dengan pengamanan yaitu kunci (kunci kriptografi dan data otentikasi seperti *password* atau *Personal Identification Number* (PIN)) dan algoritma kriptografi. Kunci dan algoritma kriptografi merupakan informasi yang berkaitan langsung dengan layanan keamanan yang diberikan oleh modul kriptografi. Keamanan informasi yang diproses menggunakan modul kriptografi tergantung pada keamanan dari kunci dan algoritma kriptografi yang tersimpan di dalam modul tersebut. Oleh karena itu, *cryptoprocessor* sebagai media penyimpanan CSP pada modul kriptografi harus dipisah dari prosesor utama. Berikut skema sistem komunikasi berbasis suara dengan *cryptoprocessor* terpisah.



Gambar 2. *Skema Sistem Komunikasi Berbasis Suara dengan Cryptoprocessor*

Pada penelitian ini, akan digunakan mikrokontroller STM32F407VG sebagai *cryptoprocessor* dengan menerapkan algoritma enkripsi AES-128. Berdasarkan NIST Recommendation, algoritma AES masih aman digunakan hingga tahun 2030[5]. *Cryptoprocessor* ini juga nantinya akan langsung diintegrasikan dengan sistem komunikasi berbasis suara yang menggunakan prosesor utama mikrokontroller STM32F407VG. Komunikasi antar mikrokontroller yang digunakan menggunakan komunikasi UART.

Berikut landasan teori yang digunakan pada penelitian ini adalah sebagai berikut:

A. Advanced Encryption Standard (AES)[3]

Algoritma *block cipher* Rijndael yang diajukan oleh Joan Daemen dan Vincent Rijmen terpilih sebagai *Advanced Encryption Standard* (AES) pada tahun 2001. Algoritma ini adalah algoritma simetrik standar berbasis *block cipher* yang mengenkripsi 128-bit blok *input* menjadi 128-bit blok *output*. Algoritma AES menggunakan panjang kunci yang beragam, yaitu 128, 192, dan 256 bit. Berdasarkan panjang kuncinya, AES dikelompokkan menjadi AES-128, AES-192, dan AES-256. AES menggunakan jumlah *round* Nr beragam bergantung dari panjang kunci yang digunakan[3]. Tabel 1 merupakan perbandingan jumlah *round* pada AES berdasarkan panjang kunci yang digunakan:

Tabel 1. *Kombinasi kunci-blok-round AES*

	Key Length (Nk words)	*Block Size (Nb words)*	*Number of Rounds (Nr)*
AES-128	4	4	10
AES-192	6	4	12
AES-256	8	4	14

Untuk memperoleh *cipher* pada algoritma AES, pada blok *input* dilakukan operasi *AddRoundKey*(), kemudian dilakukan transformasi berdasarkan fungsi *round* sebanyak 10, 12 atau 14 kali (tergantung panjang kunci yang digunakan)[3]. Adapun transformasi dalam fungsi *round* sebagai berikut:

1. Substitusi *byte* menggunakan *S-Box* (*S-box*).
   Transformasi *SubBytes()* adalah substitusi *byte nonlinear* yang mengoperasikan setiap *byte* dari *state* menggunakan tabel substitusi (*S-box*) yang *invertible.*
2. *Shifting rows* dari *state array* (*ShiftRow*)
   ShiftRows merupakan suatu transformasi yang merotasi *bytes* pada tiga baris terakhir dari *state* sebanyak beberapa *bytes.*
3. *Mixing data* dalam setiap kolom dari *state array* (*MixColumn*)
   Transformasi *MixColumns*() beroperasi pada *state* kolom per kolom yang direpresentasikan dalam bentuk perkalian matriks pada GF(2^8).
4. Penambahan *round key* ke *state* (*AddRoundKey*)
   Transformasi *AddRoundKey()* merupakan transformasi *round key* dengan *state* menggunakan operasi XOR

Kemudian untuk *round* terakhir sedikit berbeda dengan *Nr-1* round pertama yaitu tidak terdapat transformasi *MixColumn*().

Untuk penjadwalan kunci pada algoritma AES akan dijelaskan di bawah ini:
Kunci *round* diturunkan dari kunci penyandian melalui proses *key schedule*. Proses ini terdiri dari dua buah komponen, yaitu Ekspansi Kunci dan Pemilihan Kunci *Round*. Prinsip dasar dari proses ini adalah:

1. Jumlah total dari bit kunci *round* sama dengan panjang blok *input* dikalikan dengan jumlah *round* ditambah 1 (misalnya, untuk panjang blok *input* sebesar 128-bit dengan jumlah *round* 10 *round*, maka

total bit kunci *round* yand dibutuhkan adalah 1408 bit).

2. Mengekspansi kunci penyandian.
   Kunci *round* diperoleh dari kunci yang telah diekspansi dengan cara berikut: Kunci *round* terdiri dari *Nb word* pertama, bagian kedua dari *Nb word* kedua dan demikian selanjutnya.

Ekspansi Kunci [3]

Algoritma Rijndael menggunakan kunci penyandian, *K*, dan menjalankan rutin kunci ekspansi untuk menghasilkan *key schedule*. Ekspansi kunci secara keseluruhan menghasilkan Nb(Nr + 1) *word*. Algoritma tersebut memerlukan himpunan inisial dari Nb *word* dan setiap Nr *round* memerlukan Nb *word* dari data kunci. *Key schedule* yang dihasilkan terdiri dari sebuah *array* linear dengan panjang 4-*byte word*, dinotasikan oleh [*wi*], dengan nilai *i* berada di dalam interval $0 \leq i <$ Nb(Nr + 1).

Ekspansi dari kunci *input* menjadi *key schedule* diproses dengan fungsi *SubWord*() dan *RotWord*(). *SubWord*() adalah sebuah fungsi yang memproses 4-*byte input word* dan mensubstitusikannya ke dalam *S-Box* untuk menghasilkan sebuah *output word*. Fungsi *RotWord*() menggunakan *word* $[a_0, a_1, a_2, a_3]$ sebagai *input*, kemudian menerapkan permutasi siklik dan memberikan nilai balikan *word* $[a_1, a_2, a_3, a_0]$. *Array word* konstanta *round*, Rcon[i], berisikan nilai $[x^{i-1},\{00\},\{00\},\{00\}]$, dengan x^{i-1} merupakan pangkat dari *x* (*x* dinotasikan sebagai {02}) di *field* GF(2^8).

Nk *word* pertama dari kunci yang diekpansi diisi dengan kunci penyandian. Setiap *word* berikutnya, w[i], sama dengan nilai XOR dari *word* sebelumnya, w[i-1] dan posisi *word* Nk terawal, w[i-Nk]. Untuk *word* yang berada pada posisi yang merupakan kelipatan dari Nk, sebuah transformasi dilakukan pada w[i-1] sebelum proses XOR, diikuti oleh XOR dengan konstanta *round*, Rcon[i]. Transformasi ini terdiri dari pergeseran siklik dari *bytes* di dalam *word* (*RotWord*()), diikuti dengan menerapkan tabel *lookup* untuk seluruh 4 *byte* dari *word* (*SubWord*()).

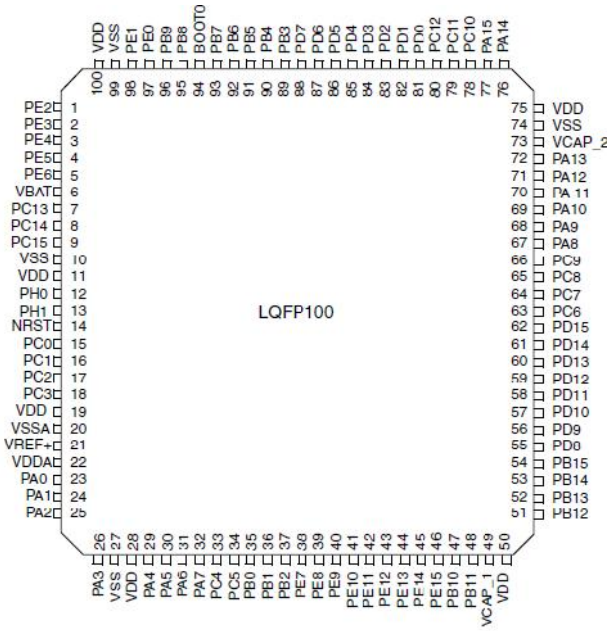
Ekspansi kunci untuk kunci penyandian 256 bit (Nk=8) sedikit berbeda dengan kunci penyandian dengan panjang 128 dan 192 bit. Jika Nk=8 dan nilai i-4 merupakan kelipatan dari Nk, maka *SubWord*() dilakukan terhadap w[i-1] sebelum proses XOR dilakukan.

B. STM32F407VG *(datasheet* STM32F407VG*)*[1]

STM32F407VG didasarkan pada kinerja tinggi ARM®Cortex®-M4 32-bit RISC core yang beroperasi pada frekuensi hingga 168 MHz. Inti Cortex-M4 memiliki Floating Point unit (FPU) single precision yang mendukung semua instruksi pemrosesan data single precision ARM dan tipe data. Ini juga mengimplementasikan satu set lengkap instruksi DSP dan memory protection unit (MPU) yang meningkatkan keamanan aplikasi. STM32F407VG menggabungkan embedded memory berkecepatan tinggi (memori Flash hingga 1 Mbyte, sampai 192 Kbytes dari SRAM), hingga 4 Kbytes dari SRAM cadangan, dan berbagai pilihan Iiput output yang disempurnakan dan peripheral terhubung ke dua APB bus, tiga bus AHB dan matriks bus multi-AHB 32-bit. STM32F407VG menawarkan tiga ADC 12 bit, dua DAC, RTC berdaya rendah, dua belas timer 16-bit tujuan umum termasuk dua timer PWM untuk motor control, dua timer 32-bit tujuan umum. generator bilangan acak (RNG). STM32F407VG juga memiliki antarmuka komunikasi standar dan lanjutan.

Gambar 3. *Mikrokontroller STM32F407VG*

Mikrokontroller STM32F407VG yang digunakan pada penelitian ini adalah STM32F407VG tipe LQFP100 yang memiliki jumlah pin sebanyak 100 pin. Mikrokontroller ini digunakan untuk *cryptoprocessor* dan juga prosesor utama. Gambar 4 merupakan gambar STM32F407VG LQFP100 dengan pinnya.



Gambar 4. *Pin pada STM32F407VG LQFP100*

Metologi yang digunakan pada penelitian ini adalah dengan memanfaatkan komunikasi UART pada STM32F407VG dan prosesor utama. Sebagaimana yang sudah dijelaskan pada FIPS PUB 140-2 bahwa sebaiknya *cryptoprocessor* harus dipisah dari prosesor utama, maka langkah-langkah yang dilakukan adalah sebagai berikut:

1. Implementasi dilakukan dengan menulis program komunikasi UART dan enkripsi AES-128 pada IDE Keil uvision yaitu compiler untuk STM32F407VG dan prosesor utama.
2. *Source code* pada IDE Keil uVision ditulis dengan menggunakan bahasa C.
3. Downloading *source code* ke STM32F407VG dilakukan menggunakan konektor ST Link Debugger.
4. Untuk membuktikan bahwa *source code* yang telah dibuat sudah benar, maka akan dilakukan tes vektor. Tes vektor dilakukan dengan mencocokkan nilai hasil enkripsi yang didapat dengan nilai hasil enkripsi yang telah dipublikasikan.
5. Pengujian hasil implementasi AES-128 dengan melakukan proses enkripsi menggunakan algoritma AES pada STM32F407VG dengan mengaktifkan mode debug dari *cryptoprocessor* dan prosesor utama pada IDE Keil uVision untuk dapat melihat proses yang berjalan pada masing-masing mikrokontroller.

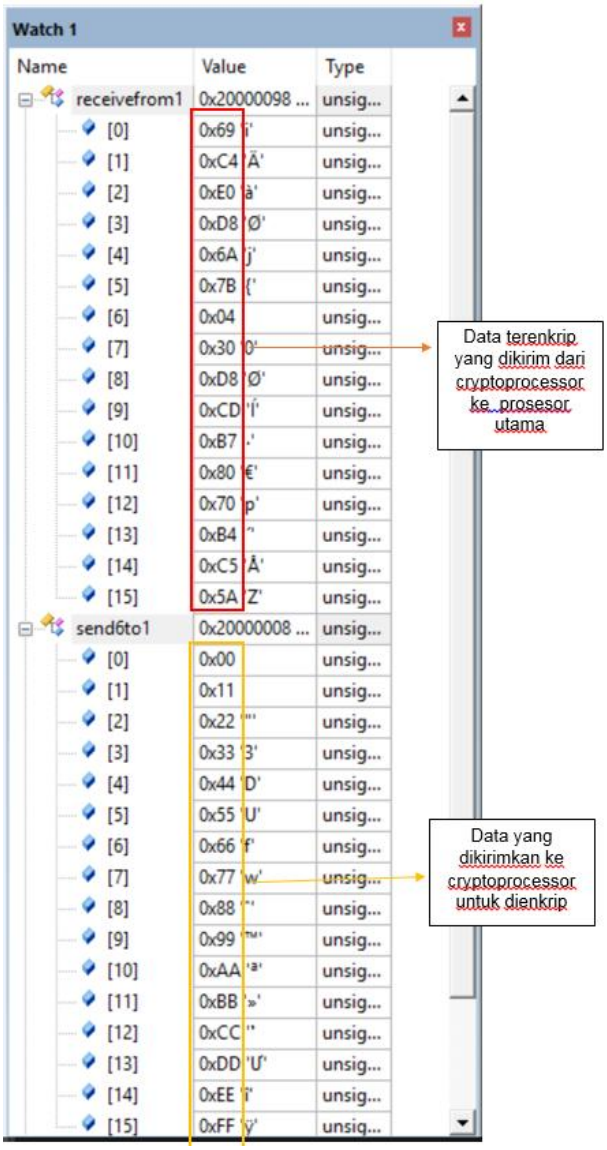
2. Pembahasan

Pada penelitian ini, untuk *cryptoprocessor* menggunakan STM32F407VG dan untuk prosesor utama juga menggunakan STM32F407VG. Komunikasi yang akan digunakan antar mikrokontroller adalah komunikasi UART. Pada prosesor utama menggunakan fungsi USART6 untuk mengirim dan menerima data. Sedangkan pada *cryptoprocessor* menggunakan fungsi USART1 untuk mengirim dan menerima data. Pada prosesor utama menggunakan pin 63 PC6 untuk mengirim data untuk dienkrip ke pin 69 PA10 di *cryptoprocessor*. Kemudian pin 68 PA9 *cryptoprocessor* mengirim data yang sudah terenkrip ke pin 64 PC7 prosesor utama untuk dapat diproses selanjutnya.

Untuk menampilkan hasil enkripsi dan proses kirim terima data antara *cryptoprocessor* dengan prosesor utama, digunakan mode debug dari *cryptoprocessor* dan prosesor utama pada IDE Keil uVision untuk dapat melihat proses yang berjalan pada masing-masing mikrokontroller. Untuk membuktikan bahwa enkripsi AES-128 yang telah dibuat sudah benar, maka akan dilakukan tes vektor. Tes vektor dilakukan dengan mencocokkan nilai hasil enkripsi yang didapat dengan nilai hasil enkripsi yang telah dipublikasikan. Berikut tes vektor AES-128 berdasarkan FIPS 197.

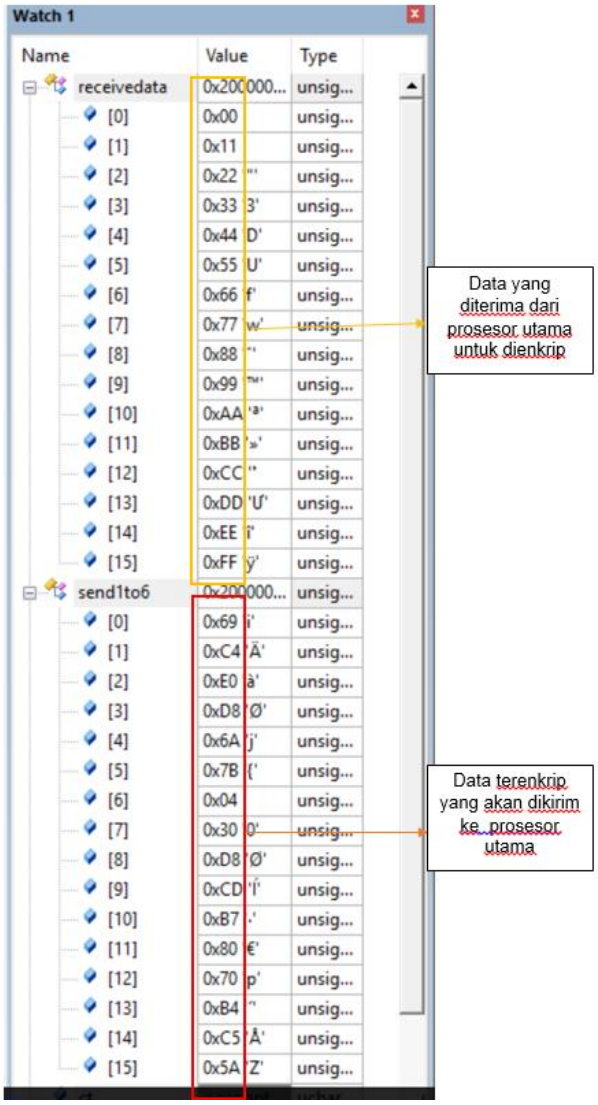
Tes Vektor AES-128
Kunci = 000102030405060708090A0B0C0D0E0F
Plainteks=00112233445566778899AABBCCDDEEFF
Cipherteks=69C4E0D86A7B0430D8CDB78070B4C55A

Tampilan data yang dikirim dan sudah terenkrip pada prosesor utama dapat dilihat pada gambar 5. Tampilan ini merupakan hasil dari pengaktifan mode debug pada prosesor utama.



Gambar 5. *Tampilan mode debug pada prosesor utama*

Tampilan data yang diterima lalu diproses enkripsi oleh *cryptoprocessor* dan dikirim kembali ke prosesor utama untuk dapat diproses selanjutnya dapat dilihat pada gambar 6. Tampilan ini merupakan hasil dari pengaktifan mode debug pada *cryptoprocessor*.



Gambar 6. *Tampilan mode debug pada cryptoprocessor*

3. Kesimpulan

Berdasarkan hasil penelitian ini, dapat disimpulkan hal-hal sebagai berikut:

1. Algoritma AES-128 berhasil diimplementasikan pada mikrokontroller STM32F407VG.
2. Hasil enkripsi AES-128 pada mikrokontroller STM32F407VG sebagai *cryptoprocessor* sudah sesuai dengan nilai tes vektor algoritma AES yang sudah dipublikasikan.
3. Hasil implementasi algoritma AES pada mikrokontroller STM32F407VG ini dapat dijadikan *cryptoprocessor* berbasis sistem embedded untuk dapat diintegrasikan pada sistem komunikasi suara berbasis suara.

Saran pada penelitian ini adalah, pada mikrokontroller STM32F407VG terdapat fungsi RNG (*Random Number Generator*). Sebaiknya fungsi RNG ini dapat diaktifkan pada mikrokontroller STM32F407VG sebagai inputan kunci untuk algoritma enkripsi yang ada. Dengan demikian maka penggunaan mikrokontroller STM32F407VG sebagai *cryptoprocessor* dapat lebih banyak dan lebih optimal.

Daftar Pustaka

[1] Datasheet STM32F405xx STM32F407xx. STMicroelectronics, 2016.
[2] Menezes, Alfred J., Paul C. Van Oorschot, Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC press LLC : Boca Raton, 1997.
[3] NIST, *Federal Information Processing Standards Publication (FIPS) 197*, Springfield : National Institute of Standards and Technology (NIST), 2001.
[4] NIST, FIPS PUB 140-2: *Security Requirements for Cryptographic Modules*, U.S America: National Institute of Standards and Technology, 2001.
[5] www.keylength.com, diakses pada 18 Desember 2017.

Biodata Penulis

Novita Angraini, memperoleh gelar Sarjana Sains Terapan Teknik Persandian (S.S.T.TP), Jurusan Teknik Persandian Sekolah Tinggi Sandi Negara, lulus tahun 2013. Saat ini menjadi pegawai negeri sipil di Deputi Bidang Pengkajian Persandian, Lembaga Sandi Negara.

ISSN : 2302-3805