BAB V

IMPLEMENTASI PERANGKAT LUNAK

5.1 Implementasi Secara Umum

Implementasi merupakan tahap dimana sistem siap dioperasikan pada keadaan sebenarnya dari sini akan diketahui apakah sistem yang dibuat benar-benar dapat menghasilkan tujuan yang diinginkan.

Sebelum program diterapkan dan diimplementasikan, maka program harus *error free* (bebas kesalahan). Kesalahan program yang mungkin terjadi antara lain kesalahan penulisan bahasa, kesalahan waktu proses, atau kesalahan logikal. Setelah program bebas dari kesalahan, program ditest dengan memasukkan data yang akan diolah.

5.2 Batasan Implementasi

Pada bagian ini akan menjelaskan apa yang menjadi batasan implementasi perangkat lunak, antara lain; bahasa yang dipakai serta alasan pemilihannya, lingkungan pengembangan perangkat lunak dan batasan-batasan lain yang juga ditemui selama pengembangan.

5.2.1 Bahasa yang Dipakai

Perangkat lunak yang dipakai untuk sistem Studi Kriptografi dan Implementasi Algoritma *Rivest Code 6* dalam Enkripsi / Dekripsi Data ini adalah Borland Delphi 6.0 yang merupakan salah satu bahasa komputasi teknis yang

sangat populer dan sangat mudah digunakan serta mudah untuk dipahami struktur bahasanya, selain itu Delphi 6.0 mempunyai beberapa fasilitas yang dapat mendukung pemrograman yang akan dibuat.

5.2.2 Lingkungan Pengembangan

Implementasi perangkat lunak untuk sistem ini digunakan dan dikembangkan pada komputer yang memenuhi standar spesifikasi yang disebutkan dibawah ini agar dapat berjalan dengan baik dan sempurna. Spesifikasi yang perlu diperhatikan dalam pengembangan dan penggunaan perangkat lunak ini adalah :

1. Perangkat keras (*Hardware*) minimal yang direkomendasikan, yaitu berupa satu unit komputer dengan prosesor Pentium 233 Mhz atau yang setara, dengan RAM 64 Mb atau lebih, dan ruang hardisk 2 GB.
2. Spesifikasi perangkat lunak (*Software*) yang dibutuhkan antara lain Delphi 6.0 dengan sistem operasi Windows Xp Service Pack 2.

5.2.3 Batasan Sistem

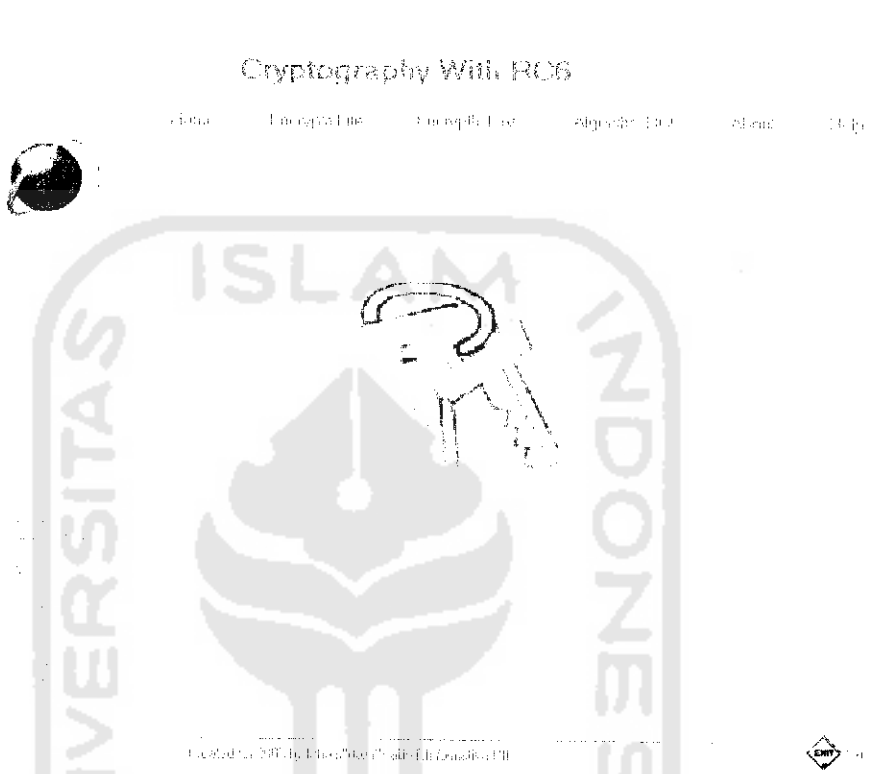
Sistem Studi Kriptografi dan Implementasi Algoritma *Rivest Code 6* dalam Enkripsi / Dekripsi Data ini dibuat untuk proses pembelajaran algoritma RC6 khususnya dan kriptografi secara umum serta mengaplikasikannya ke dalam bentuk program untuk mengenkripsi *file*/data sehingga *file*/data tersebut dapat terjamin keamanannya. Adapun batasan yang diberikan dalam pembuatan sistem ini adalah :

1. Untuk proses enkripsi *file* dimulai dengan memasukkan *key* berupa angka atau karakter ASCII, kemudian memilih *file* yang akan dienkripsi dengan ekstensi bebas seperti *.txt, *.doc, *.jpg, atau lainnya. *File* hasil enkripsi disimpan dalam ekstensi *.cry. Dalam hal pilihan *file*, hanya terbatas pada ekstensi *file* yang telah dimasukan kedalam filter.
2. Proses dekripsi *file* yaitu dimulai dengan memasukkan *key* dekripsi, *key* dekripsi harus sama dengan *key* enkripsi, jika berbeda maka setelah proses dekripsi *file* tidak dapat terbaca. Pilih *file* yang akan didekripsi yaitu dengan ekstensi *.cry, setelah itu simpan *file* dengan *format* semula.
3. Untuk proses enkripsi text menggunakan masukkan *key* angka atau karakter ASCII. Untuk text yang akan dienkripsi yaitu dengan memasukkan karakter ASCII pada memo plaintext. Hasil dari proses enkripsi akan ditampilkan pada memo *encrypt*ext berupa karakter ASCII yang tidak dapat dipahami. Sedang untuk proses dekripsi yaitu dengan memasukkan *key* dekripsi sama dengan *key* enkripsi dan hasil prosesnya ditampilkan pada memo *decrypt*text berupa karakter semula (plaintext).
4. Terdapat menu algoritma RC6, menu ini memberi penjelasan singkat tentang algoritma RC6.

5.3 Implementasi Antarmuka (*Interface*)

Pada sistem Studi Studi Kriptografi dan Implementasi Algoritma *Rivest Code 6* dalam Enkripsi / Dekripsi Data ini terdiri dari 4 *interface* yaitu menu *encrypt file* untuk proses enkripsi/dekripsi *file*, menu *encrypt* text untuk proses

enkripsi/dekripsi text, menu algoritma RC6 yaitu penjelasan tentang algoritma RC6 dan menu *about* program berisi tentang keterangan program.



Gambar 5.1 *Interface* menu utama

5.3.1 *Interface Encrypt File*

Form encrypt file yaitu berfungsi untuk proses enkripsi dan dekripsi *file*. Adapun untuk input datanya adalah *key* untuk proses enkripsi, *file* yang akan dienkripsi, *file* untuk menyimpan hasil dari proses enkripsi dalam bentuk ekstensi *.cry, *file* yang akan didekripsi dan *file* untuk menyimpan hasil dari proses dekripsi. Untuk lebih jelasnya lihat gambar 5.2 *interface encrypt file*

Dalam *form encrypt file* terdapat tombol yang digunakan untuk proses enkrispi antara lain:

1. Tombol *Encrypt File*

   Tombol ini berfungsi untuk proses enkripsi *file* yang telah diinisialisasikan terlebih dahulu *file* yang akan dienkripsi dan tempat *file* untuk menyimpan hasil enkripsi, *key* harus diisi untuk proses enkripsi *file*.

2. Tombol *Decrypt File*

   Tombol ini berfungsi untuk proses dekripsi *file*. Proses ini dimulai dari menginputkan *file* yang akan didekripsi dan tempat menyimpan *file* hasil dari dekripsi. Untuk input *key* dekripsi harus sama dengan *key* enkripsi.

3. Tombol Clear *File*

   Untuk menghapus textbox dan in*form*asi *file*.

4. Tombol *Exit*

   Keluar dari *form encrypt file*

5. *Check* Hapus *File Original*

   Check ini bersifat pilihan, bila *check* dipilih maka akan menghapus *file original* yaitu *file* yang akan dienkripsi. Hal ini dilakukan untuk menguatkan proses enkripsi. Jika *check* tidak dipilih maka *file original* tidak akan dihapus.

6. *Check* Hapus *File* Enkripsi

   Check ini berfungsi untuk menghapus *file* enkripsi yaitu dengan ekstensi *.cry setelah proses dekripsi.

7. In*form*asi *File*

   In*form*asi *file* terdiri dari ukuran *file* dan waktu proses enkripsi dan dekripsi.

8. Tombol *Browse File*

   Tombol ini berfungsi untuk mencari *file* yang akan dienkrip/dideskrip serta menunjukkan letak *file* hasil enkripsi/deskripsi disimpan.

9. Tombol *Up One Level*

   Tombol untuk mengembalikan posisi *explorer* satu *level.*

10. Tombol *Refresh*

   Tombol untuk melakukan *refresh* pada *explorer.*

11. Tombol *Large Icon*

   Tombol agar *icon-icon* pada *explorer* berbentuk *large icon.*

12. Tombol *Small Icon*

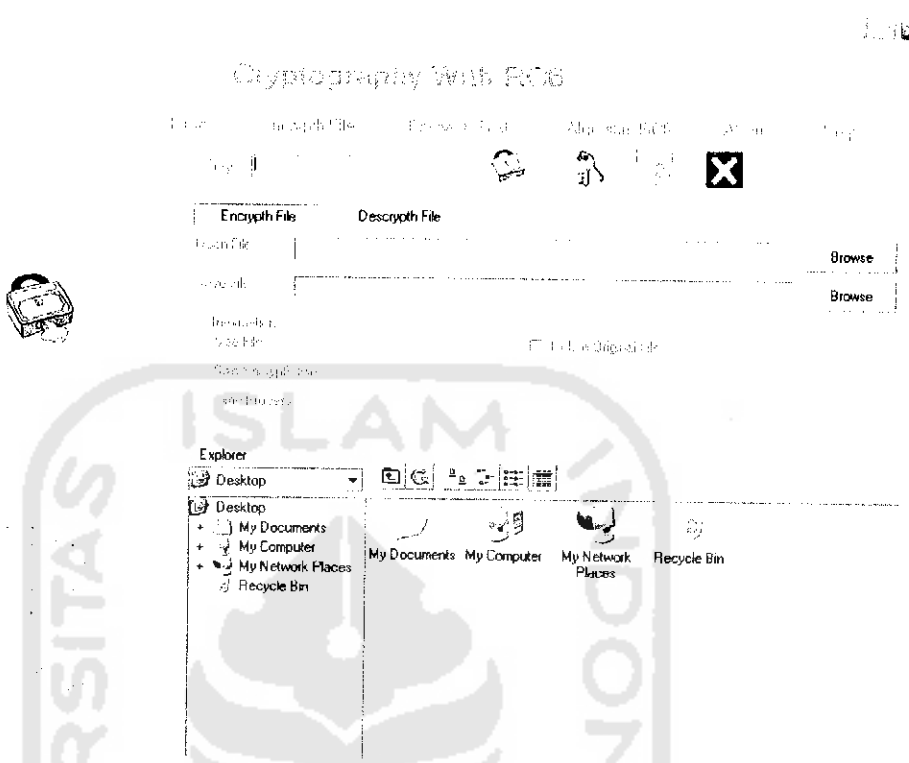
   Tombol agar *icon-i*con pada *explorer* berbentuk *small icon.*

13. Tombol *List*

   Tombol agar posisi isi *explorer* berbentuk *list.*

14. Tombol Detail

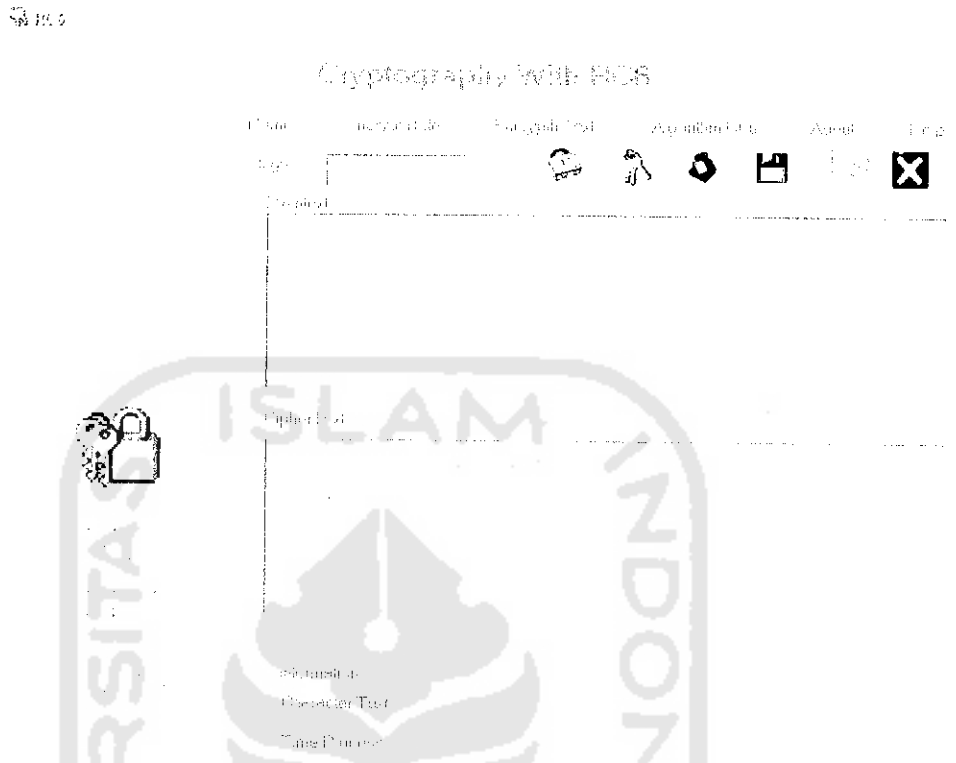
   Tombol untuk menampilkan isi *explorer* secara detail.



Gambar 5.2 *Interface encrypt file*

5.3.2 *Interface Encrypt* Text

Form encrypt text adalah *form* untuk proses enkripsi text. Input data yang diberikan adalah *key* dekripsi dimana harus sama dengan *key* enkripsi, memo *plaintext* yaitu untuk memasukkan text yang akan dienkripsi. Hasil dari proses enkripsi text ditampilkan di memo *encrypt* text berupa karakter yang tidak dapat dipahami sedang hasil dari dekripsi text ditampilkan di memo *decrypt*ext. Untuk lebih jelasnya lihat gambar 5.3 *interface encrypt* text



Gambar 5.3 *Interface encrypt* text

Dalam *form encrypt* text terdapat tombol yang digunakan untuk proses enkrispi dan deskripsi antara lain:

1. Tombol *Encrypt* Text

   Tombol ini berfungsi untuk proses enkripsi text yang telah diinisialisasikan terlebih dahulu text yang akan dienkripsi, *key* harus diisi untuk proses enkripsi text.

2. Tombol *Decrypt* Text

   Tombol ini berfungsi untuk proses dekripsi text. Proses ini dimulai dari menginputkan text yang akan didekripsi . Untuk input *key* dekripsi harus sama dengan *key* enkripsi.

3. Tombol *Open* Text

   Tombol ini berfungsi untuk membuka *file* yang akan dienkripsi ataupun didekripsi.

4. Tombol *Save* Text

   Tombol ini berfungsi untuk menyimpan hasil enkripsi maupun dekripsi *file*.

5. Tombol *Clear* Text

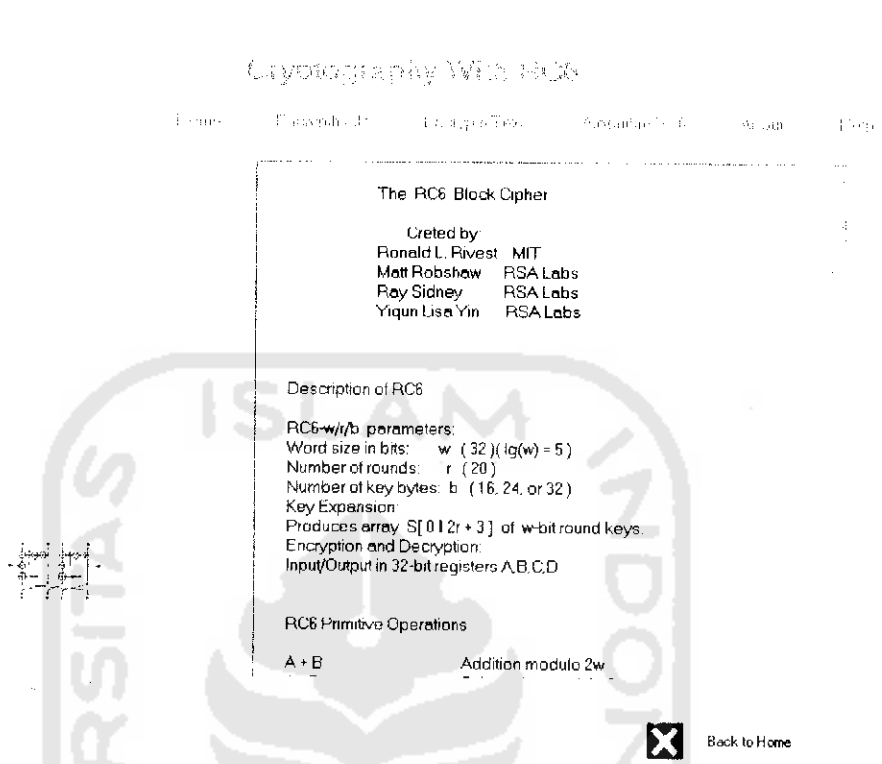
   Untuk menghapus textbox dan informasi *file*.

6. Tombol *Exit*

   Keluar dari *form encrypt* text.

5.3.3 *Interface* Algorithm RC6

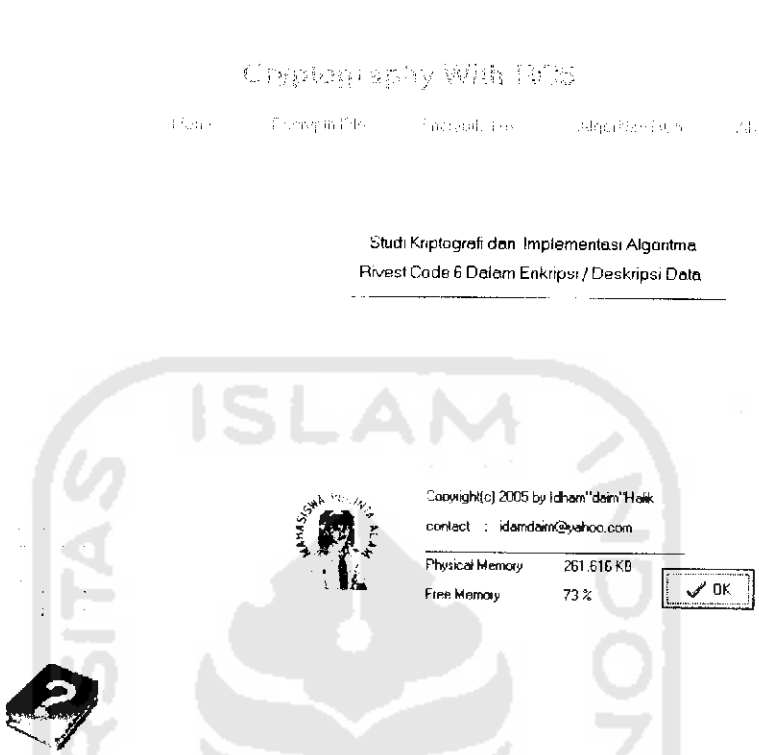
Form algorithm RC6 adalah *form* tentang penjelasan algoritma RC6, yaitu bagaimana proses enkripsi dan dekripsinya. Untuk lebih jelasnya lihat gambar 5.4 *interface algortihm* RC6.



Gambar 5.4 *Interface algorithm RC6*

5.3.4 *Interface* About Program

Pada *form* about program keterangan singkat tentang *programer*. Seperti yang terlihat pada gambar 5.5. *Interface About*



Gambar 5.5 *Interface about*

5.4. *Procedure-procedure* Algoritma RC6 dalam enkripsi/dekripsi

Algoritma RC6 mempunyai prosedur utama dalam proses enkripsi maupun dekripsi yaitu dalam *procedure CalculateSubkey*/menghitung *key*, fungsi *chiper* dan fungsi *decipher*. *Procedure CalculateSubkey* digunakan hanya sekali dalam setiap proses sedangkan fungsi *chiper* dan *decipher* digunakan pada saat *key expansion* dan enkripsi data.

5.4.1 *Procedure CalculateSubkey*/Menghitung Sub*key*

Procedure CalculateSubkey hanya digunakan sekali dalam setiap proses. *Procedure* ini berguna untuk *key expansion* atau membuat *subkey* sehingga P-Array dapat berubah secara kontinuc.

```
procedure TRC6.CalculateSubKeys;
var
      i,j,k   : Integer;
      L       : array[0..15] of LongWord;
      A,B     : LongWord;
begin
      // Copy key ke L
      Move(KeyPtr^, L, KeySize);

      // Inisialisasi tabel s
      S[0] := P32;
      for i := 1 to KeyLength-1 do
            S[i] := S[i-1] + Q32;

      // Acak tabel s dengan key
      i := 0;
      j := 0;
      A := 0;
      B := 0;
      for k := 1 to 3*KeyLength do
      begin
            A := ROL((S[i] + A + B), 3);
            S[i] := A;
            B := ROL((L[j] + A + B), (A + B));
            L[j] := B;
            i := (i + 1) mod KeyLength;
            j := (j + 1) mod 16;
      end;
end;
```

5.4.2. Fungsi *Cipher*

Fungsi ini digunakan dalam *key* expansion dan enkripsi data. Penjelasan alur *procedure* ini dapat dilihat dalam bentuk visualisasi *flowchar* pada gambar 4.4 yang telah di gambarkan sebelumnya.

```
function TRC6.EncipherBlock(var Block): Boolean;
var
      RC6Block     : TRC6Block absolute Block;
      i            : Integer;
      t,u          : LongWord;
      Temp         : LongWord;
begin
      // Inisialisasi blok
      Inc(RC6Block[2], S[0]);
      Inc(RC6Block[4], S[1]);

      // Looping untuk semua elemen s
      for i := 1 to Rounds do
      begin
            // Lakukan Penghitungan
            t := ROL((RC6Block[2] * (2*RC6Block[2] + 1)),lgw);
            u := ROL((RC6Block[4] * (2*RC6Block[4] + 1)),lgw);
            RC6Block[1] := ROL((RC6Block[1] xor t), u) + S[2*i];
            RC6Block[3] := ROL((RC6Block[3] xor u), t) + S[2*i+1];

            // Rotasikan blok
            Temp := RC6Block[1];
            RC6Block[1] := RC6Block[2];
            RC6Block[2] := RC6Block[3];
            RC6Block[3] := RC6Block[4];
            RC6Block[4] :  Temp;
      end;

      // Mengatur pertukaran terakhir
      RC6Block[1] := RC6Block[1] + S[2*Rounds+2];
      RC6Block[3] :  RC6Block[3] + S[2*Rounds+3];

      // This function cannot fail
      Result := TRUE;
end;

end.
```

5.4.3. Fungsi *Decipher*

Fungsi ini digunakan dalam *key* expansion dan dekripsi data. Penjelasan alur fungsi ini dapat dilihat dalam bentuk *visualisasi flowchar* pada gambar 4.5 yang telah di gambarkan sebelumnya.

```
function TRC6.DecipherBlock(var Block): Boolean;
var
      RC6Block     : TRC6Block absolute Block;
      i            : Integer;
      t,u          : LongWord;
      Temp         : LongWord;
begin
      // Inisialisasi blok
      RC6Block[3] := RC6Block[3] - S[2*Rounds+3];
      RC6Block[1] := RC6Block[1] - S[2*Rounds+2];

      // Looping untuk semua elemen s
      for i := Rounds downto 1 do
      begin
            // pertukaran blok
            Temp := RC6Block[4];
            RC6Block[4] := RC6Block[3];
            RC6Block[3] := RC6Block[2];
            RC6Block[2] := RC6Block[1];
            RC6Block[1] := Temp;

            // penghitungan
            u := ROL((RC6Block[4] * (2*RC6Block[4] + 1)),lgw);
            t := ROL((RC6Block[2] * (2*RC6Block[2] + 1)),lgw);
            RC6Block[3] := ROR((RC6Block[3]-S[2*i+1]), t) xor u;
            RC6Block[1] := ROR((RC6Block[1]-S[2*i]), u) xor t;
      end;

      // melakukan pertukaran terakhir
      Dec(RC6Block[4], S[1]);
      Dec(RC6Block[2], S[0]);

      // This function cannot fail
      Result := TRUE;
end;
```