BAB IV

IMPLEMENTASI DAN ANALISIS

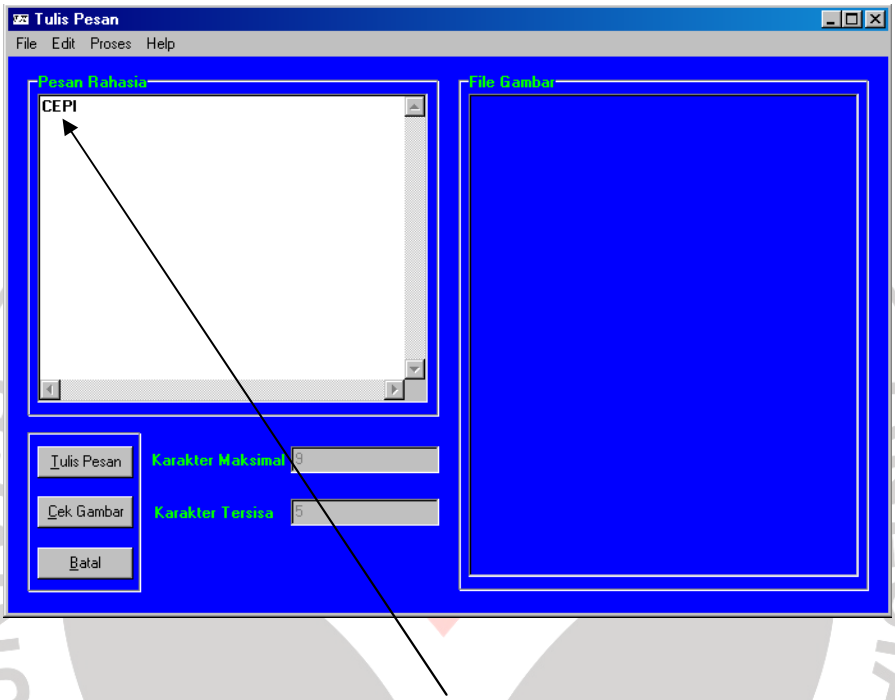
4.1 Menulis dan Membaca Pesan Rahasia

Pada BAB III, kita telah membuat rancangan sistem aplikasi steganografi. Sekarang kita akan mencoba mengimplementasikan aplikasi steganografi tersebut, di mana kita akan mencoba menuliskan dan membaca sebuah pesan rahasia yang disembunyikan pada sebuah *file image bmp 24-bit*. Berikut ini akan dijelaskan langkah-langkah bagaimana cara menuliskan sebuah pesan rahasia pada sebuah *file image bmp 24-bit*.

1. Buka program aplikasi steganografi yang telah dibuat. Setelah program aplikasi steganografi dijalankan, pada *Form* Menu Utama tekan tombol Tulis Pesan, di mana ketika tombol Tulis Pesan ditekan, maka program akan beralih ke *Form* Tulis Pesan.
2. Setelah berada pada *Form* Tulis Pesan, buka sebuah *file image bmp 24-bit*. Misalkan kita ambil sebuah contoh *file image* dengan nama *file* **'TES 1.bmp'** yang tampak seperti gambar di bawah ini:

Gambar 4.1: Contoh *file image bmp 24-bit*

3. Tuliskan pesan rahasia yang ingin kita sisipkan pada *file image* **'TES 1.bmp'** di atas. Misalkan kita akan menuliskan pesan rahasia yang isinya adalah: **'CEPI'**. Perhatikanlah gambar berikut ini:



Pesan Rahasia

4. Setelah menuliskan pesan rahasia, tekan tombol Tulis Pesan. Dengan menekan tombol Tulis Pesan ini, program akan melakukan proses peyembunyian pesan rahasia pada *file image* **'TES 1.bmp'**.
5. Langkah terakhir dari penulisan pesan rahasia ini yaitu dengan menyimpan *file image* yang sudah berisi pesan tadi. Misalkan *file image* yang akan disimpan, kita beri nama *file* **'TES 2.bmp'**.

Setelah melakukan proses penulisan pesan rahasia yang prosesnya dilakukan pada *Form* Tulis pesan, selanjutnya akan dijelaskan langkah-langkah bagaimana cara membaca sebuah pesan rahasia pada sebuah *file image bmp 24-bit* yang sudah berisi pesan rahasia.

1. Pada langkah terakhir dari langkah-langkah proses penulisan pesan rahasia, pada *Form* Tulis Pesan, kita dapat langsung menuju ke *Form* Baca Pesan.
2. Setelah berada pada *Form* Baca Pesan, buka *file image* yang telah diberi nama *file* **'TES 2.bmp'**. Setelah *file image* dibuka, cek *file image* tersebut, apakah *file image* tersebut sudah berisi pesan atau tidak? Karena *file* **'TES 2.bmp'** sudah berisi pesan, tekan tombol Baca Pesan. Dengan menekan tombol Baca Pesan ini, program akan melakukan proses pemisahan pesan rahasia dari *file image* **'TES 2.bmp'**, sehingga pesan rahasia yang tersembunyi bisa dibaca (dalam hal ini pesan rahasianya adalah **'CEPI'**, sebagaimana yang telah dilakukan pada proses penulisan pesan).
3. Setelah proses membaca pesan selesai, kita dapat keluar dari *Form* Baca Pesan dan Program Aplikasi Steganografi.

Steganografi pada media digital *file image* digunakan untuk mengeksploitasi keterbatasan kekuatan sistem penglihatan manusia dengan cara menurunkan kualitas warna pada *file image* yang belum disisipi pesan rahasia. Sehingga dengan keterbatasan tersebut, manusia sulit menemukan gradasi penurunan kualitas warna pada *file image* yang telah disisipi pesan rahasia. Untuk itu, perhatikanlah kedua *file image* berikut ini.

Gambar 4.2: Contoh *File Image* Asli

Gambar 4.3: Contoh *File Image* Rahasia

Dengan melihat kedua *file image* diatas, jelaslah bahwa kita sebagai manusia biasa yang memiliki segala kekurangan dan keterbatasan sangat sulit untuk membedakan mana *file image* yang asli (yang belum disisipi pesan rahasia) dan mana *file image* yang sudah berisi pesan rahasia.

Pada pembahasan selanjutnya akan dilakukan suatu analisis *file image* yang sudah disisipi pesan rahasia sehingga kita dapat mengetahui proses atau cara kerja dari metode yang digunakan, dalam hal ini metode yang digunakan adalah metode *Least Significant Bit* (*LSB*).

4.2 Analisis *File Image* Rahasia

Metode yang digunakan untuk penyembunyian pesan rahasia pada aplikasi ini adalah dengan cara menyisipkan pesan ke dalam bit rendah (*Least Significant Bit*) pada data *pixel* yang menyusun *file image bmp 24-bit*. Metode penyisipan *Least Significant Bit* (*LSB*) ini adalah menyisipi pesan dengan cara mengganti bit ke-8, 16, dan 24 pada representasi biner *file image* dengan representasi biner pesan rahasia yang akan disembunyikan.Untuk itu, pada pembahasan kali ini akan dilakukan suatu analisis *file image* yang sudah disisipi pesan rahasia sehingga kita dapat mengetahui proses atau cara kerja dari metode yang digunakan, dalam hal ini metode yang digunakan yaitu metode *Least Significant Bit* (*LSB*).

Pada analisis *file image bmp 24-bit* yang sudah disisipi pesan rahasia, akan digunakan *WinHex 13.0 SR-1* sebagai *software* pendukung. Dengan menggunakan bantuan *software WinHex 13.0 SR-1*, kita dapat melihat model warna *RGB* dari *file image bmp 24-bit*. Pada *software WinHex 13.0 SR*-1, model warna *RGB* dari *file image bmp 24-bit* diwakili oleh bilangan-bilangan hexadesimal. Untuk menganalisis sebuah *file image bmp 24-bit* yang sudah disisipi pesan rahasia, ambil sebuah contoh *file image bmp 24-bit* dengan warna hitam murni.

Gambar 4.4: Contoh *file image bmp 24-bit* dengan warna hitam murni

Dengan menggunakan *software* pendukung *WinHex 13.0 SR*-1, susunan warna hitam yang diwakili oleh bilangan-bilangan hexadesimal dari contoh *file image bmp 24-bit* diatas, terlihat seperti gambar di bawah ini:

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
00000000	42	4D	C6	00	00	00	00	00	00	00	36	00	00	00	28	00	BMÆ.....6...(.
00000010	00	00	18	00	00	00	02	00	00	00	01	00	18	00	00	00	.....
00000020	00	00	90	00	00	00	00	00	00	00	00	00	00	00	00	00	..▌.....
00000030	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000040	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000050	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000060	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000070	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000080	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000090	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000C0	00	00	00	00	00	00											.....

Gambar 4.5: Susunan bilangan hexadesimal hitam murni

Sekarang kita telah mengetahui susunan bilangan hexadesimal dari warna hitam murni. Selajutnya, tuliskan sebuah pesan rahasia yang akan kita sisipkan pada contoh *image bmp 24-bit* dari **Gambar 4.4** di atas. Misalkan, pesan rahasia yang akan kita sisipkan/sembunyikan adalah kata **'CEPI'**. Berikut ini merupakan *file image bmp 24-bit* dengan warna hitam murni yang sudah berisi pesan rahasia.

Gambar 4.6: *File image bmp 24-bit* dengan warna hitam murni yang sudah berisi pesan rahasia

Sekali lagi, Dengan menggunakan *software* pendukung *WinHex 13.0 SR*-1, susunan warna hitam yang diwakili oleh bilangan-bilangan hexadesimal dari contoh *file image bmp 24-bit* yang sudah disisipi pesan rahasia diatas, dapat dilihat seperti gambar di bawah ini:

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
00000000	42	4D	C6	00	00	00	00	00	00	00	36	00	00	00	28	00	BMÆ.....6...(.
00000010	00	00	18	00	00	00	02	00	00	00	01	00	18	00	00	00	.....
00000020	00	00	90	00	00	00	00	00	00	00	00	00	00	00	00	00	..▌.....
00000030	00	00	00	00	00	00	00	01	00	00	00	00	01	01	00	01	.....
00000040	00	00	00	01	00	01	00	01	00	01	00	00	00	00	00	01	.....
00000050	00	00	01	00	00	01	00	00	00	00	00	00	00	00	00	00	.....
00000060	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000070	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	01	.....
00000080	00	00	00	00	00	00	00	00	01	00	00	00	01	01	00	00	.....
00000090	01	00	00	01	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000A0	00	00	00	00	00	00	00	00	00	00	00	00	00	01	00	00	.....
000000B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000C0	00	01	00	00	01	00											.....

Gambar 4.7: Susunan bilangan hexadesimal hitam murni yang sudah disisipi pesan rahasia

Sekarang pehatikanlah perbedaan susunan bilangan hexadesimal hitam murni dari **Gambar 4.5 (***data original***)** dan **Gambar 4.7**. Selanjutnya, kita akan membandingkan *file image bmp 24-bit* yang belum disisipi pesan rahasia dengan *file image bmp 24-bit* yang sudah disisipi pesan rahasia, sedemikian sehingga kita dapat mengetahui proses atau cara kerja dari metode *Least Significant Bit* (*LSB*). Untuk mengetahui proses atau cara kerja dari metode *Least Significant Bit* (*LSB*), ada dua hal yang harus diperhatikan, yaitu:

1. Pesan rahasia yang telah kita tuliskan adalah kata **'CEPI'**. Berdasarkan tabel ASCII, masing-masing karakter yang mewakili kata **'CEPI'** tersebut**,** dapat dikonversi ke dalam bilangan-bilangan biner, sehingga diperoleh:

   Representasi biner huruf **C** adalah 1000011,

   Representasi biner huruf **E** adalah 1000101,

   Representasi biner huruf **P** adalah 1010000,

   Representasi biner huruf **I** adalah 1001001,
2. *File image bmp 24-bit* dengan warna hitam murni dalam format biner akan terlihat sebagai berikut:

   00000000 00000000 00000000

   Atau dalam format hexadesimal akan terlihat sebagai berikut:

   00 00 00

Setelah memperhatikan kedua hal di atas, sekarang kita bisa menganalisis contoh *file image bmp 24-bit* pada **Gambar 4.6** yang telah berisi pesan rahasia, di mana dalam hal ini pesan rahasia yang disisipkan adalah kata **'CEPI'**. Berdasarkaan metode *Least Significant Bit* (*LSB*), representasi bilangan biner dari

masing-masing karakter (huruf **C**, **E**, **P**, dan **I**) yang mewakili kata **'CEPI',** akan mengganti bit ke-8, 16, dan 24 pada representasi biner *file image bmp 24-bit* dari **Gambar 4.4.** Perhatikanlah proses atau cara kerja penyisipan representasi biner masing-masing karekter pada representasi biner *file image bmp 24-bit original* (**Gambar 4.4**) di bawah ini:

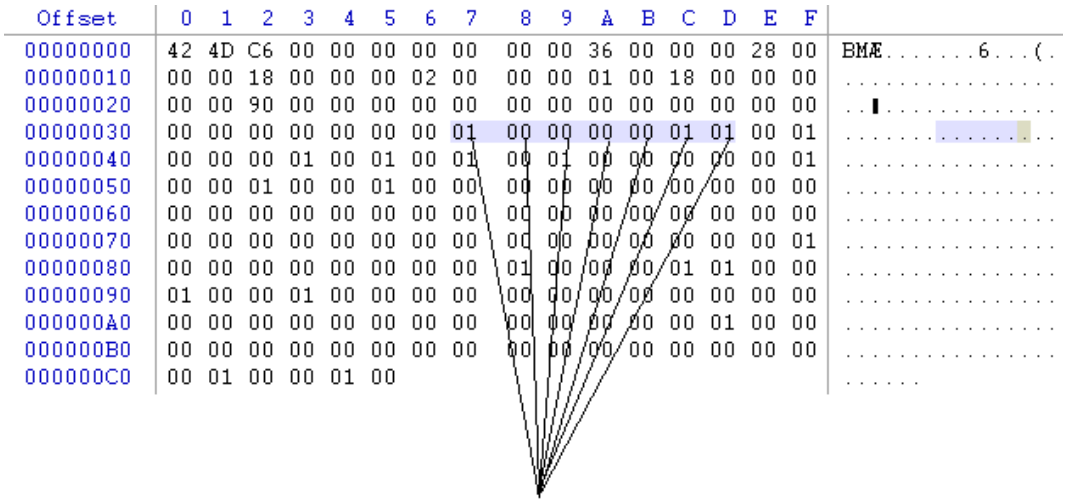
1. Berdasarkan metode *Least Significant Bit* (*LSB*), untuk karakter huruf **C** yang mempunyai representasi bilangan biner 1000011, setelah disisipkan pada representasi biner *file image bmp 24-bit original* (**Gambar 4.4**) akan menghasilkan:

   00000001 00000000 00000000 00000000 00000000 00000001 00000001

   Atau dalam format hexadesimal akan terlihat sebagai berikut:

   01 00 00 00 00 01 01

   Berdasarkan analisis di atas dapat dilihat bahwa setiap bit ke-8 diganti dengan representasi biner huruf **C** dan hanya tiga bit rendah yang berubah (berwarna biru). Perubahan data *original* setelah disisipi karakter huruf **C** yang mempunyai representasi bilangan biner 1000011, juga terlihat pada **Gambar 4.7**. Agar lebih jelas, perhatikan gambar di bawah ini:



Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
00000000	42	4D	C6	00	00	00	00	00	00	00	36	00	00	00	28	00	BMÆ.....6...(.
00000010	00	00	18	00	00	00	02	00	00	00	01	00	18	00	00	00	.....
00000020	00	00	90	00	00	00	00	00	00	00	00	00	00	00	00	00	..▌.....
00000030	00	00	00	00	00	00	00	01	00	00	00	00	01	01	00	01	.....
00000040	00	00	00	01	00	01	00	01	00	01	00	00	00	00	00	01	.....
00000050	00	00	01	00	00	01	00	00	00	00	00	00	00	00	00	00	.....
00000060	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
00000070	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	01	.....
00000080	00	00	00	00	00	00	00	00	01	00	00	00	01	01	00	00	.....
00000090	01	00	00	01	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000A0	00	00	00	00	00	00	00	00	00	00	00	00	00	01	00	00	.....
000000B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	.....
000000C0	00	01	00	00	01	00											.....

Representasi biner huruf **C** yang disisipkan pada data *original*

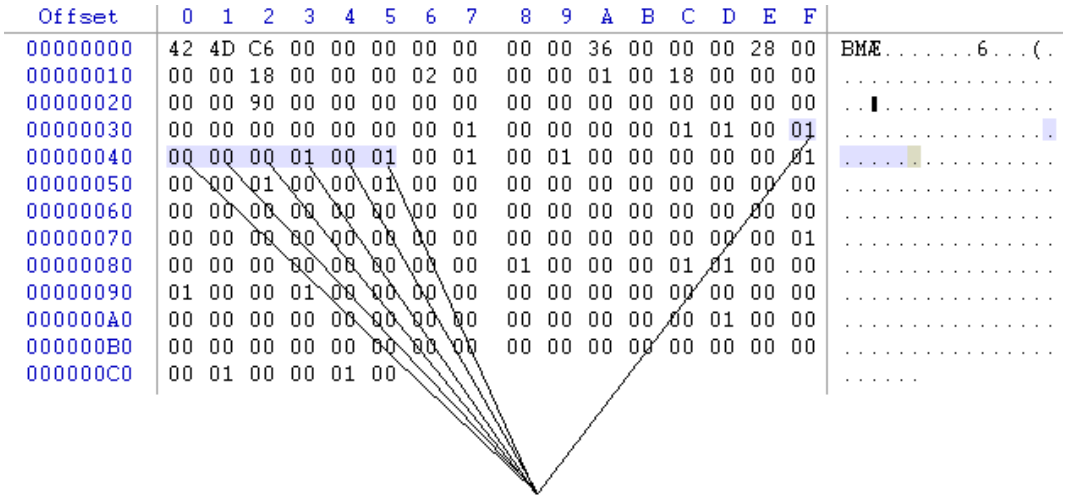
2. Untuk karakter huruf **E** yang mempunyai representasi bilangan biner 1000101, setelah disisipkan pada representasi biner *file image bmp 24-bit original* (**Gambar 4.4**) akan menghasilkan:

   00000001 00000000 00000000 00000000 00000001 00000000 00000001

   Atau dalam format hexadesimal akan terlihat sebagai berikut:

   01 00 00 00 01 00 01

   Berdasarkan analisis di atas dapat dilihat bahwa setiap bit ke-8 diganti dengan representasi biner huruf **E** dan hanya tiga bit rendah yang berubah (berwarna biru). Perubahan data *original* setelah disisipi karakter huruf **E** yang mempunyai representasi bilangan biner 1000101, juga terlihat pada **Gambar 4.7**. Agar lebih jelas, perhatikan gambar di bawah ini:



Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00000000	42	4D	C6	00	00	00	00	00	00	00	36	00	00	00	28	00
00000010	00	00	18	00	00	00	02	00	00	00	01	00	18	00	00	00
00000020	00	00	90	00	00	00	00	00	00	00	00	00	00	00	00	00
00000030	00	00	00	00	00	00	00	01	00	00	00	00	01	01	00	01
00000040	00	00	00	01	00	01	00	01	00	01	00	00	00	00	00	01
00000050	00	00	01	00	00	01	00	00	00	00	00	00	00	00	00	00
00000060	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000070	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	01
00000080	00	00	00	00	00	00	00	00	01	00	00	00	01	01	00	00
00000090	01	00	00	01	00	00	00	00	00	00	00	00	00	00	00	00
000000A0	00	00	00	00	00	00	00	00	00	00	00	00	00	01	00	00
000000B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000C0	00	01	00	00	01	00										

Representasi biner huruf **E** yang disisipkan pada data *original*

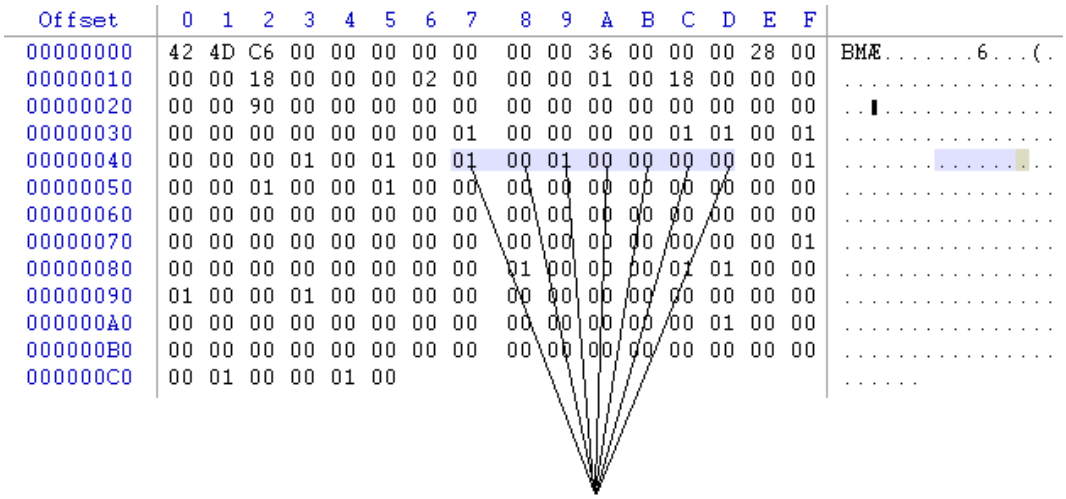
3. Untuk karakter huruf **P** yang mempunyai representasi bilangan biner 1010000, setelah disisipkan pada representasi biner *file image bmp 24-bit original* (**Gambar 4.4**) akan menghasilkan:

   00000001 00000000 00000001 00000000 00000000 00000000 00000000

   atau dalam format hexadesimal akan terlihat sebagai berikut:

   01 00 01 00 00 00 00

   Berdasarkan analisis di atas dapat dilihat bahwa setiap bit ke-8 diganti dengan representasi biner huruf **P** dan hanya dua bit rendah yang berubah (berwarna biru). Perubahan data *original* setelah disisipi karakter huruf **P** yang mempunyai representasi bilangan biner 1010000, juga terlihat pada **Gambar 4.7**. Agar lebih jelas, perhatikan gambar di bawah ini:



Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00000000	42	4D	C6	00	00	00	00	00	00	00	36	00	00	00	28	00
00000010	00	00	18	00	00	00	02	00	00	00	01	00	18	00	00	00
00000020	00	00	90	00	00	00	00	00	00	00	00	00	00	00	00	00
00000030	00	00	00	00	00	00	00	01	00	00	00	00	01	01	00	01
00000040	00	00	00	01	00	01	00	01	00	01	00	00	00	00	00	01
00000050	00	00	01	00	00	01	00	00	00	00	00	00	00	00	00	00
00000060	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000070	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	01
00000080	00	00	00	00	00	00	00	00	01	00	00	00	01	01	00	00
00000090	01	00	00	01	00	00	00	00	00	00	00	00	00	00	00	00
000000A0	00	00	00	00	00	00	00	00	00	00	00	00	00	01	00	00
000000B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000C0	00	01	00	00	01	00										

Representasi biner huruf **P** yang disisipkan pada data *original*

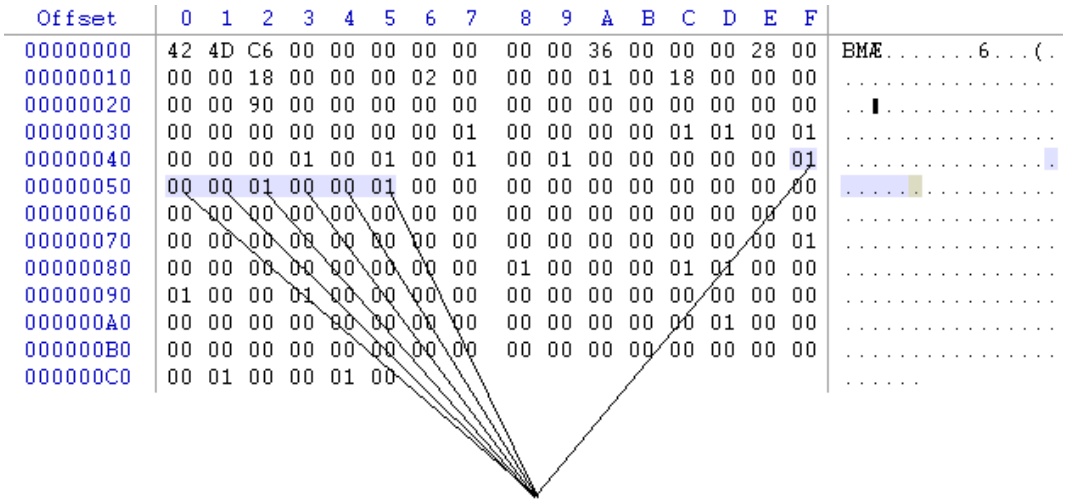
4. Untuk karakter huruf **I** yang mempunyai representasi bilangan biner 1001001, setelah disisipkan pada representasi biner *file image bmp 24-bit original* (**Gambar 4.4**) akan menghasilkan:

   00000001 00000000 00000000 00000001 00000000 00000000 00000001

   Atau dalam format hexadesimal akan terlihat sebagai berikut:

   01 00 00 01 00 00 01

   Berdasarkan analisis di atas dapat dilihat bahwa setiap bit ke-8 diganti dengan representasi biner huruf **I** dan hanya tiga bit rendah yang berubah (berwarna biru). Perubahan data *original* setelah disisipi karakter huruf **I** yang mempunyai representasi bilangan biner 1001001, juga terlihat pada **Gambar 4.7**. Agar lebih jelas, perhatikan gambar di bawah ini:



Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00000000	42	4D	C6	00	00	00	00	00	00	00	36	00	00	00	28	00
00000010	00	00	18	00	00	00	02	00	00	00	01	00	18	00	00	00
00000020	00	00	90	00	00	00	00	00	00	00	00	00	00	00	00	00
00000030	00	00	00	00	00	00	00	01	00	00	00	00	01	01	00	01
00000040	00	00	00	01	00	01	00	01	00	01	00	00	00	00	00	01
00000050	00	00	01	00	00	01	00	00	00	00	00	00	00	00	00	00
00000060	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000070	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	01
00000080	00	00	00	00	00	00	00	00	01	00	00	00	01	01	00	00
00000090	01	00	00	01	00	00	00	00	00	00	00	00	00	00	00	00
000000A0	00	00	00	00	00	00	00	00	00	00	00	00	00	01	00	00
000000B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000C0	00	01	00	00	01	00										

Representasi biner huruf **I** yang disisipkan pada data *original*

Sekarang perhatikan kembali **Gambar 4.5** dan **Gambar 4.7**. Dengan membandingkan kedua gambar ini, secara keseluruhan masing-masing karakter yang mewakili kata **'CEPI'** yang merupakan pesan rahasia yang disisipkan pada data *original* dapat terlihat dengan jelas seperti tampak di bawah ini.

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00000000	42	4D	C6	00	00	00	00	00	00	00	36	00	00	00	28	00
00000010	00	00	18	00	00	00	02	00	00	00	01	00	18	00	00	00
00000020	00	00	90	00	00	00	00	00	00	00	00	00	00	00	00	00
00000030	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000040	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000050	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000060	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000070	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000080	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000090	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000C0	00	00	00	00	00	00										

Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00000000	42	4D	C6	00	00	00	00	00	00	00	36	00	00	00	28	00
00000010	00	00	18	00	00	00	02	00	00	00	01	00	18	00	00	00
00000020	00	00	90	00	00	00	00	00	00	00	00	00	00	00	00	00
00000030	00	00	00	00	00	00	00	01	00	00	00	00	01	01	00	01
00000040	00	00	00	01	00	01	00	01	00	01	00	00	00	00	00	01
00000050	00	00	01	00	00	01	00	00	00	00	00	00	00	00	00	00
00000060	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00000070	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	01
00000080	00	00	00	00	00	00	00	00	01	00	00	00	01	01	00	00
00000090	01	00	00	01	00	00	00	00	00	00	00	00	00	00	00	00
000000A0	00	00	00	00	00	00	00	00	00	00	00	00	00	01	00	00
000000B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000C0	00	01	00	00	01	00										

Dari hasil-hasil analisis di atas, kita telah membuktikan kebenaran dari metode *Least Significant Bit* (*LSB*), yaitu suatu metode penyembunyian pesan rahasia melalui media *digital file image*, khususnya media *digital file image bmp 24-bit* dengan cara mengganti bit ke-8, 16, dan 24 pada representasi biner *file image bmp 24-bit* dengan representasi biner pesan rahasia yang akan disembunyikan.