BAB IV

PENGUJIAN, ANALISIS DAN PEMBAHASAN SISTEM

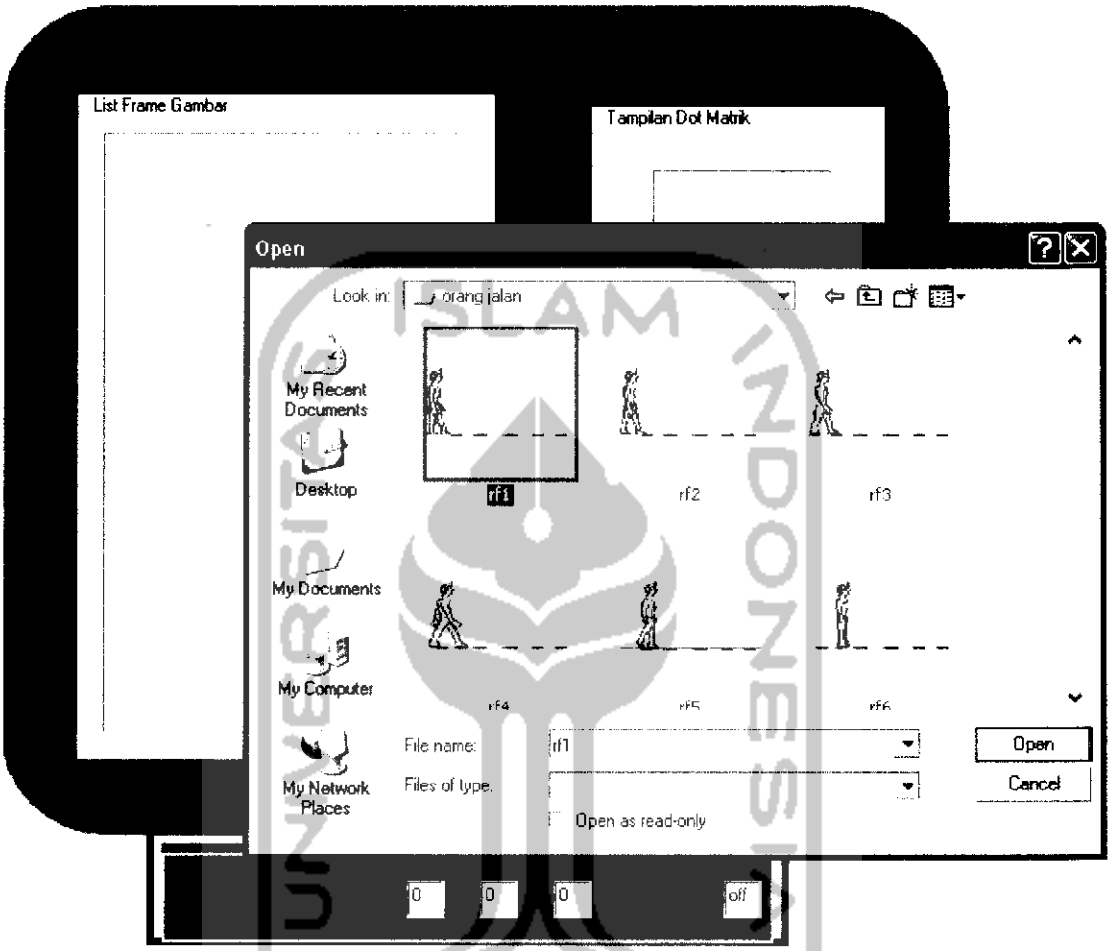
Perangkat lunak (*software*) untuk menampilkan animasi pada *dot matrix display* ini terdiri atas beberapa tombol kontrol utama yang dipasang pada antarmuka dalam menjalankan animasi yang akan ditampilkan pada perangkat keras *dot matrix display* diantaranya yaitu tombol *command1* ("Kirim Gambar"), tombol *command2* ("Mainkan" dan "Berhenti"), tombol *command3* ("Hapus Gambar"), dan tombol *command4* ("Tutup").

4.1 Pengujian *User Interface*/Antarmuka Pada Perangkat Lunak (*Software*)

Sebelum menjalankan antarmuka program aplikasi yang telah dirancang, pastikan terlebih dahulu *port* komunikasi antara *MSComm* dengan COM *number* yang ada di PC (*Personal Computer*) pengaturannya sudah sama.

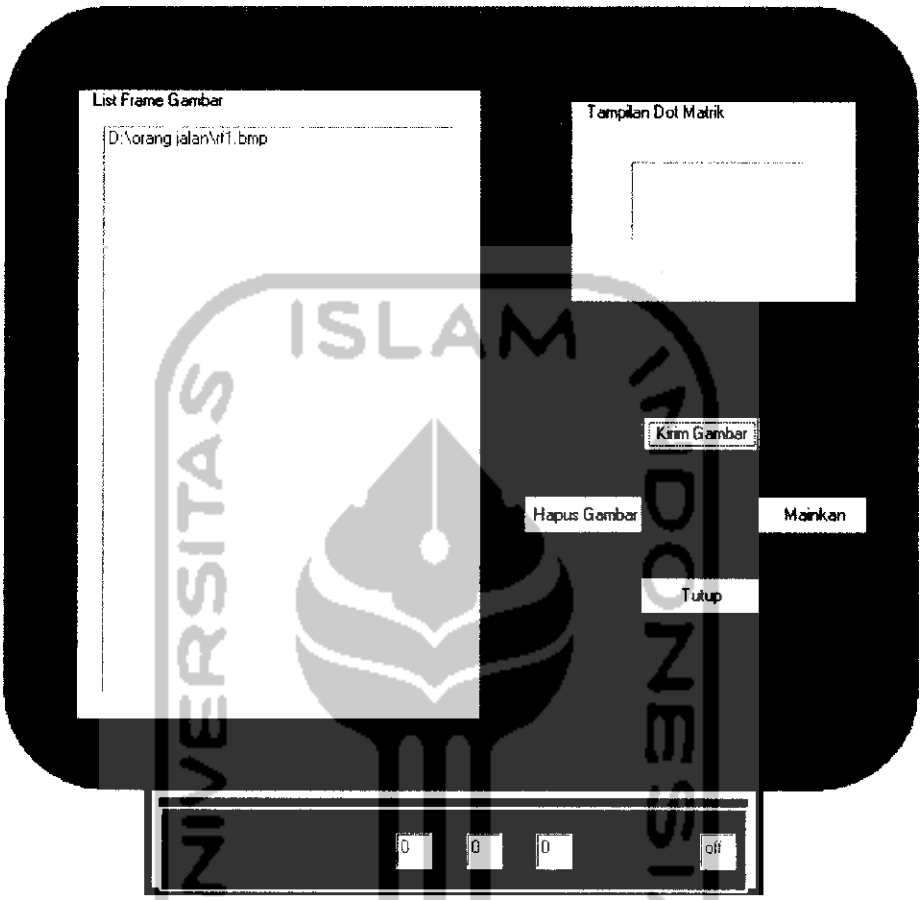
4.1.1 Pengujian Tombol *Command1* ("Kirim Gambar")

Tombol perintah "Kirim Gambar" pada saat setelah melakukan pengeklikkan, melakukan pemanggilan terhadap kotak dialog "*Open*" untuk mengirimkan data berformat *.bmp* atau berupa *frame* gambar dengan jenis warna yang terdefenisikan hitam dan putih untuk dikumpulkan pada kotak "*List Frame* Gambar" sesuai dengan alur animasi gambar yang dirancang. Pengiriman *frame* gambar dilakukan secara satu per satu setiap dilakukan pemanggilan terhadap kotak dialog "*open*".



Gambar 4.1 Tampilan antarmuka saat tombol "Kirim Gambar" diklik

Setiap *frame* gambar yang dikirim ke kotak "*List Frame Gambar*" akan diberi nama sesuai dengan *source path* (saluran sember data) lokasi tempat *frame* gambar disimpan.

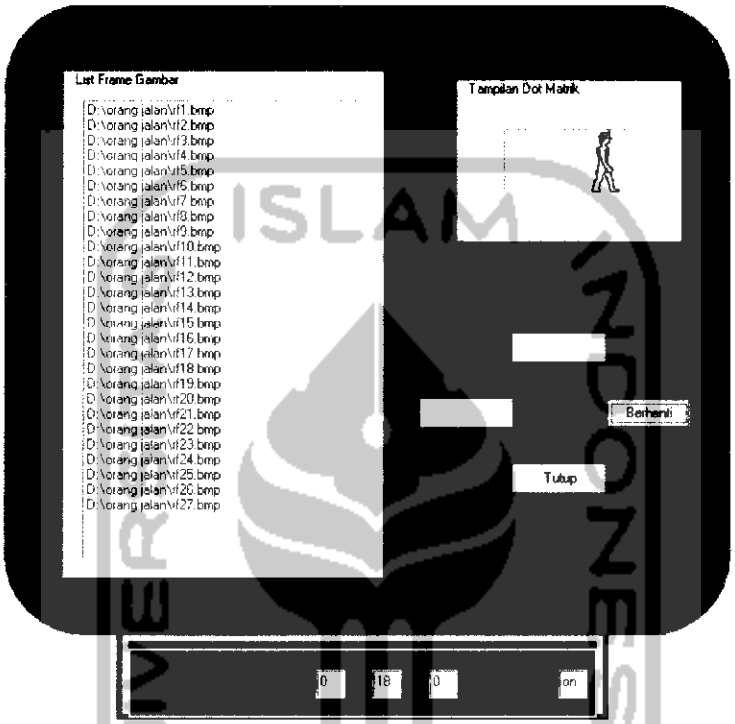


Gambar 4.2 Tampilan antarmuka ketika *frame* gambar telah dikirim ke *listbox*

4.1.2 Pengujian Tombol *Command2* ("Mainkan" dan "Berhenti")

Tombol perintah "Mainkan" akan menjalankan animasi gambar yang dibangun sesuai dengan susunan rancangan *frame* gambar yang telah dibuat sebelumnya. Proses animasi ini terjadi karena *frame-frame* gambar yang ada di *listbox* "*List Frame* gambar" ditampilkan satu persatu pada *picturebox* ("Tampilan Dot Matrik") sesuai *interval* waktu yang telah diatur pada pengaturan kecepatan

pengiriman data (*baudrate*) yaitu sebesar 9600 *bits per second* yang ada di properti *MSComm* dan COM *number*.

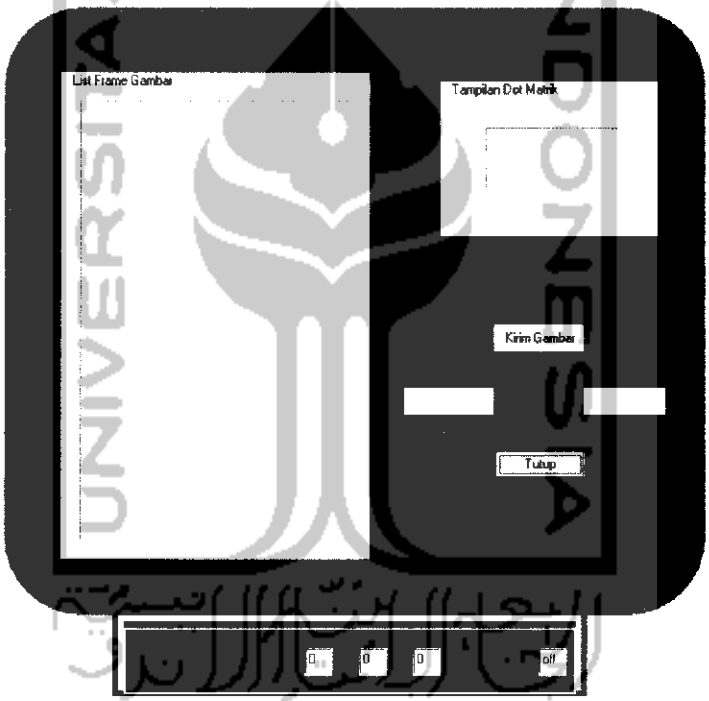


Gambar 4.3 Tampilan antarmuka saat proses penganimasian gambar sedang berlangsung

Proses penganimasian gambar ini akan berlangsung terus menerus secara berulang-ulang sampai tombol perintah "Berhenti" ditekan. Saat tombol perintah "Berhenti" ditekan kemudian tombol perintah "Mainkan" ditekan kembali maka penganimasian gambar akan dimulai dari *frame* gambar terakhir saat proses penganimasian gambar dihentikan.

4.1.3 Pengujian Tombol *Command3* ("Hapus Gambar")

Tombol ini akan aktif setelah adanya pengeklikkan pada tombol "Berhenti" saat proses penganimasian gambar sedang berlangsung, apabila tombol "Hapus Gambar" ini diklik maka *frame* gambar yang ada di *listbox* akan dihapus seluruhnya secara sekaligus. Hal ini disebabkan karena tombol ini melakukan aksi menjadikan *listbox* bernilai 0.



Gambar 4.4 Tampilan antarmuka setelah dilakukan pengeklikkan pada tombol "Hapus Gambar"

Tampilan ini sama seperti tampilan awal antarmuka saat program pertama kali dijalankan, yaitu saat belum adanya *frame* gambar yang dikirimkan ke *listbox*.

4.1.4 Pengujian Tombol *Command4* ("Tutup")

Tombol ini menutup program aplikasi yang sedang dijalankan sesaat setelah penganimasian gambar telah dihentikan. Apabila tombol ini diklik sewaktu proses penganimasian gambar sedang berlangsung, maka tombol ini akan berubah fungsi menjadi fungsi tombol "Berhenti" untuk menghentikan proses penganimasian gambar terlebih dahulu dan setelah itu barulah program akan ditutup setelah dilakukan pengeklikkan kedua.

4.2 Pengujian Perangkat Lunak Terhadap Perangkat Keras *Dot Matrix Display*

4.2.1 Pengujian Nilai *Baudrate*

Program aplikasi dari perangkat lunak yang dijalankan dihubungkan dengan menggunakan kabel serial sebagai media pengiriman data ke perangkat keras *dot matrix display*. Karena banyak data yang dikirimkan untuk sekali pengiriman ke perangkat keras *dot matrix display* adalah per 8 bit (1 *byte*), maka untuk setiap pengiriman *frame* gambar membutuhkan waktu sesuai dengan perhitungan di bawah ini:

Baudrate	=	9600 bps
Banyak data yang dikirimkan	=	8 bit (1 *byte*)
Ukuran dimensi *dot matrix display*	=	8960 *pixel* (1 *pixel* mewakili 1 bit data)

Maka, waktu yang dibutuhkan untuk sekali menampilkan satu *frame* gambar ke *dot matrix display* adalah:

$$\frac{8960}{9600} \quad x\ 1 \text{ detik} \quad = \quad 0{,}933 \text{ detik}$$

Dapat diketahui bahwa semakin besar nilai *baudrate* yang digunakan semakin cepat waktu yang dibutuhkan untuk sekali pengiriman *frame* gambar ke *dot matrix display*. Karena pada perangkat keras *dot matrix display* ini menggunakan *oscillator* sebesar 11.0592 MHz, maka ada beberapa pilihan penggunaan *baudrate* yang dapat dikonfigurasi pada program assembly didalam mikrokontrolernya.

Tabel 4.1 Konfigurasi *baudrate* dalam bahasa *assembly*

Baudrate (bps)	Frekuensi *Oscillator* (MHz)					SMOD
	11.0592	12	14.7456	16	20	
150	40h	30h	00h			0
300	A0h	98h	80h	75h	52h	0
600	D0h	CCh	C0h	BBh	A9h	0
1200	E8h	E6h	E0h	DEh	D5h	0
2400	F4h	F3h	F0h	EFh	EAh	0
4800		F3h	EFh	EFh		1
4800	FAh		F8h		F5h	0
9600	FDh		FCh			0
9600					F5h	1
19200	FDh		FCh			1
38400			FEh			
76800			FFh			

SMOD merupakan bit yang ada pada *register* PCON yang mana bila nilainya di*set* 1, maka kecepatan data (*baudrate*) akan dikalikan dua. Setelah dilakukan uji coba pengubahan nilai *baudrate* dari 9600 bps ke 19200 bps pada mikrokontrolernya yang diikuti dengan menyesuaikan nilai *baudrate* MSComm dan

COM pada PC maka dapat dianalisa bahwa perubahan yang terjadi pada nyala LED *dot matrix* dapat dilihat pada tabel berikut:

Tabel 4.2 Uji coba pengaruh peningkatan nilai *baudrate* terhadap LED *dot matrix*

Baudrate (bps)	Kondisi LED *Dot Matrix*
9600	Nyala terang
19200	*error*

Baudrate dipengaruhi oleh besarnya nilai TH1 (*timer high*1). *Timer* yang dipakai pada program *assembly* pengendalian nyala LED *dot matrix* ini adalah jenis *timer1 mode2* yang dipengaruhi oleh SMOD yang merupakan bit yang ada pada *register* PCON. Apabila SMOD di*set* aktif (SMOD = 1) pada *register* ini maka nilai *baudrate* digandakan menjadi 19200 bps persamaan yang dipakai untuk menghitung nilai TH1 adalah:

TH1 = 256 – (frekuensi osilator / (12*16)) / *baudrate*

(12*32) di sini adalah untuk 12 merupakan pembagi frekuensi waktu sistem (*system clock frequency*) sedangkan 32 merupakan frekuensi yang melebihi batas *baudrate* yang digunakan dari pewaktu yang dijalankan (*timer overflow frequency*). Dengan demikian dari persamaan di atas didapat nilai TH1 adalah 253 (0FDh). Sedangkan SMOD di*set* nonaktif (SMOD = 0) maka persamaan yang digunakan untuk menghitung nilai TH1 adalah

TH1 = 256 – (frekuensi osilator / (12*32)) / *baudrate*

Karena batasan masalah yang sudah ditentukan dalam penelitian tugas akhir ini yaitu *baudrate* yang diisikan ke mikrokontroler lewat program menggunakan bahasa *assembly* adalah 9600 *bits per second* maka waktu yang diperlukan untuk satu kali menampilkan *frame* gambar adalah 0,9 detik.

Waktu yang dihasilkan oleh *baudrate* 9600 ini dinilai kurang cepat untuk melakukan proses penganimasian gambar. maka dari itu perlu dilakukan sedikit uji coba pengubahan *delay* (tunda) waktu yang terdapat pada program *assembly* di dalam mikrokontrolernya.

Waktu tampil tiap 1 *frame* gambar = 1 detik / (TH1 * waktu tunda)

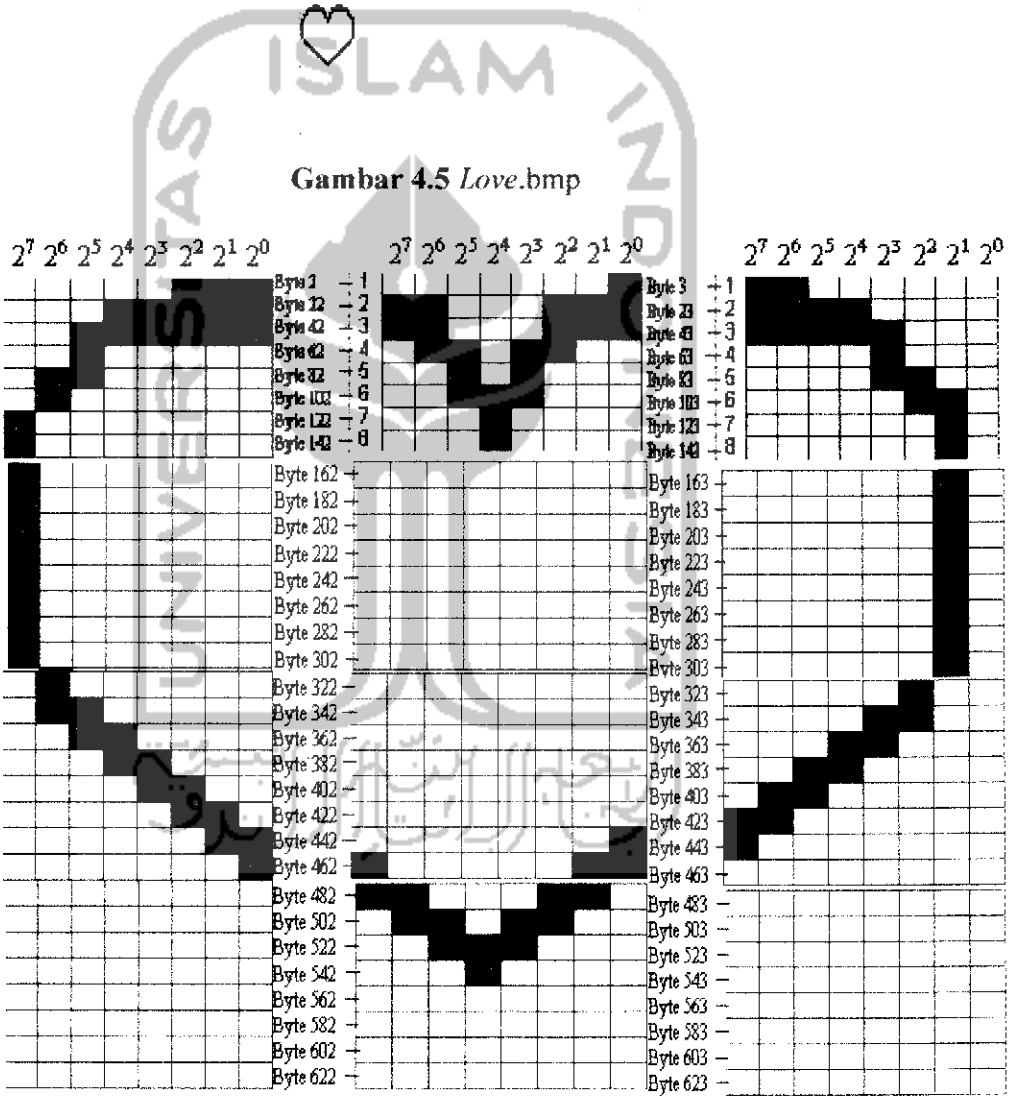
Tabel 4.3 Uji coba pengaruh pengubahan nilai waktu tunda

Waktu Tunda	Waktu tampil tiap 1 *frame* gambar	Satuan
-1000	0,004	mS
-2000	0,002	mS
-6000	0,6	uS

4.2.2 Pengujian Pengiriman *Frame* Gambar

Frame gambar yang dikirimkan dari antarmuka perangkat lunak dipilah menjadi 8 bit penyalaan LED *dot matrix* (1 bit mewakili nyala 1 LED) yang dimulai dari *byte* 1, *byte* 2, *byte* 3 dan seterusnya sesuai dengan urutan alur *scanning* penyalaan LED dari kiri paling atas (*byte*1) hingga kanan paling bawah (*byte* 1120). Karena pada perangkat keras *display dot matrix* ini hanya terpasang 15 keping *dot matrix* untuk panjang *display* yang seharusnya 20 keping *dot matrix* (sesuai dengan program assembly yang diisikan pada setiap mikrokontrolernya)

maka, untuk penyalaan LED dari *byte* 16 sampai *byte* 20 hingga seterusnya sejajar sampai kebawah tidak dapat dilakukan. Akan tetapi, alur penyalaan LED *dot matrix* tetap diproses dari *byte 1* sampai *byte* 20 dan seterusnya. Berikut adalah contoh proses pengiriman sebuah *frame* gambar yang dinyalakan pada *dot matrix display*:

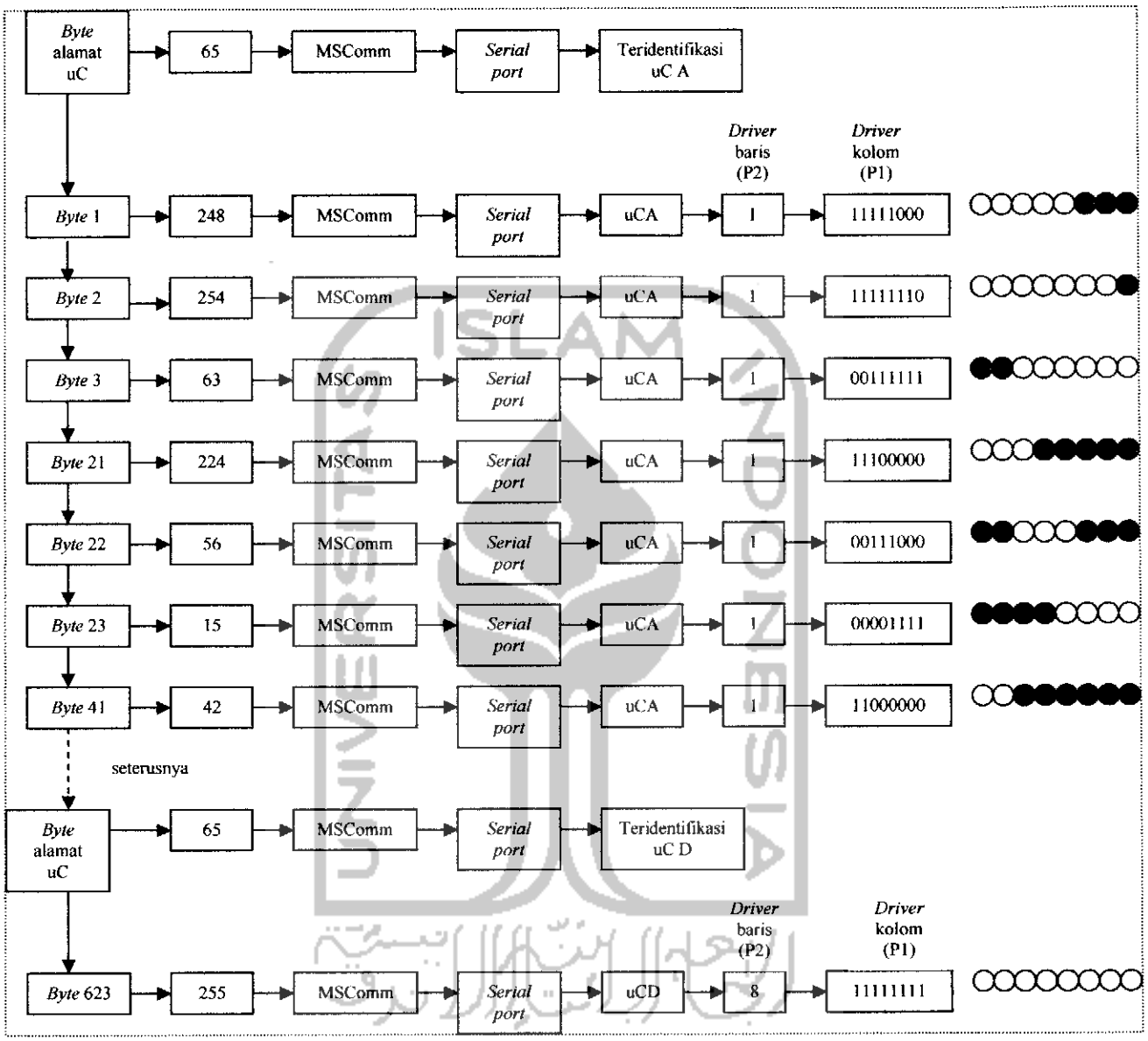


Gambar 4.5 *Love*.bmp

Gambar 4.6 Alur proses *scanning* penyalaan LED *dot matrix*

Tabel 4.4 Proses pengolahan data dari bilangan biner ke bentuk bilangan ASCII

Byte	Biner	ASCII	Byte	Biner	ASCII	Byte	Biner	ASCII
uC A								
1	11111000	248	2	11111110	254	3	00111111	63
21	11100000	224	22	00111000	56	23	00001111	15
41	11000000	192	42	00111000	56	43	00000111	7
61	11011111	223	62	10010011	147	63	11110111	247
81	10011111	159	82	11010111	215	83	11110011	243
101	10111111	191	102	11000111	199	103	11111001	249
121	01111111	127	122	11101111	239	123	11111101	253
141	01111111	127	142	11101111	239	143	11111101	253
uC B								
161	01111111	127	162	11111111	255	163	11111101	253
181	01111111	127	182	11111111	255	183	11111101	253
201	01111111	127	202	11111111	255	203	11111101	253
221	01111111	127	222	11111111	255	223	11111101	253
241	01111111	127	242	11111111	255	243	11111101	253
261	01111111	127	262	11111111	255	263	11111101	253
281	01111111	127	282	11111111	255	283	11111101	253
301	01111111	127	302	11111111	255	303	11111101	253
uC C								
321	10111111	191	322	11111111	255	323	11111011	251
341	10011111	159	342	11111111	255	343	11110011	243
361	11001111	207	362	11111111	255	363	11100111	231
381	11100111	231	382	11111111	255	383	11001111	207
401	11110011	243	402	11111111	255	403	10011111	159
421	11111001	249	422	11111111	255	423	00111111	63
441	11111100	252	442	11111110	254	443	01111111	127
461	11111110	254	462	01111100	124	463	11111111	255
uC D								
481	11111111	255	482	00111001	57	483	11111111	255
501	11111111	255	502	10010011	147	503	11111111	255
521	11111111	255	522	11000111	199	523	11111111	255
541	11111111	255	542	11101111	239	543	11111111	255
561	11111111	255	562	11111111	255	563	11111111	255
581	11111111	255	582	11111111	255	583	11111111	255
601	11111111	255	602	11111111	255	603	11111111	255
621	11111111	255	622	11111111	255	623	11111111	255



Gambar 4.7 Blok diagram alur proses penyalaan LED *dot matrix*

Kode-kode ASCII diatas akan dikirimkan lewat MSComm, karena pengolahan data yang dilakukan di perangkat lunak hanya mengerti tipe data yang

berformat ASCII. *Serial port* yang mengkonversi kode-kode ASCII tersebut kedalam bentuk bilangan biner yang kemudian akan dibaca oleh mikrokontroler.

Alur proses penyalaan LED *dot matrix* pada blok diagram menggambarkan proses pengiriman data ASCII dari antarmuka *visual basic* sampai pada penyalaan LED *dot matrix* diperangkat kerasnya. Untuk gambar "*love*" pengiriman data ASCII digambarkan hanya sampai pada alamat mikrokontroler "D", sedangkan untuk mikrokontroler "E" sampai "G" kondisinya adalah sama yaitu berlogika 1.

Gambar 4.8 Tampilan gambar "*love*" pada *dot matrix display*

Perancangan gambar animasi yang sudah diujikan di sini adalah gambar animasi orang yang sedang berjalan. Gambar animasi ini dibentuk oleh 27 *frame* gambar yang menyusun pergerakan orang yang tampak sedang berjalan. Pengujian gambar animasi ini dilakukan sebanyak tiga kali yaitu berdasarkan pengubahan nilai waktu tunda pada program *assembly* yang diisikan ke dalam mikrokontrolernya.

Gambar 4.9 Tampilan gambar animasi orang sedang berjalan