

## **BAB IV**

### **HASIL PENGUJIAN DAN PENGAMATAN**

Dalam bab ini penulis akan menguraikan dan menjelaskan hasil analisa pengujian dari hasil penelitian tugas akhir ini yang telah dilakukan, pengujian dilakukan dalam beberapa bagian yang disusun dalam urutan dari yang sederhana menuju sistem yang lengkap. Pengujian dilakukan meliputi pengujian perangkat keras (*hardware*) dan perangkat lunak (*software*) diharapkan didapat suatu sistem yang dapat menjalankan rancangan alat berjalan dengan baik dan optimal.

#### **4.1 Pengujian Mikrokontroler ATmega32**

##### **4.1.1 Tujuan Pengujian**

Pengujian Minimum Sistem bertujuan mengetahui apakah Mikrokontroler dapat melakukan proses *download* program sehingga dapat dinyatakan bahwa Mikrokontroler dapat digunakan dan berjalan dengan baik.

##### **4.1.2 Alat Yang Dibutuhkan**

1. Rangkaian Minimum Sistem Mikrokontroler ATmega32.
2. Komputer.
3. *Downloader* ATMEL USB ASP.
4. Adaptor 12 Volt.

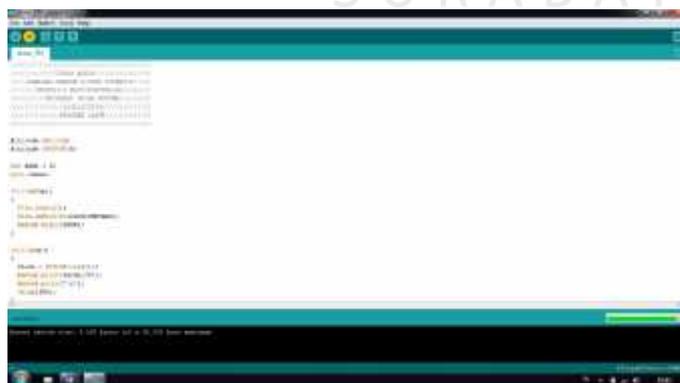
##### **4.1.3 Prosedur Pengujian**

1. Letakkan file boards.txt pada C:\Program Files\Arduino\hardware\arduino.
2. Letakkan file pins\_arduino.h pada folder C:\Program Files\Arduino\hardware\arduino\variants\mega32.

3. Hubungkan baterai pada input tegangan Minimum Sistem dan nyalakan switch on pada minimum sistem.
4. Selanjutnya aktifkan komputer dan jalankan program Arduino IDE.
5. Sambungkan *downloader* ATMEL USB ASP dengan komputer.
6. Sambungkan Minimum Sistem dengan kabel *downloader*.
7. Buka *sketch* yang akan diupload.
8. *Setting board, serial port* dan *programmer* sesuai dengan yang digunakan.
9. Kemudian *upload sketch* dan tunggu hingga selesai.
10. Setelah *upload* selesai akan diketahui program berhasil di *download* apa tidak.

#### 4.1.4 Hasil Pengujian

Dari percobaan di atas apabila terjadi proses *upload* program seperti Gambar 4.1 dan tidak ada *comment* yang menunjukkan kegagalan dalam sambungan antara *downloader* dan minimum sistem maka proses *upload* program akan berjalan dengan baik yang di tandai dengan tampil *comment* seperti yang di tunjukan pada Gambar 4.2.



**Gambar 4.1** Tampilan Proses *Upload* dari Arduino IDE



**Gambar 4.2** Tampilan *Comment* saat Program Berhasil di *Upload*

## 4.2 Pengujian Arduino Uno

### 4.2.1 Tujuan Pengujian

Pengujian Arduino Uno bertujuan mengetahui apakah Arduino Uno dapat melakukan proses *upload* program sehingga dapat dinyatakan bahwa Arduino Uno dapat digunakan dan berjalan dengan baik.

### 4.2.2 Alat Yang Dibutuhkan

1. Rangkaian Arduino Uno.
2. Rangkaian Power.
3. Adaptor 12 Volt.
4. Komputer

### 4.2.3 Prosedur Pengujian

1. Hubungkan adaptor 12 Volt dengan rangkaian power.
2. Hubungkan Arduino Uno dengan rangkain power.
3. Hubungkan Arduino Uno dengan komputer menggunakan komunikasi serial.
4. Buka aplikasi arduino.
5. Buka *sketch* yang akan di *upload*.
6. Tekan menu *upload* pada aplikasi arduino dan tunggu hingga proses *upload* selesai.

#### 4.2.4 Hasil Pengujian

Dari percobaan di atas apabila terjadi proses *upload* program seperti Gambar 4.3 dan tidak ada *comment* yang menunjukkan kegagalan dalam sambungan antara komputer dan Arduino Uno maka proses *upload* program akan berjalan dengan baik yang di tandai dengan tampil *comment* seperti yang di tunjukan pada Gambar 4.4.



Gambar 4.3 Tampilan Proses Download Arduino



Gambar 4.4 Tampilan *Comment* saat Arduino Berhasil di *Download*

### 4.3 Pengujian LCD (*Liquid Cristal Display*)

#### 4.3.1 Tujuan Pengujian

Pengujian LCD (*Liquid Cristal Display*) bertujuan untuk mengetahui apakah LCD (*Liquid Cristal Display*) dapat terkoneksi dengan Mikrokontroler

dan dapat berjalan dengan baik sesuai dengan tampilan yang diharapkan program yang telah dibuat dan dapat digunakan.

#### 4.3.2 Alat Yang Dibutuhkan

1. Rangkaian Arduino Uno.
2. LCD (*Liquid Cristal Display*).
3. Komputer.
4. Rangkaian I2C.

#### 4.3.3 Prosedur Pengujian

1. Hubungkan rangkaian Arduino Uno dengan komputer.
2. Sambungkan LCD (*Liquid Cristal Display*) dengan rangkaian I2C.
3. Sambungkan rangkaian I2C dengan Arduino Uno.
4. Pastikan *sketch* telah di *upload*.

#### 4.3.4 Hasil Pengujian

Dari percobaan di atas apabila LCD (*Liquid Cristal Display*) menunjukkan tampilan yang sesuai dengan *sketch* yang telah dibuat dan di *upload* sebelumnya pada Arduino Uno seperti pada Gambar 4.5, maka dapat dikatakan LCD (*Liquid Cristal Display*) dapat berfungsi dengan baik dan dapat digunakan dalam penelitian ini.



**Gambar 4.5** Tampilan LCD (*Liquid Cristal Display*)

## **4.4 Pengujian RFID**

### **4.4.1 Tujuan Pengujian**

Tujuan pengujian RFID ini adalah untuk memperoleh sampel hasil pembacaan RFID dan memastikan RFID dapat berjalan dengan baik untuk melakukan proses scan *tag* RFID.

### **4.4.2 Alat Yang Dibutuhkan**

1. Komputer.
2. Rangkaian Arduino Uno.
3. RFID.
4. *Tag* RFID.

### **4.4.3 Prosedur Pengujian**

1. Hubungkan rangkaian Arduino Uno dengan RFID.
2. Hubungkan Arduino Uno dengan komputer menggunakan komunikasi serial.
3. *Upload sketch* uji RFID.
4. Uji RFID dengan menyentuhkan *tag* RFID pada RFID.
5. Jika sudah terlihat isi dari *tag* RFID, maka dapat disimpulkan bahwa RFID dapat berjalan dengan baik.

### **4.4.4 Hasil Pengujian**

Dari pengujian yang telah dilakukan berdasarkan prosedur pengujian RFID, maka diperoleh sampel data hasil pembacaan RFID dari beberapa sumber yang ditunjukkan pada Tabel 4.1. Dapat dipastikan pula RFID dapat berfungsi dengan baik untuk pembacaan *tag* RFID dan dapat digunakan pada penelitian ini.

Tabel 4.1 Hasil Sampel ID Pembacaan RFID

NO	SUMBER	PEMBACAAN RFID		
		HASIL 1	HASIL 2	HASIL 3
1.	TAG	140 129 196 101	140 129 196 101	140 129 196 101
2.	TAG	196 245 56 165	196 245 56 165	196 245 56 165
3.	KTM	4 86 9 198	4 86 9 198	4 86 9 198
4.	KTM	16 95 132 42	16 95 132 42	16 95 132 42
5.	KTM	109 128 31 203	109 128 31 203	109 128 31 203
6.	KTM	180 233 6 198	180 233 6 198	180 233 6 198
7.	KTM	84 168 7 198	84 168 7 198	84 168 7 198
8.	KTM	112 135 114 42	112 135 114 42	112 135 114 42
9.	KTM	48 242 136 42	48 242 136 42	48 242 136 42
10.	KTM	164 255 11 198	164 255 11 198	164 255 11 198
11.	KTM	4 142 8 198	4 142 8 198	4 142 8 198
12.	KTM	192 82 132 42	192 82 132 42	192 82 132 42
13.	KTM	228 108 10 198	228 108 10 198	228 108 10 198
14.	KTM	48 170 138 42	48 170 138 42	48 170 138 42
15.	KTM	16 79 140 42	16 79 140 42	16 79 140 42
16.	KTM	228 223 4 198	228 223 4 198	228 223 4 198
17.	KTM	20 104 13 198	20 104 13 198	20 104 13 198
18.	KTM	84 120 11 198	84 120 11 198	84 120 11 198
19.	KTM	68 110 16 198	68 110 16 198	68 110 16 198
20.	KTM	228 14 8 198	228 14 8 198	228 14 8 198
21.	KTM	164 59 14 198	164 59 14 198	164 59 14 198
22.	KTM	52 234 6 198	52 234 6 198	52 234 6 198
23.	KTM	212 93 13 198	212 93 13 198	212 93 13 198
24.	KTM	224 186 142 42	224 186 142 42	224 186 142 42
25.	KTM	36 127 9 198	36 127 9 198	36 127 9 198
26.	KTM	36 104 10 198	36 104 10 198	36 104 10 198
27.	KTM	240 101 142 42	240 101 142 42	240 101 142 42
28.	KTM	132 83 6 198	132 83 6 198	132 83 6 198
29.	KTM	160 131 106 42	160 131 106 42	160 131 106 42
30.	KTM	192 254 108 42	192 254 108 42	192 254 108 42

Kesimpulan dari Tabel 4.1 dengan melakukan *scan* RFID sebanyak 3 kali adalah mendapatkan hasil ID yang berbeda untuk setiap kartu sehingga kartu tidak akan tertukar satu dengan yang lainnya.

## 4.5 Pengujian Komunikasi I2C dan Penyimpanan EEPROM

### 4.5.1 Tujuan Pengujian

Pengujian Komunikasi I2C dan penyimpanan pada EEPROM ini bertujuan untuk mengetahui apakah komunikasi antara *master* dan *slave* dapat berjalan dengan baik. Pengujian ini dilakukan dengan bantuan pengujian RFID dan LCD untuk mempermudah dalam mengetahui proses komunikasi dan proses

penyimpanan pada EEPROM berjalan atau tidak serta memperoleh sampel data yang berhasil diproses. RFID dihubungkan dengan Arduino Uno, sedangkan LCD diakses oleh Mikrokontroler ATmega32 melalui rangkaian I2C.

#### 4.5.2 Alat Yang Dibutuhkan

1. Rangkaian Minimum Sistem Mikrokontroler ATmega32.
2. Rangkaian Arduino Uno.
3. RFID.
4. *tag* RFID
5. LCD.
6. Rangkaian I2C.
7. Adaptor 12 Volt dan 5 Volt.

#### 4.5.3 Prosedur Pengujian

1. Hubungkan adaptor 12 Volt dengan Arduino Uno dan 5 Volt dengan Mikrokontroler ATmega32.
2. Hubungkan Arduino Uno, Mikrokontroler ATmega32, LCD dan rangkaian I2C.
3. Hubungkan RFID dengan Arduino Uno.
4. Pastikan program telah di *upload*.
5. Sentuhkan *tag* RFID ke RFID.
6. Kemudian pastikan data tampil pada LCD.

#### 4.5.4 Hasil Pengujian

Dari pengujian yang telah dilakukan berdasarkan prosedur di atas, maka didapatkan bahwa komunikasi I2C dan penyimpanan pada EEPROM berjalan

dengan baik dan hasil proses data yang dikirim, diterima dan disimpan pada EEPROM dapat dilihat pada Tabel 4.2.

**Tabel 4.2** Hasil Sampel Data Proses Komunikasi I2C Dan Penyimpanan Pada EEPROM

NO	ID KARTU	DATA DIKIRIM MASTER	DATA DITERIMA SLAVE	DATA SIMPAN DAN BACA EEPROM
1.	140 129 196 101	140 129 196 101	140 129 196 101	140 129 196 101
2.	196 245 56 165	196 245 56 165	196 245 56 165	196 245 56 165
3.	4 86 9 198	4 86 9 198	4 86 9 198	4 86 9 198
4.	16 95 132 42	16 95 132 42	16 95 132 42	16 95 132 42
5.	109 128 31 203	109 128 31 203	109 128 31 203	109 128 31 203
6.	180 233 6 198	180 233 6 198	180 233 6 198	180 233 6 198
7.	84 168 7 198	84 168 7 198	84 168 7 198	84 168 7 198
8.	112 135 114 42	112 135 114 42	112 135 114 42	112 135 114 42
9.	48 242 136 42	48 242 136 42	48 242 136 42	48 242 136 42
10.	164 255 11 198	164 255 11 198	164 255 11 198	164 255 11 198
11.	4 142 8 198	4 142 8 198	4 142 8 198	4 142 8 198
12.	192 82 132 42	192 82 132 42	192 82 132 42	192 82 132 42
13.	228 108 10 198	228 108 10 198	228 108 10 198	228 108 10 198
14.	48 170 138 42	48 170 138 42	48 170 138 42	48 170 138 42
15.	16 79 140 42	16 79 140 42	16 79 140 42	16 79 140 42
16.	228 223 4 198	228 223 4 198	228 223 4 198	228 223 4 198
17.	20 104 13 198	20 104 13 198	20 104 13 198	20 104 13 198
18.	84 120 11 198	84 120 11 198	84 120 11 198	84 120 11 198
19.	68 110 16 198	68 110 16 198	68 110 16 198	68 110 16 198
20.	228 14 8 198	228 14 8 198	228 14 8 198	228 14 8 198
21.	164 59 14 198	164 59 14 198	164 59 14 198	164 59 14 198
22.	52 234 6 198	52 234 6 198	52 234 6 198	52 234 6 198
23.	212 93 13 198	212 93 13 198	212 93 13 198	212 93 13 198
24.	224 186 142 42	224 186 142 42	224 186 142 42	224 186 142 42
25.	36 127 9 198	36 127 9 198	36 127 9 198	36 127 9 198
26.	36 104 10 198	36 104 10 198	36 104 10 198	36 104 10 198
27.	240 101 142 42	240 101 142 42	240 101 142 42	240 101 142 42
28.	132 83 6 198	132 83 6 198	132 83 6 198	132 83 6 198
29.	160 131 106 42	160 131 106 42	160 131 106 42	160 131 106 42
30.	192 254 108 42	192 254 108 42	192 254 108 42	192 254 108 42

Kesimpulan dari Tabel 4.2 adalah proses pengiriman data pada *master*, penerimaan data pada *slave* dan penulisan serta pembacaan data pada EEPROM memperoleh data yang sama yang membuktikan tidak ada kesalahan data pada *master* dan *slave*.

## 4.6 Pengujian Komponen *Slave*

### 4.6.1 Tujuan Pengujian

Pengujian komponen *slave* ini terdiri dari solenoid *door lock*, *limit switch*, dan Led. Pengujian ini bertujuan untuk mengetahui apakah komponen *slave* dapat terintegrasi dan berjalan dengan baik.

#### 4.6.2 Alat Yang Dibutuhkan

1. Rangkaian Minimum Sistem Mikrokontroler ATmega32.
2. Rangkaian Relay.
3. *Limit Switch*.
4. Solenoid *Door Lock*.
5. Rangkaian Led.
6. Adaptor 12 Volt dan 5 Volt.

#### 4.6.3 Prosedur Pengujian

1. Hubungkan adaptor 5 Volt dengan Mikrokontroler ATmega32 dan *limit switch*
2. Hubungkan *limit switch* dengan Mikrokontroler ATmega32.
3. Hubungkan rangkaian led dengan Mikrokontroler ATmega32.
4. Hubungkan solenoid *door lock* dengan rangkaian relay kemudian hubungkan relay dengan Mikrokontroler ATmega32.
5. Setelah seluruh komponen telah terintegrasi dengan baik, pengujian dilakukan dengan melihat kondisi solenoid *door lock* terkunci atau tidak.
6. Jika *limit switch* ditekan, maka solenoid *door lock* akan terbuka dan led akan menyala, begitu juga sebaliknya.

#### 4.6.4 Hasil Pengujian

Dari pengujian yang telah dilakukan berdasarkan prosedur pengujian di atas, maka didapatkan hasil bahwa komponen *slave* dapat berjalan dengan baik. Dengan pembuktian ketika *limit switch* ditekan, maka solenoid *door lock* akan terbuka dan led akan menyala, begitu juga sebaliknya.

## 4.7 Pengujian Sistem

### 4.7.1 Tujuan Pengujian

Pengujian keseluruhan sistem ini bertujuan untuk melihat kinerja sistem berdasarkan kondisi-kondisi yang akan diterima oleh *locker*. Dalam pengujian sistem ini akan dibuat beberapa skenario dimana dari kondisi tersebut dapat ditarik kesimpulan apakah sistem dapat bekerja dengan baik atau tidak.

### 4.7.2 Alat Yang Dibutuhkan

1. Rangkaian Minimum Sistem Mikrokontroler ATmega32.
2. Rangkaian Arduino Uno.
3. RFID.
4. *tag* RFID
5. LCD.
6. Rangkaian I2C.
7. Adaptor 12 Volt dan 5 Volt.
8. *Locker*.
9. LED masing-masing *locker*.
10. Tombol Keluar.
11. Tombol masing-masing *locker*.
12. Rangkaian *Relay*.
13. *Limit Switch*.
14. Solenoid *Door Lock*.

### 4.7.3 Prosedur Pengujian

1. Memastikan seluruh komponen telah terpasang dan terintegrasi dengan baik

2. Lakukan skenario 1 yaitu kondisi *locker* dalam keadaan kosong. Kemudian lakukan *scan tag* RFID secara acak sampai *locker* penuh. Skenario ini bertujuan untuk mengetahui apakah *locker* dapat diakses dan *user* mendapat *locker* secara acak.
3. Lakukan skenario 2 yaitu kondisi *locker* dalam keadaan penuh. Lakukan *scan tag* RFID yang sudah terdaftar sesuai skenario 1. Setelah itu pencet tombol pada *locker* yang ditunjuk. Kemudian buka dan tutup *locker*. Ulangi skenario 2 sampai 5 kali. Skenario ini bertujuan untuk mengetahui bahwa *user* hanya dapat mengakses *locker* yang telah terdaftar dengan *tag* RFIDnya.
4. Lakukan skenario 3 yaitu kondisi *locker* dalam keadaan penuh. Lakukan *scan tag* RFID yang belum terdaftar pada *locker*. Lakukan skenario 3 sampai 5 kali. Skenario ini bertujuan untuk mengetahui apakah *locker* bisa diakses oleh *user* lain yang belum terdaftar.
5. Lakukan skenario 4 yaitu kondisi awal *locker* dalam keadaan penuh. Lakukan *scan tag* RFID dan menekan tombol keluar secara bersama untuk mengeluarkan semua *tag* RFID yang terdaftar. Kemudian lakukan *scan tag* RFID secara acak sampai *locker* penuh. Skenario ini bertujuan untuk mengetahui bahwa *user* dapat keluar dari *locker* dan *locker* dapat digunakan oleh *user* yang belum terdaftar.
6. Lakukan skenario 5 yaitu kondisi *locker* dalam keadaan penuh atau tidak penuh. Lakukan *scan tag* RFID yang belum terdaftar pada *locker* dan menekan tombol keluar secara bersama. Skenario ini bertujuan untuk

mengetahui bahwa *user* yang belum terdaftar tidak bisa keluar dari *locker* dan tidak akan menghapus pemilik *locker* yang sudah terdaftar.

#### 4.7.4 Hasil Pengujian

1. Hasil skenario 1 yaitu kondisi yang diawali *locker* dalam keadaan kosong dan melakukan *scan tag* RFID secara acak sampai *locker* penuh. Dari Tabel 4.3 dapat dilihat bagaimana hasil nomor *locker* yang diperoleh masing-masing ID dari *Tag* RFID.

**Tabel 4.3** Hasil Skenario 1

No	ID	Locker
1.	4 86 9 198	4
2.	36 76 10 198	1
3.	160 225 102 42	2
4.	224 90 140 142	3

Kesimpulan dari Tabel 4.3 yaitu *user* dapat mengakses *locker* jika masih ada yang kosong dan akan mendapat *locker* secara acak.

2. Hasil skenario 2 yaitu kondisi *locker* dalam keadaan penuh kemudian *Tag* RFID yang telah terdaftar dapat mengakses *locker* yang sesuai dengan nomor *locker* yang diperoleh dari skenario 1. Pengujian dilakukan sebanyak 5 kali untuk memastikan satu buah ID dapat mengakses nomor *locker* yang sama, hasil dapat dilihat pada Tabel 4.4.

**Tabel 4.4** Hasil Skenario 2

No	ID	Locker				
		Hasil 1	Hasil 2	Hasil 3	Hasil 4	Hasil 5
1.	4 86 9 198	4	4	4	4	4
2.	36 76 10 198	1	1	1	1	1
3.	160 225 102 42	2	2	2	2	2
4.	224 90 140 142	3	3	3	3	3

Kesimpulan dari Tabel 4.4 yaitu *user* dengan ID yang sama akan mendapat *locker* yang sama.

- Hasil skenario 3 yaitu kondisi *locker* dalam keadaan penuh, kemudian RFID diakses oleh ID *tag* RFID lain yang belum terdaftar, dengan melakukan prosedur dari ID *tag* RFID yang telah terdaftar ketika ingin mengakses nomer *locker* dari ID *tag* RFID yang telah terdaftar. Hasil dari skenario 3 dapat dilihat pada Tabel 4.5.

**Tabel 4.5** Hasil Skenario 3

No	ID	Locker				
		Hasil 1	Hasil 2	Hasil 3	Hasil 4	Hasil 5
1.	196 245 56 165	0	0	0	0	0
2.	140 129 196 101	0	0	0	0	0

Kesimpulan dari Tabel 4.5 adalah *tag* RFID yang belum terdaftar tidak bisa mengakses *locker*. Angka 0 di Tabel 4.5 menandakan bahwa 2 *tag* RFID di atas belum terdaftar.

- Hasil skenario 4 yaitu kondisi *locker* yang awalnya penuh, kemudian dilakukan prosedur dari ID *tag* RFID yang telah terdaftar untuk membuka *locker*. Selanjutnya ID *tag* RFID lain yang belum terdaftar dapat memperoleh nomer *locker* secara acak karena kondisi *locker* tidak lagi dalam keadaan penuh. Hasil acak dari nomer *locker* ID *Tag* RFID yang baru terdaftar dijabarkan pada Tabel 4.6.

**Tabel 4.6** Hasil Skenario 4

No	ID	Locker
1.	196 245 56 165	2
2.	140 129 196 101	1

3.	36 76 10 198	3
4.	224 90 140 42	4

Kesimpulan dari Tabel 4.6 adalah bahwa *user* yang sudah terdaftar dapat keluar dari *locker*. Hal ini dibuktikan dengan ID pada masing-masing *locker* berbeda dengan hasil skenario 1.

5. Hasil skenario 5 yaitu kondisi *locker* yang diawali dalam keadaan penuh atau tidak penuh, kemudian ID *tag* RFID yang belum terdaftar melakukan prosedur untuk keluar dari *locker* atau telah selesai menggunakan *locker*. Jika hasil pengujian sesuai dengan program maka ID *tag* RFID tersebut tidak dapat membuka *locker*, hasil pengujian dapat dilihat pada Tabel 4.7.

**Tabel 4.7** Hasil Skenario 5

No	ID	Pesan				
		Hasil 1	Hasil 2	Hasil 3	Hasil 4	Hasil 5
1.	4 86 9 198	0	0	0	0	0
2.	160 225 102 42	0	0	0	0	0

Kesimpulan dari Tabel 4.7 adalah *user* yang belum terdaftar tidak bisa keluar dari *locker*. Angka 0 pada Tabel di atas menunjukkan bahwa *tag* RFID tersebut belum terdaftar dan akan menampilkan pesan pada LCD serta tidak akan menghapus daftar pemilik *locker* saat itu yang tersimpan dalam EEPROM. Sehingga daftar pemilik *locker* tetap dan akan dibuktikan pada Tabel 4.8

**Tabel 4.8** Daftar Pemilik *Locker*

No	ID	<i>Locker</i>
1.	196 245 56 165	2
2.	140 129 196 101	1
3.	36 76 10 198	3
4.	224 90 140 42	4

#### 4.8 Hasil Analisa Keseluruhan Sistem

Berdasarkan pengujian-pengujian yang telah dilakukan dapat diketahui bahwa akan ada 5 skenario atau kondisi yang mungkin terjadi pada sistem *locker* otomatis. Kondisi-kondisi tersebut adalah :

1. Skenario 1 yaitu kondisi yang diawali *locker* dalam keadaan kosong dan melakukan *scan tag* RFID secara acak sampai *locker* penuh.
2. Skenario 2 yaitu kondisi *locker* dalam keadaan penuh kemudian *Tag* RFID yang telah terdaftar dapat mengakses *locker* yang sesuai dengan nomor *locker* yang diperoleh dari kondisi yang diawali *locker* dalam keadaan kosong dan melakukan *scan tag* RFID secara acak sampai *locker* penuh.
3. Skenario 3 yaitu kondisi *locker* dalam keadaan penuh, kemudian RFID diakses oleh ID *tag* RFID lain yang belum terdaftar, dengan melakukan prosedur dari ID *tag* RFID yang telah terdaftar ketika ingin mengakses nomor *locker* dari ID *tag* RFID yang telah terdaftar.
4. Skenario 4 yaitu kondisi yang diawali *locker* dalam keadaan penuh. Lakukan *scan tag* RFID dan menekan tombol keluar secara bersama untuk mengeluarkan semua *tag* RFID yang terdaftar. Kemudian lakukan *scan tag* RFID secara acak sampai *locker* penuh.
5. Skenario 5 yaitu kondisi *locker* yang diawali dalam keadaan penuh atau tidak penuh, kemudian ID *tag* RFID yang belum terdaftar melakukan prosedur untuk keluar dari *locker* atau telah selesai menggunakan *locker*.

Dari semua kondisi yang telah diuji, dapat dikatakan semua kondisi dapat berjalan dengan baik sesuai dengan program yang telah dibuat selama *user* melakukan seluruh prosedur penggunaan rancang bangun dengan benar. Sistem

ini akan mempermudah *user* ketika akan menggunakan dan mencari *locker* yang akan digunakan. Selain itu juga komunikasi I2C antara mikrokontroler *master* dengan mikrokontroler *slave* telah berjalan dengan baik, terbukti dari hasil-hasil proses pengiriman data yang dilakukan, begitu juga sebaliknya.

