

BAB V

HASIL DAN PENGUJIAN SISTEM

5.1 HASIL

Dari perancangan alat penuntun jalan bagi penyandang tunanetra yang dirancang dihasilkan yang sederhana dan ringkas yang dapat saling terhubung satu sama lain. Hasil rancangan dapat dilihat pada gambar 5.1 dan 5.2 berikut.



Gambar 5.1 Alat tampak dari atas



Gambar 5.2 Alat tampak dari samping

Dari gambar 5.1 dan 5.2 dapat kita lihat hasil rancangan sesuai dengan skema rancangan sebelumnya.

5.2 PENGUJIAN SISTEM

Sistem yang dirancang meruokan suatu system yang berkaitan dan membutuhkan,sesuai dengan fungsi alat yang dibuat ialah agar penyandang tunanetra dapat berjalan secara mandiri tanpa harus menggunakan tongkat ataupun bimbingan orang lain dalam melakukan aktifitas dalam ruangan ataupun didalam rumah. Karena alat ini terdidri dari beberapa bagian I/O yang saling mendukung menjadikan system dapat bekerja dengan baik,tentu tidak lepas dari beberapa masalah yang telah dilalui dalam perancangan dan pembuatan alat ini. Masih banyak hal-hal baru yang akan kita temui hingga akhirnya akan semakin meminimalkan kekurangan sistem,untuk hal ini dilakukan beberapa langkah kongkrit untuk tujuan pengujian sistem yang akhirnya diharapkan untuk mendapatkan sistem yang lebih sempurna.

Pengujian alat penuntun jalan bagi penyandang tunanetra ini memiliki beberapa tahap, tahapan ini bertujuan untuk memperkecil untuk memperkecil kemungkinan alat tidak berja disaat dialakukan uji coba atau perbedaan hasil yang diinginkan.

Untuk alat yang dirancang dan telah direalisasikan pada tugas akhir ini akan dilakukan pengujian dengan cara :

1. Pengujian dengan perangkat lunak (*software*)
2. Pengujian dengan perangkat Keras (*hardware*)

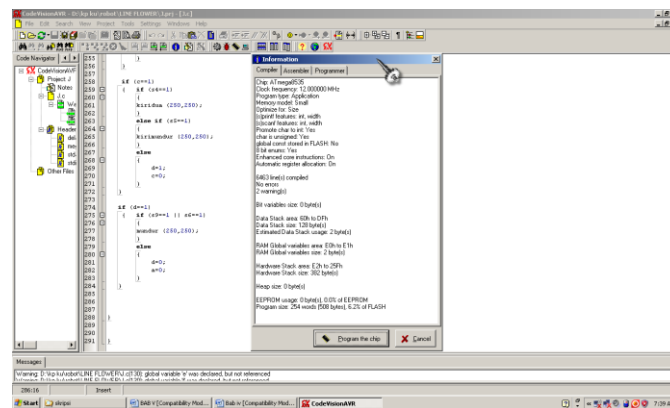
5.1.1 Pengujian Perangkat Lunak (*Software*)

Hal yang pertama dilakukan dalam pengujian perangkat lunak adalah menentukan aplikasi (*Software*) yang akan digunakan. Untuk mikrokontroler Atmega 8535 dan Atmega 16 sinkron dengan banyak bahasa pemrograman seperti bahasa pemrograman Assembler, AVR Studio, bahasa C, Bascom AVR dan lainnya, dalam pembuatan alat ini kami menggunakan *software* Codevision AVR, sebab Codevision AVR bahasa pemrogramannya lebih mudah dipahami dibandingkan yang lainnya karena telah menggunakan bahasa C yang *coding-codingnya* mudah dimengerti.

Codevision AVR merupakan sebuah cross-compiler C, *Integrated Development Environment* (IDE), dan *Auto0c Program Generator* yang didesain untuk mikrokontroler buatan Atmel seri AVR. Codevision AVR dapat dijalankan pada sistem operasi Windows 95, 98, Me, NT4, 2000, XP, dan Windows 7. Cross-compiler C mampu menerjemahkan hampir semua perintah dari bahasa ANSI C, sejauh yang diijinkan oleh arsitektur dari AVR, dengan tambahan beberapa fitur untuk mengambil kelebihan khusus dari arsitektur AVR dan kebutuhan pada sistem *embedded*.

Untuk pengujian software berikutnya yaitu pembuatan *listing program* dengan cara membuka Aplikasi Codevision AVR yang telah terinstal di PC dengan mengeklik dua kali pada icon Codevision AVR maka akan tampil menu seperti pada gambar 4.28.

Setelah program Codevision AVR terbuka maka dimulailah pengaturan chip Atmega dan dilanjutkan pembuatan atau pengetikan *listing program* setelah pengetikan selesai maka tahap berikutnya adalah mengkompile program yang dibuat atau menguji kebenaran *Coding-coding* program yang kita buat dengan cara menekan CTRL+F9 dan apabila program yang kita buat salah maka akan terdapat petunjuk dimana terjadi kesalahan tersebut, dan apabila program yang kita buat benar maka program langsung *mengcompile* program tersebut maka akan tampil perintah seperti gambar dibawah ini.



Gambar 5.3 Compiler Codevision AVR

Setelah tampil menu seperti gambar 5.3 Untuk menginputkan program kemikrokontroler maka *click* tombol *Program* maka dengan otomatis aplikasi Codevision akan mendeteksi mikrokontroler yang kita gunakan, jika mikrokontroler yang kita gunakan tidak terdeteksi maka terdapat kesalahan pada rangkaian downloader yang kita buat.

5.1.2 Pengujian Perangkat Keras (*Hardware*)

5.1.2.1 Pengujian *Downloader*

Tahap pertama dilakukan pengujian *downloader*. Yang pertama dilakukan adalah pemeriksaan rangkaian *regulator* ke mikrokontroler apakah sudah berukuran 5VDC, sebab tegangan yang baik untuk mikrokontroler adalah 5VDC, apabila kurang maupun lebih dapat mengganggu kinerja mikrokontroler bahkan dapat merusak mikrokontroler.

Hal berikutnya yang akan dilakukan adalah menghubungkan PC dengan *downloader* menggunakan *port* paralel (DB25), rangkaian *downloader* terdiri sepasang port paralel, 4 buah resistor 330 ohm, 1 buah *regulator* IC 7805, 1 buah mikrokontroler dan soket 40 pin, dan 1 buah Xtal 12 *Mhz* serta 2 buah kapasitor. Untuk pin 6 DB25 dihubungkan ke pin SCK mikrokontroler yang terletak pada PB.7 melalui resistor 330 ohm, pin 7 DB25 dihubungkan ke pin MOSI mikrokontroler yang terletak pada PB.5 melalui resistor 330 ohm, pin 10 DB25 dihubungkan ke pin MISO yang terletak pada PB.6 melalui resistor 330 ohm dan selanjutnya pin 9 DB25 dihubungkan ke pin RESET mikrokontroler melalui resistor 330 ohm. Sedangkan pin 20 sampai 25 DB25 dihubungkan ke *ground*.

Tabel 5.1 Pengujian Tegangan *Regulator* Ke Mikrokontroler

Tegangan Input (AC)	Tegangan Output (DC)
5	4.8
6	5
9	5
12	5

Langkah selanjutnya adalah menghubungkan *downloader* dengan PC menggunakan kabel dan DB25 (jantan), kemudian buka aplikasi *CodeVision AVR* dan lakukan pengisian program sesuai dengan yang diinginkan, lalu tekan *Ctrl F9* untuk proses *download* program ke mikrokontroler.

5.1.2.2 Pengujian Sensor Jarak SRF04

Rangkaian sensor jarak SRF04 berfungsi mengkonversi setiap perubahan jarak yang dideteksi menjadi besaran tegangan analog. *Output* rangkaian sensor jarak ini berbentuk tegangan *analog*. Setelah rangkaian sensor jarak dibuat seperti gambar 4.3, maka dilakukan tahapan pengujian rangkaian sensor jarak tersebut. Pengujian yang dilakukan yaitu dengan mengukur jarak sebenarnya dengan jarak yang ditampilkan pada LCD.

Setelah dilakukan pengujian terhadap parameter tegangan *output* diperoleh hasil pengukuran seperti pada tabel 5.2.

Tabel 5.2 Hasil Pengukuran Output Rangkaian Sensor Jarak SRF04

Jarak SRF04 ke Dinding	Tampilan pada LCD
2 cm	2 cm
5 cm	5 cm
20 cm	20 cm
50 cm	50 cm
100 cm	100 cm
200 cm	200 cm

Untuk pengujian program terhadap rangkaian sensor jarak SRF04 dijelaskan pada sub bab pengujian rangkaian mikrokontroler ATmega8535. Hal ini dikarenakan di dalam mikrokontroler ATmega8535 sudah terdapat rangkaian

ADC (*Analog Digital Converter*). Sehingga *output* rangkaian sensor jarak SRF04 sudah dapat diproses langsung oleh mikrokontroler ATmega8535 tersebut. Hanya saja sebelum program pengujian sensor jarak SRF04 dibuat, harus dipahami prinsip pengujian program pengaktifan ADC yang ada pada mikrokontroler ATmega8535. Jadi pada saat pengujian program pengaktifan ADC pada rangkaian mikrokontroler ATmega8535, *output* tegangan *analog* dari sensor jarak SRF04 dapat diproses dengan baik oleh mikrokontroler ATmega8535. Sehingga pada sub bab ini tidak perlu dibuat lagi program pengujian rangkaian sensor jarak SRF04.

Tabel 5.3 Hasil Pengukuran Tegangan pada Mikrokontroler

Logika Port	Tegangan Pada Port A, B, C Dan D ATmega8535
Low (0)	0,00 V
High (1)	4,8 - 4,92 V

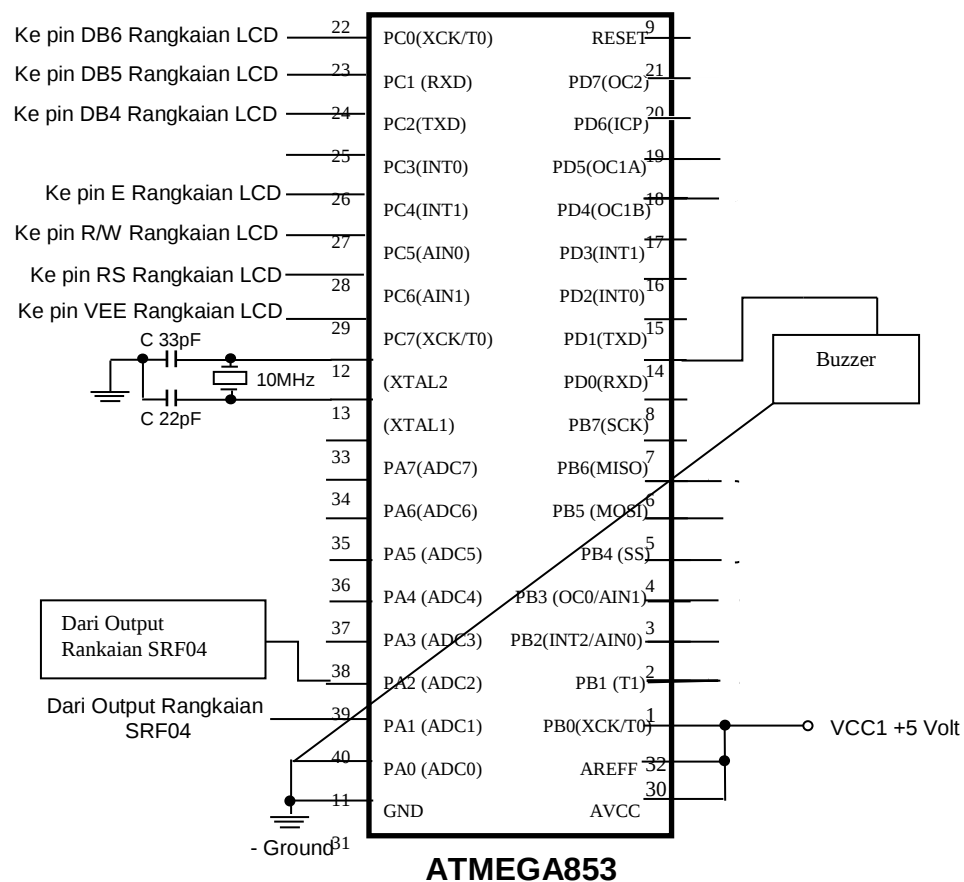
Mikrokontroler bekerja pada dua kondisi logika yaitu kondisi *low* (0) dimana tegangan yang terbaca pada instrumen pengukuran tegangan didapatkan tegangan *port* sebesar 0,00 V yang berarti sistem dalam batas ideal. Logika yang kedua yaitu kondisi *high* (1) dimana tegangan yang terbaca pada instrumen pengukuran tegangan didapatkan tegangan *port* sebesar 4,92 V yang berarti sistem masih dalam batas ideal.

Dari kondisi pengukuran di atas, maka rangkaian mikrokontroler ATmega8535 telah dapat bekerja dengan baik.

Di dalam *chip* mikrokontroler ATmega8535 terdapat rangkaian ADC yang mempunyai 8 buah *input analog* ADC (ADC0 s/d ADC7). Untuk mengaktifkan

ADC pada mikrokontroler ATmega8535 tersebut diperlukan program pengendali pengaktifan ADC.

Lalu pengujian berikutnya yaitu menguji pengaktifan rangkaian ADC yang terdapat dalam mikrokontroler ATmega8535. Pengujian yang dilakukan yaitu dengan membuat rangkaian dan program pengujian. Untuk pengujian pengaktifan rangkaian ADC yang ada dalam mikrokontroler ATmega8535 yaitu dengan menggunakan rangkaian sensor jarak SRF04 pada gambar 4.3 dan rangkaian LCD pada gambar 4.5. Rangkaian LCD tersebut akan menampilkan perubahan-perubahan besaran tegangan analog yang masuk ke mikrokontroler ATmega8535 dan menampilkan ke LCD berupa angka desimal dan huruf abjad.



Gambar 5.4 Pengujian Pengaktifan ADC Pada Mikrokontroler ATmega8535

Rangkaian sensor jarak SRF04 terhubung ke ADC 0 (PA.0) mikrokontroler ATmega8535. Sedangkan pin data (D0 s/d D7) rangkaian LCD terhubung ke port C (PC.0 s/d PC.7) mikrokontroler ATmega8535. Untuk pin RS dan E pada LCD masing-masing terhubung ke PC.6 dan PC.4 mikrokontroler ATmega8535. Rangkaian pengujian tersebut dapat dilihat pada gambar 5.4.

Setelah rangkaian pengujian pengaktifan ADC pada gambar 5.4 dibuat, lalu dilakukan pembuatan program pengendali pengaktifan ADC di mikrokontroler tersebut. Program pengendali pengaktifan ADC yang dimaksud adalah sebagai berikut :

```
#include <mega8535.h>
#include <stdio.h>
#include <delay.h>

unsigned int count=0;
unsigned char dpn1;
unsigned char jrk1;
unsigned char dpn2;
unsigned char jrk2;
unsigned char kata1[16];
unsigned char kata2[16];

// Alphanumeric LCD Module functions
#asm
.equ __lcd_port=0x15 ;PORTC
#endasm
#include <lcd.h>

unsigned char depan1()
{
count=0;
PORTA.1=1;
delay_us(15);
PORTA.1=0;
```

```

//menunggu hingga ECHO = 1
while(PINA.0==0){};
//mencacah untuk mengukur waktu lamanya pulsa 1 dari ECHO
while(PINA.0==1)
{count++;
};
//mengkonversikan hasil cacahan menjadi jarak dalam cm
dpn1=count*5/232;
return dpn1;

}

unsigned char depan2()
{
count=0;
PORTA.3=1;
delay_us(15);
PORTA.3=0;
//menunggu hingga ECHO = 1
while(PINA.2==0){};
//mencacah untuk mengukur waktu lamanya pulsa 1 dari ECHO
while(PINA.2==1)
{count++;
};
//mengkonversikan hasil cacahan menjadi jarak dalam cm
dpn2=count*5/232;
return dpn2;

}

void s_depan1()
{

    jrk1=depan1();
    sprintf(kata1,"Jarak =%3d cm" ,jrk1);

    //lcd_clear();
    lcd_gotoxy(1,0);
    lcd_puts(kata1);
    delay_ms(50);

}

void s_depan2()
{

```

```

    jrk2=depan2();
    sprintf(kata1,"Jarak =%3d cm" ,jrk2);

    //lcd_clear();
    lcd_gotoxy(1,1);
    lcd_puts(kata1);
    delay_ms(50);

}

void main(void)
{
// Declare your local variables here

// Input/Output Ports initialization
// Port A initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In
Func0=In
// State7=P State6=P State5=P State4=P State3=P State2=P State1=P State0=P
PORTA=0x05;
DDRA=0x0A;

// Port B initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In
Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTB=0x00;
DDRB=0xFF;

// Port C initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=Out
Func1=Out Func0=Out
// State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0 State0=0
PORTC=0x00;
DDRC=0xFF;

// Port D initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In
Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTD=0x00;
DDRD=0xff;

// Timer/Counter 0 initialization
// Clock source: System Clock

```

```

// Clock value: Timer 0 Stopped
// Mode: Normal top=FFh
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer 1 Stopped
// Mode: Normal top=FFFFh
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer 1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer 2 Stopped
// Mode: Normal top=FFh
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
MCUCR=0x00;
MCUCSR=0x00;

```

```
// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x00;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;

// LCD module initialization
lcd_init(16);

while (1)
{
    s_depan2();
    s_depan1();
    if (dpn1 <= 40)
    {
        PORTD.6=1;
    }
    else if (dpn2 <= 45)
    {
        PORTD.6=1;
        delay_ms(100);
        PORTD.6=0;
    }
    else
    {
        PORTD.6=0;
    }
};
}
```

Setelah program *download* ke mikrokontroler ATmega8535, diperoleh hasil sebagai berikut.

Tabel 5.4 Pengujian Program SRF04 dan LCD

Kondisi Sensor Jarak SRF04	Tampilan Jarak Pada LCD
2 cm	2 cm
5 cm	5 cm
10 cm	10 cm
30 cm	30 cm
100 cm	100 cm
200 cm	200 cm

Dari tabel 5.4, dapat diambil kesimpulan bahwa rangkaian ADC pada mikrokontroler ATmega8535 telah dapat bekerja dengan baik.

5.1.2.3 Rangkaian LCD (*Liquid Crystal Display*) 16x2

Rangkaian LCD (*Liquid Crystal Display*) 16x2 berfungsi sebagai penampil suhu level kecepatan kipas angin. Dalam hal ini pengujian rangkaian LCD hanya dilakukan lewat pembuatan program (*software*), bukan dengan pengukuran parameter logika ‘0’ atau logika ‘1’ yang diberikan ke rangkaian LCD tersebut. Hal ini disebabkan untuk menampilkan teks pada LCD diperlukan beberapa perintah, seperti inisialisasi LCD dan beberapa perintah yang lain.

Jadi setelah rangkaian LCD (*Liquid Crystal Display*) dibuat seperti pada gambar 4.5, program pengujian dibuat, *download* ke mikrokontroler ATmega8535 kemudian diamati hasilnya pada LCD. Jika teks yang ditampilkan sesuai dengan yang ada diprogram, berarti rangkaian LCD dan program

pengujiannya sudah bekerja dengan baik. Program pengujian rangkaian LCD tersebut adalah seperti berikut:

```
{
    lcd_gotoxy (1,0);
    lcd_putsf ("JARAK:");
    delay_ms(1500);
    hapus();
}
```

Setelah program *download* ke mikrokontroler ATmega8535, diperoleh hasil seperti pada tabel 5.5.

Tabel 5.5 Pengujian Rangkaian LCD 16x2

Teks Pada Program Yang Ingin Ditampilkan	Teks Yang Ditampilkan Pada LCD 16x2
("JARAK:")	JARAK:
("BELOK_KANAN")	BELOK_KANAN
("BELOK_KIRI")	BELOK_KIRI
("BERHENTI")	BERHENTI

Dari hasil pengujian pada tabel 5.5 dapat dikatakan bahwa rangkaian LCD (*Liquid Crystal Display*) 16x2 serta program pengujiannya telah bekerja dengan baik.

5.1.2.4 Pengujian Rangkaian Buzzer

Buzzer menggunakan relay sebagai saklar agar bunyi *buzzer* lebih kencang maka perlu tegangan 12V. Port yang digunakan untuk *buzzer* ialah PORTB.1, jika nilai konversi sensor jarak SRF 04 bernilai <100 cm maka tegangan 5v akan sampai pada *buzzer* sehingga menimbulkan bunyi. Berikut tabel pengujian pengaktif *buzzer*.

Tabel 5.6 Pengujian Rangkaian Buzzer

Posisi SRF04	Jarak	Buzzer
Sensor depan	≤ 60 cm	Berbunyi Panjang
Sensor Bawah	≤ 50 cm	Berpunyi putus-putus

5.1.2.5 Pengujian Rangkaian Keseluruhan

Dalam pengujian rangkaian keseluruhan adalah menggabungkan semua rangkaian yang dibuat seperti rangkaian mikrokontroler ATmega8535, rangkaian sensor jarak SRF04, rangkaian LCD (*Liquid Display Digital*) 16x2 dan rangkaian buzzer. Pada pengujian rangkaian keseluruhan hanya jika semua inputnya bernilai *high* (1) atau hidup maka outputnya akan bernilai *high* (1) atau hidup pula, selain itu semua outputnya bernilai *low* (0) atau mati. Rangkaian yang berperan sebagai *input* yaitu rangkaian ATmega8535 dan rangkaian sensor jarak SRF04, sedangkan rangkaian yang berperan sebagai *output* yaitu rangkaian LCD dan rangkaian buzzer.

Tabel 5.7 Pengujian Rangkaian Keseluruhan

Kondisi sensor SRF 04		Tampilan Ke LCD		Kondisi Buzzer	
Sensor Depan	Sensor Bawah	Jarak Sensor Depan	Jarak Sensor Bawah	Tegangan Buzzer	Buzzer
≤ 60 cm	≤ 50 cm	≤ 60 cm	≤ 50 cm	5	Bunyi Panjang
≤ 60 cm	≥ 51 cm	≤ 60 cm	≥ 51 cm	5	Bunyi Panjang
≥ 61 cm	≤ 50 cm	≥ 61 cm	≤ 50 cm	5	Bunyi Putus-Putus
≥ 61 cm	≥ 51 cm	≥ 61 cm	≥ 51 cm	0	Diam

5.2 ANALISA PENGUJIAN

Dari proses yang telah dilakukan terhadap pengujian rangkaian baik dari segi perangkat keras (*hardware*) maupun perangkat lunak (*software*) nya, dapat dikatakan bahwa alat yang dirancang sudah sesuai dengan spesifikasi yang diinginkan. Dari segi perangkat keras, mikrokontroler ATmega8535 mempunyai batas toleransi tegangan kerja 2.7V-5.5V sehingga memungkinkan pemakaian komponen yang tidak terlalu presisi. Sebagai contoh, jika tegangan untuk sumber daya (VCC) mikrokontroler ATmega8535 sebesar 3.5 VDC, berarti tegangan tergolong kecil. Namun hal tersebut tidak menjadi masalah jika masih dalam batas toleransi. Toleransi ini memberikan kemudahan bagi perancangan alat yang menggunakan mikrokontroler ATmega8535, karena tidak perlu khawatir jika sewaktu-waktu *supply* tegangan VCC tidak stabil.

Untuk mengkonversi setiap perubahan jarak menjadi besaran tegangan analog digunakan rangkaian sensor jarak SRF04. Setelah dilakukan pengujian program dan pengukuran *output* rangkaian sensor jarak SRF04 tersebut diperoleh hasil yang sama dengan *datasheet* SRF04 tersebut. Dengan demikian, hasil pengukuran rangkaian sensor jarak SRF04 dengan besaran *centi meter* (cm) telah bekerja dengan baik.

Untuk menampilkan informasi jarak dalam besaran '*centi meter*' dalam bentuk angka digunakan rangkaian LCD (*Liquid Crystal Display*). Pengujian dilakukan dengan membuat program pengujian rangkaian LCD (*Liquid Crystal Display*) 16x2. Setelah diamati ternyata teks yang ditampilkan pada layar LCD sama dengan teks yang dibuat pada program pengujian yaitu dapat menampilkan

judul dari sistem yang dirancang. Dengan demikian rangkaian LCD 16x2 dinyatakan telah bekerja dengan baik.

Keberhasilan sistem pengendalian pada alat yang dibuat ini tidak terlepas dari program pengendali yang dibuat. Dalam hal ini program pengendali disimpan dalam mikrokontroler ATmega8535. Jika program pengendali yang dibuat benar, maka pengendalian dapat dilakukan dengan benar. Begitu juga sebaliknya jika program pengendali yang dibuat salah, maka pengendalian yang dilakukan juga ikut salah. Bagaimanapun juga mikrokontroler ATmega8535 hanyalah *Single Chip Microcomputer* (SCM), yang memiliki arsitektur yang sama dengan arsitektur komputer dan dalam istilah komputer dikenal prinsip bahwa '*garbage in garbage out*', maksudnya : 'sampah yang masuk, sampah juga yang keluar'.

Di sisi lain peranan manusia (*brainware*) sebagai pembuat program pengendali (*software*) maupun perancang perangkat keras (*hardware*) turut mendukung keberhasilan pembuatan alat yang dirancang.