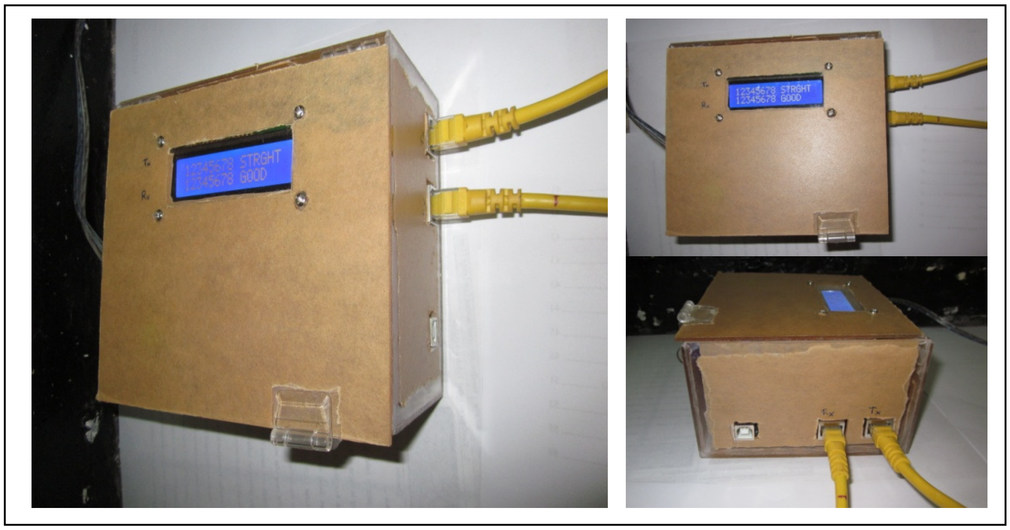
Lampiran 1 Gambar dan Deskripsi Alat

Alat Penguji Kabel LAN Berbasis Mikrokontroler

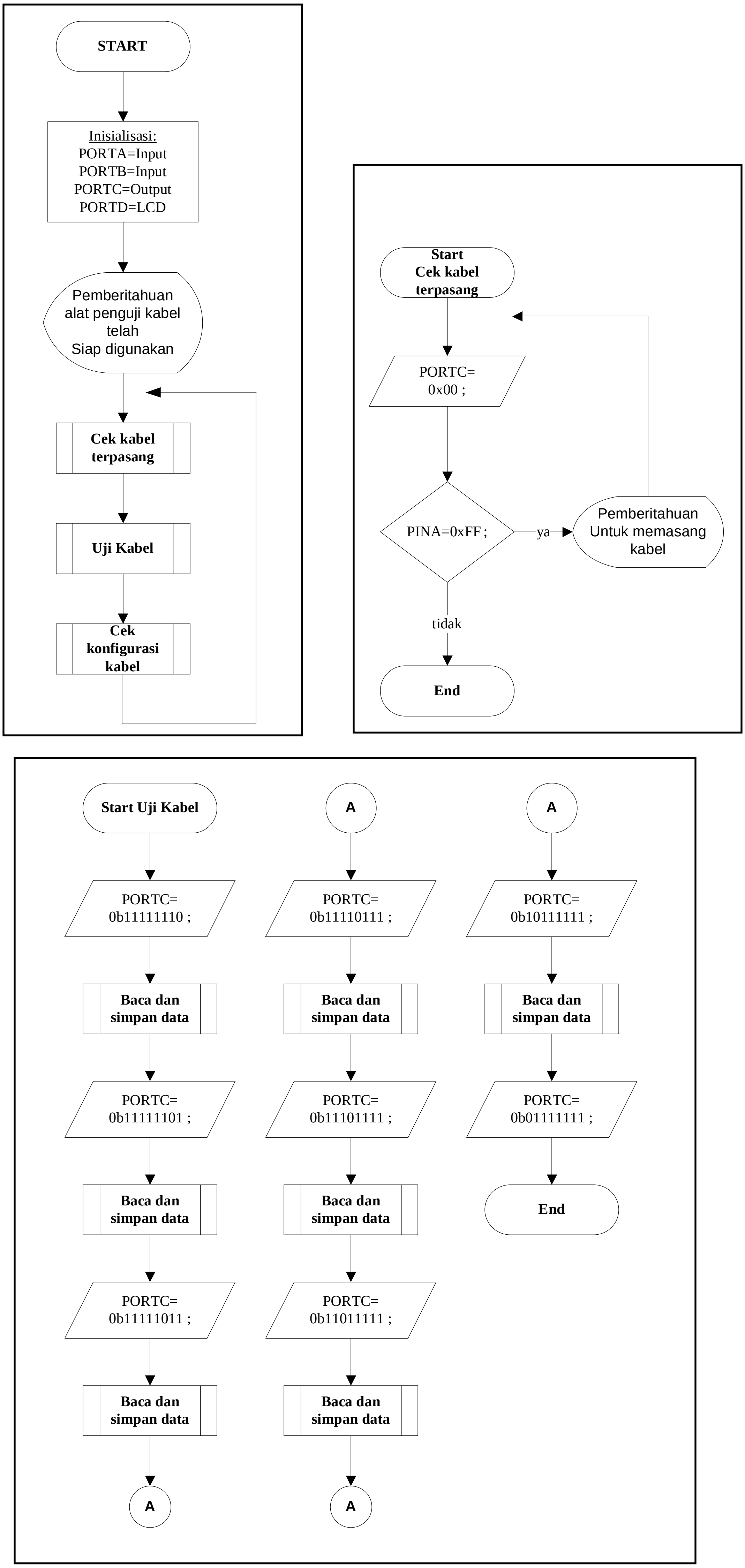


Gambar alat penguji kabel LAN Berbasis Mikrokontroler

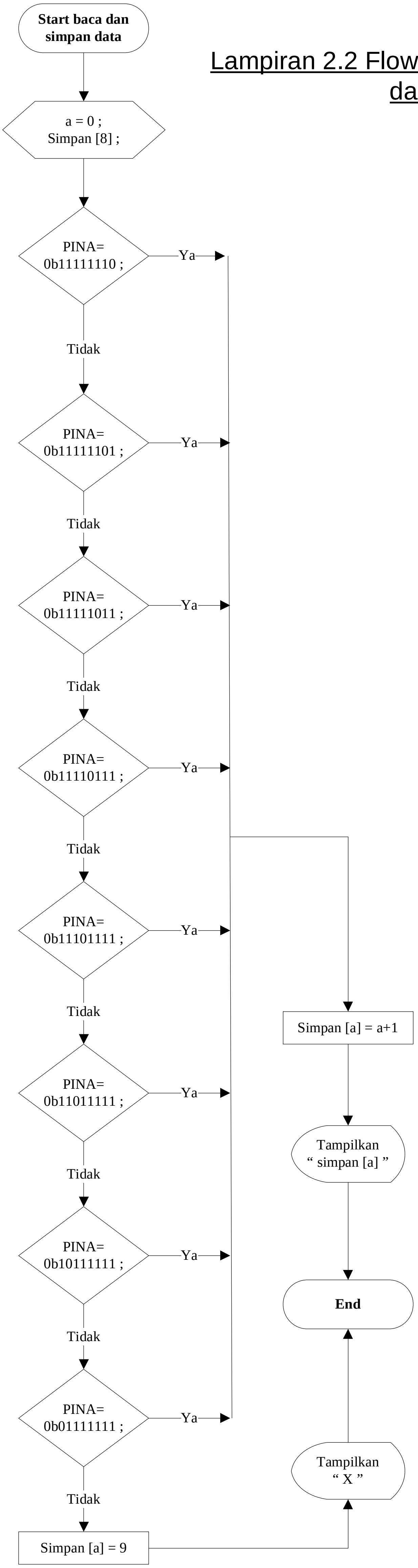
Alat penguji kabel LAN berbasis mikrokontroler dapat mengetahui urutan kabel UTP yang terpasang pada pin konektor RJ45 *male*. Mampu membedakan konfigurasi kabel straight, cross, dan rollover. Menggunakan tampilan LCD 16 kolom 2 baris, dengan latar belakang lampu berwarna biru gelap yang berfungsi sebagai antarmuka bagi pengguna.

Alat penguji kabel LAN menggunakan mikrokontroler sebagai komponen utama yang bertugas sebagai pengirim data dan pembaca data serta mengelola data untuk kemudian ditampilkan pada LCD 16x2. Dibuat untuk mengetahui konfigurasi kabel serta sambungan kabel pada pin-pin konektor RJ45. Alat penguji kabel LAN terdiri dari satu buah modul mikroAVR yang telah terpasang mikrokontroler ATmega8535, dua buah konektor RJ45 *female,* dan modul LCD 16x2 serta sumber tegangan DC +9V atau +5V

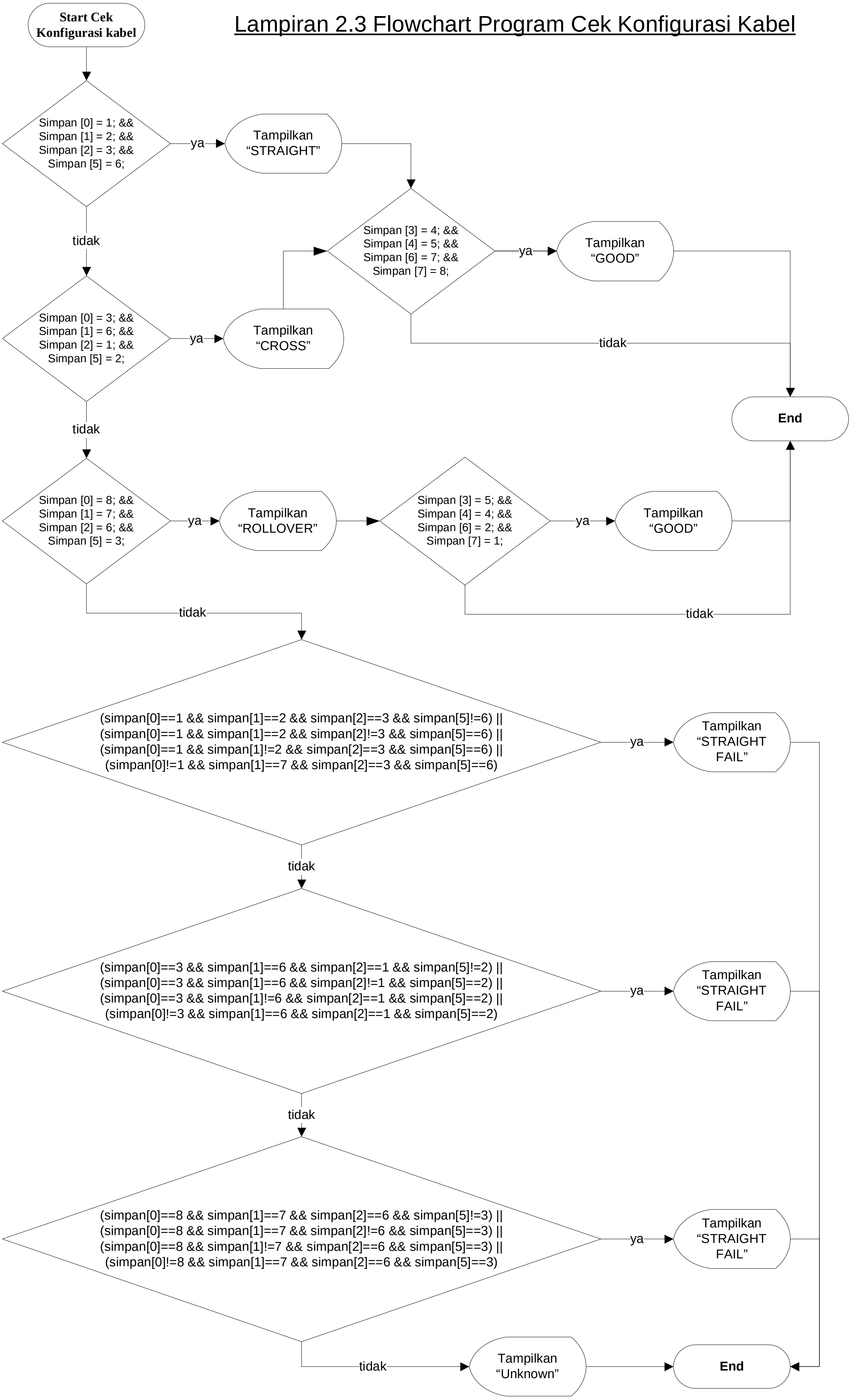
Lampiran 2.1 Flowchart Program Secara Umum, Cek Kabel Terpasang, dan Uji kabel



Lampiran 2.2 Flowchart Program Baca, Simpan dan Tampilkan Data



Lampiran 2.3 Flowchart Program Cek Konfigurasi Kabel


Start Cek Konfigurasi kabel
Simpan [0] = 1; && Simpan [1] = 2; && Simpan [2] = 3; && Simpan [5] = 6;
ya
Tampilkan "STRAIGHT"
tidak
Simpan [3] = 4; && Simpan [4] = 5; && Simpan [6] = 7; && Simpan [7] = 8;
ya
Tampilkan "GOOD"
Simpan [0] = 3; && Simpan [1] = 6; && Simpan [2] = 1; && Simpan [5] = 2;
ya
Tampilkan "CROSS"
tidak
End
tidak
Simpan [0] = 8; && Simpan [1] = 7; && Simpan [2] = 6; && Simpan [5] = 3;
ya
Tampilkan "ROLLOVER"
Simpan [3] = 5; && Simpan [4] = 4; && Simpan [6] = 2; && Simpan [7] = 1;
ya
Tampilkan "GOOD"
tidak
tidak
(simpan[0]==1 && simpan[1]==2 && simpan[2]==3 && simpan[5]!=6) || (simpan[0]==1 && simpan[1]==2 && simpan[2]!=3 && simpan[5]==6) || (simpan[0]==1 && simpan[1]!=2 && simpan[2]==3 && simpan[5]==6) || (simpan[0]!=1 && simpan[1]==7 && simpan[2]==3 && simpan[5]==6)
ya
Tampilkan "STRAIGHT FAIL"
tidak
(simpan[0]==3 && simpan[1]==6 && simpan[2]==1 && simpan[5]!=2) || (simpan[0]==3 && simpan[1]==6 && simpan[2]!=1 && simpan[5]==2) || (simpan[0]==3 && simpan[1]!=6 && simpan[2]==1 && simpan[5]==2) || (simpan[0]!=3 && simpan[1]==6 && simpan[2]==1 && simpan[5]==2)
ya
Tampilkan "STRAIGHT FAIL"
tidak
(simpan[0]==8 && simpan[1]==7 && simpan[2]==6 && simpan[5]!=3) || (simpan[0]==8 && simpan[1]==7 && simpan[2]!=6 && simpan[5]==3) || (simpan[0]==8 && simpan[1]!=7 && simpan[2]==6 && simpan[5]==3) || (simpan[0]!=8 && simpan[1]==7 && simpan[2]==6 && simpan[5]==3)
ya
Tampilkan "STRAIGHT FAIL"
tidak
Tampilkan "Unknown"
End

Lampiran 3.1 Kode Program Alat Penguji Kabel LAN Berbasis Mikrokontroler

```c
/*****************************************************
This program was produced by the
CodeWizardAVR V2.05.1b Evaluation
Automatic Program Generator
© Copyright 1998-2011 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com

Project : Skripsi LAN Tester
Version :
Date    : 04/06/2011
Author  : Fajar Ramadhan
Company : Pendidikan Teknik Elektronika UNJ
Comments:

Chip type               : ATmega8535L
Program type            : Application
AVR Core Clock frequency: 8,000000 MHz
Memory model            : Small
External RAM size       : 0
Data Stack size         : 128
*****************************************************/

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

// Alphanumeric LCD Module functions
#include <alcd.h>

// Declare your global variables here
char buffer_lcd[]="LAN TESTER READY                ";
char lcd_number = 16;
char buf[2],buff[2];
char simpan[8],simpan2[8],reset[8];
char a,b,c,d;

void tampilan()
{
unsigned int i,j,k;
unsigned int data_len  = strlen(buffer_lcd);
lcd_clear();
lcd_gotoxy(0,0);
lcd_putsf("ELEKTRONIKA>>UNJ");

for (i=lcd_number; i>=0; --i){
    if (i > lcd_number)break;
    lcd_gotoxy(i,1);
    for (j=0; j<(lcd_number-i); j++){
        lcd_putchar(buffer_lcd[j]);
        }
    delay_ms(200);
    };
k=0;
for (i=0; i<=data_len; i++){
    k++;
    lcd_gotoxy(0,1);
    for (j=0; j<lcd_number; j++){
        if (buffer_lcd[j+k] == NULL){
            return;
            }
        lcd_putchar(buffer_lcd[j+k]);
        }
    delay_ms(200);
    };
}

void warning()      // indikator jika kabel belum terpasang
{
 b=0;c=4;PORTC=0x00;
 if (PINA==0xFF)
 {
 lcd_clear();
 for (a=0;a<=7;a++)
 {simpan2[a]=reset[a];}
 loop:
    lcd_gotoxy(0,0);
    lcd_putsf("  NOT CONNECT!  ");
    delay_ms(1000);
 loop1:
    lcd_gotoxy(c,1);
    lcd_putsf(">");
    delay_ms(1000);
    c++;b++;
  if (b==8)
  {
     delay_ms(500);
     lcd_gotoxy(0,0);
     lcd_putsf("     Please     ");
     lcd_gotoxy(0,1);
     lcd_putsf(" Insert Cable!! ");
     b=0;c=4;
  loop2:
      if (PINA==0xFF)
      {
        b++;
        delay_ms(1000);
            if (b==10)
            {b=0;lcd_clear();goto loop;}
        else
          goto loop2;
       }
  }
  if (PINA==0xFF)
  {goto loop1;}
       else
          {b=0;c=4;lcd_clear();}
 }
}

 void kirim_data ()
{
    PORTC=~d;
    d<<=1;
    delay_ms(500);
}

void tampil_data()   //menampilkan hasil baca data yang telah disimpan
 {
   lcd_gotoxy(a,0);
   sprintf(buf,"%x",a+1);
   lcd_puts(buf);
  if(simpan[a]==9)
    {lcd_gotoxy(a,1);
    lcd_puts("X");}
    else
    {lcd_gotoxy(a,1);
     sprintf(buff,"%x",simpan[a]);
     lcd_puts(buff);}
 }

void baca_data ()    // baca data kabel lalu simpan
 {
    if (PINA==~0b00000001)
    {simpan[a]=1;}
    else if (PINA==~0b00000010)
    {simpan[a]=2;}
    else if (PINA==~0b00000100)
    {simpan[a]=3;}
    else if (PINA==~0b00001000)
    {simpan[a]=4;}
    else if (PINA==~0b00010000)
    {simpan[a]=5;}
    else if (PINA==~0b00100000)
    {simpan[a]=6;}
    else if (PINA==~0b01000000)
    {simpan[a]=7;}
    else if (PINA==~0b10000000)
    {simpan[a]=8;}
    else
    {simpan[a]=9;}
  }

void cek_kabel()     //konfigurasi kabel
{
   if (simpan[0]==1 && simpan[1]==2 && simpan[2]==3 && simpan[5]==6)
   {
```

```
 if (simpan[3]==4 && simpan[4]==5 && simpan[6]==7 && simpan[7]==8)
 {
  lcd_gotoxy(9,0);
  lcd_putsf("STRGHT ");
  lcd_gotoxy(9,1);
  lcd_putsf("GOOD   ");
 }
  else
  {
   lcd_gotoxy(9,0);
   lcd_putsf("STRGHT ");
   lcd_gotoxy(9,1);
   lcd_putsf("       ");
  }
}
else if ((simpan[0]==1 && simpan[1]==2 && simpan[2]==3 && simpan[5]!=6) ||
         (simpan[0]==1 && simpan[1]==2 && simpan[2]!=3 && simpan[5]==6) ||
         (simpan[0]==1 && simpan[1]!=2 && simpan[2]==3 && simpan[5]==6) ||
         (simpan[0]!=1 && simpan[1]==2 && simpan[2]==3 && simpan[5]==6))
{
 lcd_gotoxy(9,0);
 lcd_putsf("STRGHT ");
 lcd_gotoxy(9,1);
 lcd_putsf("FAIL!  ");
 }
else if (simpan[0]==3 && simpan[1]==6 && simpan[2]==1 && simpan[5]==2)
{
 lcd_gotoxy(9,0);
 lcd_putsf("CROSS  ");
 lcd_gotoxy(9,1);
 lcd_putsf("       ");
 if (simpan[3]==4 && simpan[4]==5 && simpan[6]==7 && simpan[7]==8)
 {
 lcd_gotoxy(9,1);
 lcd_putsf("GOOD   ");
 }
}
else if ((simpan[0]==3 && simpan[1]==6 && simpan[2]==1 && simpan[5]!=2) ||
         (simpan[0]==3 && simpan[1]==6 && simpan[2]!=1 && simpan[5]==2) ||
         (simpan[0]==3 && simpan[1]!=6 && simpan[2]==1 && simpan[5]==2) ||
         (simpan[0]!=3 && simpan[1]==6 && simpan[2]==1 && simpan[5]==2))
{
 lcd_gotoxy(9,0);
 lcd_putsf("CROSS  ");
 lcd_gotoxy(9,1);
 lcd_putsf("FAIL!  ");
}
else if (simpan[0]==8 && simpan[1]==7 && simpan[2]==6 && simpan[5]==3)
{
 if (simpan[3]==5 && simpan[4]==4 && simpan[6]==2 && simpan[7]==1)
 {
 lcd_gotoxy(9,0);
 lcd_putsf("ROLLOVR");
 lcd_gotoxy(9,1);
 lcd_putsf("GOOD   ");
 }
  else
  {
   lcd_gotoxy(9,0);
   lcd_putsf("ROLL   ");
   lcd_gotoxy(9,1);
   lcd_putsf("OVER   ");
  }
}
else if ((simpan[0]==8 && simpan[1]==7 && simpan[2]==6 && simpan[5]!=3) ||
         (simpan[0]==8 && simpan[1]==7 && simpan[2]!=6 && simpan[5]==3) ||
         (simpan[0]==8 && simpan[1]!=7 && simpan[2]==6 && simpan[5]==3) ||
         (simpan[0]!=8 && simpan[1]==7 && simpan[2]==6 && simpan[5]==3) )
{
 lcd_gotoxy(9,0);
 lcd_putsf("ROLLOVR");
 lcd_gotoxy(9,1);
 lcd_putsf("FAIL!  ");
}
else if (simpan[0]==9 && simpan[1]==9 && simpan[2]==9 && simpan[3]==9
      && simpan[4]==9 && simpan[5]==9 && simpan[6]==9 && simpan[7]==9)
{lcd_clear();warning();}
else
{
    lcd_gotoxy(9,0);
    lcd_putsf("UNKNOWN");
    lcd_gotoxy(9,1);
    lcd_putsf("CONFIG!");
   }
}

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

// Input/Output Ports initialization
// Port A initialization
PORTA=0xFF;
DDRA=0x00;

// Port B initialization
PORTB=0xFF;
DDRB=0x00;

// Port C initialization
PORTC=0x00;
DDRC=0xFF;

// Port D initialization
PORTD=0x00;
DDRD=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=0xFF
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer1 Stopped
// Mode: Normal top=0xFFFF
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer1 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: Timer2 Stopped
// Mode: Normal top=0xFF
// 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;

```

```
// USART initialization
// USART disabled
UCSRB=0x00;

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

// ADC initialization
// ADC disabled
ADCSRA=0x00;

// SPI initialization
// SPI disabled
SPCR=0x00;

// TWI initialization
// TWI disabled
TWCR=0x00;

// Alphanumeric LCD initialization
// Connections are specified in the
// Project|Configure|C Compiler|Libraries|Alphanumeric LCD menu:
// RS - PORTD Bit 0
// RD - PORTD Bit 1
// EN - PORTD Bit 2
// D4 - PORTD Bit 4
// D5 - PORTD Bit 5
// D6 - PORTD Bit 6
// D7 - PORTD Bit 7
// Characters/line: 16
lcd_init(16);
tampilan();
lcd_clear();

while (1)
      {
      // Place your code here
      b=0;c=0;d=1;
      ulang:
       PORTC=0x00;
       delay_ms(500);
        if (PINA==0xFF)
        {warning();}
        else
        {
           for(a=0;a<=7;a++)
            {
            if(c<=7)
              {lcd_gotoxy(a,0);
               sprintf(buf,"%x",a+1);
               lcd_puts(buf);c++;}
               kirim_data();
               baca_data();
            if(simpan[a]!=simpan2[a])
              {simpan2[a]=simpan[a];
               tampil_data();
               delay_ms(500);
               }
            else
            {b++;}
            }
        if(c==8)
        {cek_kabel();c++;d=1;goto ulang;}
        else if(b!=8)
        {cek_kabel();b=0;d=1;goto ulang;}
        else
        {b=0;d=1;goto ulang;}
        }
      }
}
```

Lampiran 3.2 Kode Program Uji Modul

```c
/*****************************************************
This program was produced by the
CodeWizardAVR V2.05.1b Evaluation
Automatic Program Generator
© Copyright 1998-2011 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com
Project :
Version :
Date    : 04/06/2011
Author  : Freeware, for evaluation and non-commercial use only
Company :
Comments:
Chip type               : ATmega8535L
Program type            : Application
AVR Core Clock frequency: 8,000000 MHz
Memory model            : Small
External RAM size       : 0
Data Stack size         : 128
*****************************************************/

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

// Alphanumeric LCD Module functions
#include <alcd.h>

// Declare your global variables here
char buffer_lcd[]="PROGRAM UJI MODUL                ";
char lcd_number = 16;
//char buf[3],buff[8];
//unsigned char simpan[10];
//int a,b,d=1;
// Function prototype
//void tampilan(void);
//void data (void);
// Display routine :D
void tampilan()
{
unsigned int i,j,k;
unsigned int data_len  = strlen(buffer_lcd);
//_lcd_ready();
lcd_clear();
lcd_gotoxy(0,0);
lcd_putsf("#FAJAR RAMADHAN#");

for (i=lcd_number; i>=0; --i){
    if (i > lcd_number)break;
    lcd_gotoxy(i,1);
    for (j=0; j<(lcd_number-i); j++){
        lcd_putchar(buffer_lcd[j]);
        }
    delay_ms(150);
    };
k=0;
for (i=0; i<=data_len; i++){
    k++;
    lcd_gotoxy(0,1);
    for (j=0; j<lcd_number; j++){
        if (buffer_lcd[j+k] == NULL){
            return;
            }
        lcd_putchar(buffer_lcd[j+k]);
        }
    delay_ms(150);
    };
}


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

// Input/Output Ports initialization
// Port A initialization
```

```
PORTA=0xFF;
DDRA=0x00;

// Port B initialization
PORTB=0xFF;
DDRB=0x00;

// Port C initialization
PORTC=0x00;
DDRC=0xFF;

// Port D initialization
PORTD=0x00;
DDRD=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=0xFF
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer1 Stopped
// Mode: Normal top=0xFFFF
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer1 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: Timer2 Stopped
// Mode: Normal top=0xFF
// 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;

// USART initialization
// USART disabled
UCSRB=0x00;

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

// ADC initialization
// ADC disabled
ADCSRA=0x00;
```

```

// SPI initialization
// SPI disabled
SPCR=0x00;

// TWI initialization
// TWI disabled
TWCR=0x00;

// Alphanumeric LCD initialization
// Connections are specified in the
// Project|Configure|C Compiler|Libraries|Alphanumeric LCD menu:
// RS - PORTD Bit 0
// RD - PORTD Bit 1
// EN - PORTD Bit 2
// D4 - PORTD Bit 4
// D5 - PORTD Bit 5
// D6 - PORTD Bit 6
// D7 - PORTD Bit 7
// Characters/line: 16
lcd_init(16);
//_lcd_ready();

tampilan();
lcd_clear();

while (1)
    {
      // Place your code here

      lcd_gotoxy(0,0);
      lcd_putsf("  UJI PORT I/O  ");
      PORTC=0x00;
      delay_ms(2000);
    if (PINA==0x00)
     {lcd_gotoxy(0,1);
      lcd_putsf(" PORT I/O = OK! ");
     }
    else
     {
      lcd_gotoxy(0,1);
      lcd_putsf("PERIKSA PORT I/O");
     }
    }
}
```

Lampiran 3.2 Kode ProgramVoltmeter dengan ADC

```
/*****************************************************
This program was produced by the
CodeWizardAVR V2.05.0 Professional
Automatic Program Generator
© Copyright 1998-2010 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com

Project :
Version :
Date    : 22/06/2011
Author  :
Company :
Comments:
Chip type               : ATmega8535L
Program type            : Application
AVR Core Clock frequency: 8,000000 MHz
Memory model            : Small
External RAM size       : 0
Data Stack size         : 128
*****************************************************/

#include <mega8535.h>
#include <math.h>
#include <delay.h>
#include <string.h>
#include <stdio.h>

// Alphanumeric LCD Module functions
#include <alcd.h>

#define ADC_VREF_TYPE 0x40

// Read the AD conversion result
unsigned int read_adc(unsigned char adc_input)
{
ADMUX=adc_input | (ADC_VREF_TYPE & 0xff);
// Delay needed for the stabilization of the ADC input voltage
delay_us(10);
// Start the AD conversion
ADCSRA|=0x40;
// Wait for the AD conversion to complete
while ((ADCSRA & 0x10)==0);
ADCSRA|=0x10;
return ADCW;
}

// Declare your global variables here
char buffer[16];
float tegangan; //Untuk data perhitungan tegangan (float)
unsigned int temporary; //Untuk temporary data ADC (komparasi)

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

// Input/Output Ports initialization
PORTA=0xFF;
DDRA=0x00;

PORTB=0x00;
DDRB=0x00;

PORTC=0x00;
DDRC=0xFF;

PORTD=0x00;
DDRD=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=0xFF
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

```

```
// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer1 Stopped
// Mode: Normal top=0xFFFF
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer1 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: Timer2 Stopped
// Mode: Normal top=0xFF
// 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;

// USART initialization
// USART disabled
UCSRB=0x00;

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

// ADC initialization
// ADC Clock frequency: 125,000 kHz
// ADC Voltage Reference: AVCC pin
// ADC High Speed Mode: Off
// ADC Auto Trigger Source: ADC Stopped
ADMUX=ADC_VREF_TYPE & 0xff;
ADCSRA=0x86;
SFIOR&=0xEF;

// SPI initialization
// SPI disabled
SPCR=0x00;

// TWI initialization
// TWI disabled
TWCR=0x00;

// Alphanumeric LCD initialization
// Connections specified in the
// Project|Configure|C Compiler|Libraries|Alphanumeric LCD menu:
// RS - PORTD Bit 0
// RD - PORTD Bit 1
// EN - PORTD Bit 2
// D4 - PORTD Bit 4
// D5 - PORTD Bit 5
// D6 - PORTD Bit 6
// D7 - PORTD Bit 7
// Characters/line: 16
lcd_init(16);
while (1)
      {
      // Place your code here

         read_adc(0);
         if (temporary != read_adc(0)){
            tegangan = 5-((read_adc(0)*4.8828125)/1000);
            lcd_gotoxy(0,0);
            lcd_putsf("   - 737rr -   ");
            lcd_gotoxy(0,1);
            sprintf(buffer,"VOLT:%f V",tegangan);
            lcd_puts(buffer);
            temporary = read_adc(0);
            delay_ms(1000);
        }
      }
}
```