

## DAFTAR LAMPIRAN

//\*\*\*\*\*

*This program was produced by the*

*CodeWizardAVR V2.05.0 Evaluation*

*Automatic Program Generator*

*© Copyright 1998-2010 Pavel Haiduc, HP InfoTech s.r.l.*

*<http://www.hpinfotech.com>*

*Project :*

*Version :*

*Date : 29/02/2016*

*Author : Freeware, for evaluation and non-commercial use only*

*Company :*

*Comments:*

*Chip type : ATmega16*

*Program type : Application*

*AVR Core Clock frequency: 11.059200 MHz*

*Memory model : Small*



*External RAM size* : 0

*Data Stack size* : 256

\*\*\*\*\*/

*#include <mega16.h>*

*#include <delay.h>*

*// Alphanumeric LCD Module functions*

*#include <alcd.h>*

*// Standard Input/Output functions*

*#include <stdio.h>*

*char data;*

*int i;*

*char buf[33];*

*char bif[33];*

*char a=0,b=0,c=0,d=0;*

*#define ADC\_VREF\_TYPE 0x20*

*// Read the 8 most significant bits*



```
// of the AD conversion result

unsigned char 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

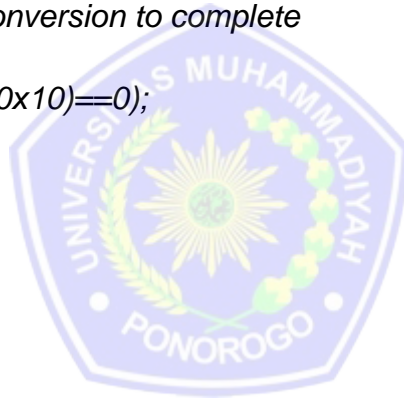
    ADCSRA|=0x40;

    // Wait for the AD conversion to complete

    while ((ADCSRA & 0x10)==0);

    ADCSRA|=0x10;

    return ADCH;
}
```



```
//=====
==

void enter(void)
{
    putchar(13);
}

//=====
==
```

```
void buzzer()
{
    PORTB.3=1;

    delay_ms(500);

    PORTB.3=0;

    delay_ms(500);
}
```

```
//=====
```

```
void pesan()
{
    if(data==1){printf("!!RADIO ON AIR!!");}
    if(data==2){printf("!!RADIO OFF AIR!!");}
    if(data==3){printf("!!RADIO BERMASALAH!!");}
}
```

```
//=====
```

```
void sms()
{
    for(i=0;i<2;i++)
    {
```

```
printf("AT+CMGS=");

putchar(34);

printf("085655517953");

putchar(34);

enter();

pesan();

putchar(26);

delay_ms(500);

}

}

// Declare your global variables here

void main(void)

{

// Declare your local variables here

char data_serial,data_adc;

char i,j,radio_on;

// Input/Output Ports initialization

// Port A 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
```

```
PORTA=0x00;
```

```
DDRA=0x00;
```

```
// Port B initialization
```

```
// Func7=In Func6=In Func5=In Func4=In Func3=Out Func2=Out  
Func1=In Func0=In
```

```
// State7=T State6=T State5=T State4=T State3=0 State2=0 State1=P  
State0=T
```

```
PORTB=0x02;
```

```
DDRB=0x0C;
```



```
// Port C 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
```

```
PORTC=0x00;
```

```
DDRC=0x00;
```

```
// 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=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*

*// Communication Parameters: 8 Data, 1 Stop, No Parity*

*// USART Receiver: On*

*// USART Transmitter: On*

*// USART Mode: Asynchronous*

*// USART Baud Rate: 9600*

*UCSRA=0x00;*

*UCSRB=0x18;*

*UCSRC=0x86;*

*UBRRH=0x00;*

*UBRRL=0x47;*

*// Analog Comparator initialization*

*// Analog Comparator: Off*

*// Analog Comparator Input Capture by Timer/Counter 1: Off*

*ACSR=0x80;*

*SFIOR=0x00;*



*// ADC initialization*

*// ADC Clock frequency: 691.200 kHz*

*// ADC Voltage Reference: AREF pin*

*// ADC Auto Trigger Source: ADC Stopped*

*// Only the 8 most significant bits of*

*// the AD conversion result are used*

*ADMUX=ADC\_VREF\_TYPE & 0xff;*

*ADCSRA=0x84;*

```
// 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 - PORTC Bit 0
```

```
// RD - PORTC Bit 1
```

```
// EN - PORTC Bit 2
```

```
// D4 - PORTC Bit 4
```

```
// D5 - PORTC Bit 5
```

```
// D6 - PORTC Bit 6
```

```
// D7 - PORTC Bit 7
```

```
// Characters/line: 16
```

```
lcd_init(16);
```



```

while (1)

{

// Place your code here

data_adc=read_adc(0);

sprintf(buf,"DATA: %i",data_adc,data_adc);

lcd_gotoxy(0,0);

lcd_puts(buf);

sprintf(bif,"UDR: %i",data_serial,data_serial);

lcd_gotoxy(0,1);

lcd_puts(bif);

delay_ms(500);

lcd_clear();

if ((UCSRA & (1 << RXC))) data_serial = UDR;

if (data_serial == 13)

{

PORTB.2=1;      //on relay

delay_ms(500);

PORTB.2=0;

```



```
delay_ms(5000);  
  
printf("ATH"); //hang up  
  
putchar(13);
```

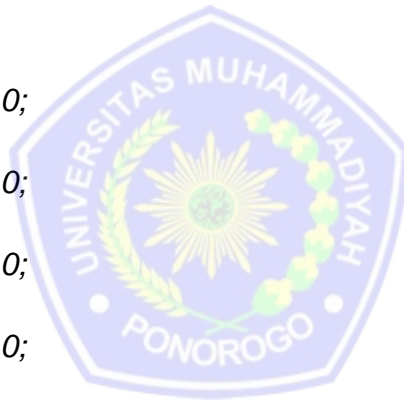
```
data=1;  
  
sms();  
  
delay_ms(10000);
```

```
data_serial=10;  
data_serial=10;  
data_serial=10;  
data_serial=10;  
data_serial=10;  
data_serial=10;  
data_serial=10;  
data_serial=10;  
  
a=1;
```

```
}
```

```
while(a==1)
```

```
{
```



```
if(PINB.1 == 1) //("!!RADIO ON AIR!!")

{

    //radio_on = 0;          //diinisialisasi radio trouble

    i=1;

    j=1;

    do

    {

        data_adc=read_adc(0);

        sprintf(buf,"DATA: %i",data_adc,data_adc);

        lcd_gotoxy(0,0);

        lcd_puts(buf);

        sprintf(bif,"UDR: %i",data_serial,data_serial);

        lcd_gotoxy(0,1);

        lcd_puts(bif);

        delay_ms(500);

        lcd_clear();

        if ((UCSRA & (1 << RXC))) data_serial = UDR;

        if (data_serial == 13)

        {
```

```
PORTB.2=1;    //on relay
```

```
delay_ms(500);
```

```
PORTB.2=0;
```

```
delay_ms(5000);
```

```
printf("ATH"); //hang up
```

```
putchar(13);
```

```
data=2;
```

```
sms();
```

```
delay_ms(10000);
```

```
a=0;
```

```
}
```

```
data_adc=read_adc(0);
```

```
if (data_adc>=95) // adc radio on
```

```
{
```

```
    //radio_on = 1; //jika data > 23 maka radio tidak trouble
```

```
    break;
```

```
}
```

```
delay_ms(1000);
```

```
i++;
```



```
}  
  
while (i<10);  
  
do  
  
{  
  
data_adc=read_adc(0);  
  
sprintf(buf,"DATA: %i",data_adc,data_adc);  
  
lcd_gotoxy(0,0);  
  
lcd_puts(buf);  
  
sprintf(bif,"UDR: %i",data_serial,data_serial);  
  
lcd_gotoxy(0,1);  
  
lcd_puts(bif);  
  
delay_ms(500);  
  
lcd_clear();  
  
  
if ((UCSRA & (1 << RXC))) data_serial = UDR;  
  
if (data_serial == 13)  
  
{  
  
PORTB.2=1;      //on relay  
  
delay_ms(500);
```



```
PORTB.2=0;

delay_ms(5000);

printf("ATH"); //hang up

putchar(13);

data=2;

sms();

delay_ms(10000);

a=0;
}
data_adc=read_adc(0);
if (data_adc<=95) //adc radio trobel
{

//radio_on = 1; //jika data > 23 maka radio tidak trouble

buzzer();

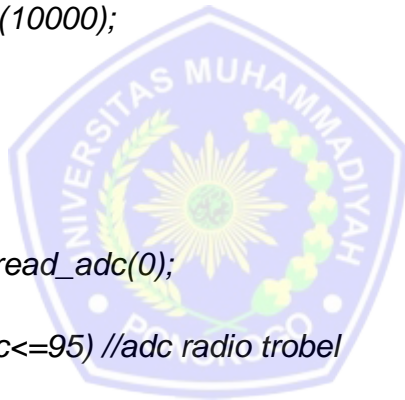
buzzer();

buzzer();

data=3;

sms();

delay_ms(10000);
```



```
break;  
}  
delay_ms(1000);  
j++;  
}  
while (j<10);  
}  
}  
}
```

