

LAMPIRANA Listing Program

```
/******  
Chip type           : ATmega8  
Program type       : Application  
AVR Core Clock frequency : 8,000000 MHz  
Memory model      : Small  
External RAM size  : 0  
Data Stack size   : 256  
*****/  
  
#include <mega8.h>  
  
// I2C Bus functions  
#include <i2c.h>  
  
// DS1307 Real Time Clock functions  
#include <ds1307.h>  
  
// Alphanumeric LCD functions  
#include <alcd.h>  
// delay library  
#include <delay.h>  
// standrt i/o library  
#include <stdio.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;  
}  
  
#define s1 PINB.1  
#define s2 PINB.0  
#define s3 PIND.7
```

```

#define s4 PIND.6
#define s5 PIND.5

#define buzzer PORTB.2

#define leda PORTB.5
#define ledb PORTB.4
#define ledc PORTB.3

#define vsensor 0

char lcd_buffer[33];

// Declare your global variables here

// ram
unsigned char jam,menit,detik;
bit tanda_alarm=0,mode_simulasi=0,buzz=0;
float voltmeter;

// eeprom
eeprom int alarmjam[3]={7,7,7};
eeprom int alarmmenit[3]={1,3,5};

eeprom unsigned char alarmcount=0;

float read_sensor(){
float volt,voltout;
float voltmax=3.5,involtmax=20.0;
int data=read_adc(vsensor);
// rumus sensor tegangan
// r1= 4,7k r2= 1k
// involt max= 20.0 volt (sesuai kebutuhan yang peting hasil perhitungan voltmax
// tidak lebih dari 5v)
// mencari voltmax ( max 5V), voltmax=r2/(r1+r2)*involtmax
// voltmax=1k/(4,7+1)*20.0
// voltmax= 3.50 volt

volt=data*((float)5/1023); // mengubah nilai adc ke tegangan
voltout=volt*((float)involtmax/voltmax); // mengubah nilai tegangan kecil ke
// tegangan sensor
return voltout;
}

```

```

void set_jam()
{
int x=0,sjam,smenit,sdetik;
sjam=jam;
smenit=menit;
sdetik=0;
buzzer=1;
leda=0;
ledb=0;
ledc=0;
lcd_clear();
delay_ms(200);
while(1){
// masukan nilai setingan ke RTC
rtc_set_time(sjam,smenit,sdetik);

// pilih menu
if(!s1){x++;delay_ms(150);}
if(!s2){x--;delay_ms(150);}
// batas nilai menu
if(x>1)x=0;
if(x<0)x=1;

// menu 0 (set jam)
if(x==0){
lcd_gotoxy(0,0);
lcd_putsf("Set Jam      ");
lcd_gotoxy(0,1);
sprintf(lcd_buffer," [%02d]:%02d:%02d  ",sjam,smenit,sdetik);
lcd_puts(lcd_buffer);
// naik turunkan angka
if(!s3){sjam++;delay_ms(150);}
if(!s4){sjam--;delay_ms(150);}
// batas angka
if(sjam>23)sjam=0;
if(sjam<0)sjam=23;
}

//menu 1 (set menit)
if(x==1){
lcd_gotoxy(0,0);
lcd_putsf("Set Menit      ");
lcd_gotoxy(0,1);
sprintf(lcd_buffer," %02d:[%02d]:%02d  ",sjam,smenit,sdetik);
lcd_puts(lcd_buffer);
}
}

```

```

if(!s3){smenit++;delay_ms(150);}
if(!s4){smenit--;delay_ms(150);}
if(smenit>59)smenit=0;
if(smenit<0)smenit=59;
}
// refresh
delay_ms(50);
// keluar dari menu
if(!s5){break;}
}

lcd_clear();
delay_ms(200);

}

void set_alarm_jam()
{
int x=0,y=0;
buzzer=1;
leda=0;
ledb=0;
ledc=0;
lcd_clear();
delay_ms(200);
while(1){

// pilih menu
if(!s1){y++;delay_ms(150);}
if(!s2){y--;delay_ms(150);}
// batas nilai menu
if(y>1){y=0;x++;}
if(y<0){y=1;x--;}

if(x>2){x=0;}
if(x<0){x=2;}

// menu 0 (set jam)
if(y==0){
lcd_gotoxy(0,0);
sprintf(lcd_buffer,"Alarm %d Jam ",x+1);
lcd_puts(lcd_buffer);
lcd_gotoxy(0,1);
sprintf(lcd_buffer," [%02d]:%02d ",alarmjam[x],alarmmenit[x]);
lcd_puts(lcd_buffer);
// naik turunkan angka

```

```

if(!s3){alarmjam[x]++;delay_ms(150);}
if(!s4){alarmjam[x]--;delay_ms(150);}
// batas angka
if(alarmjam[x]>23)alarmjam[x]=0;
if(alarmjam[x]<0)alarmjam[x]=23;
}

// menu 1 (set menit)
if(y==1){
lcd_gotoxy(0,0);
sprintf(lcd_buffer,"Alarm %d Menit",x+1);
lcd_puts(lcd_buffer);
lcd_gotoxy(0,1);
sprintf(lcd_buffer," %02d:[%02d]  ",alarmjam[x],alarmmenit[x]);
lcd_puts(lcd_buffer);
// naik turunkan angka
if(!s3){alarmmenit[x]++;delay_ms(150);}
if(!s4){alarmmenit[x]--;delay_ms(150);}
// batas angka
if(alarmmenit[x]>59)alarmmenit[x]=0;
if(alarmmenit[x]<0)alarmmenit[x]=59;
}
// refresh
delay_ms(50);
// keluar dari menu
if(!s5){break;}
}
// set 0 alarm count

lcd_clear();
delay_ms(200);

}

// suara alarm
void suaraalarm(int brapax,int jeda){
int i;
// pengulangan bunyi bip
for(i=0;i<brapax;i++){
buzzer=0;
delay_ms(100);
buzzer=1;
delay_ms(100);
}
// jeda buzzer off

```

```

buzzer=1;
delay_ms(jeda);

}

// program utama
void program_utama(){
voltmeter=read_sensor();
rtc_get_time(&jam,&menit,&detik);      // Ambil data dari RTC jam,menit,detik

// tampilakn di lcd
lcd_clear();
lcd_gotoxy(0,0);
sprintf(lcd_buffer,"TIME %02d:%02d:%02d ",jam,menit,detik);
lcd_puts(lcd_buffer);
lcd_gotoxy(0,1);
sprintf(lcd_buffer,"ALM:%d V:%.2f",alarmcount+1,voltmeter);
lcd_puts(lcd_buffer);

// tekan s1 untuk masuk set jam
if(!s1)set_jam();
// tekan s2 untuk masuk set alarm
if(!s2)set_alarm_jam();
// tekan s3 untuk set 0 alarm counter
if(!s3){
alarmcount=0;
leda=0;
ledb=0;
ledc=0;
}

// jika jam dan menit sama dengan parameter aktifkan tanda alarm
if(jam==alarmjam[alarmcount]&&menit==alarmmenit[alarmcount])
    tanda_alarm=1;
else tanda_alarm=0;

// jika jam dan menit sama dengan parameter dan tanda alarm bernilai 1 aktifkan
buzzer
if(tanda_alarm==1){
// buzzer on
buzz=1;
// komdisi led
if(alarmcount==0){leda=1;ledb=0;ledc=0;suaraalarm(1,200);}
if(alarmcount==1){leda=1;ledb=1;ledc=0;suaraalarm(2,200);}
if(alarmcount==2){leda=1;ledb=1;ledc=1;suaraalarm(3,200);}
}

```

```

}
else{
// alarm count mencacah setelah buzzer mulai mati
if(buzz==1)alarmcount++;
// buzzer off
buzz=0;

}

// delay refresh
delay_ms(50);
}

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

// Input/Output Ports initialization
// Port B initialization
// Func7=Out Func6=In Func5=Out Func4=Out Func3=Out Func2=Out Func1=In
Func0=In
// State7=0 State6=T State5=0 State4=0 State3=0 State2=1 State1=P State0=P
PORTB=0x07;
DDRB=0xBC;

// Port C initialization
// Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// 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=P State6=P State5=P State4=T State3=T State2=T State1=T State0=T
PORTD=0xE0;
DDRD=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
TCCR0=0x00;
TCNT0=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
MCUCR=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: 1000,000 kHz
// ADC Voltage Reference: AVCC pin
ADMUX=ADC_VREF_TYPE & 0xff;
ADCSRA=0x83;

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

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

// I2C Bus initialization
// I2C Port: PORTC
// I2C SDA bit: 4
// I2C SCL bit: 5
// Bit Rate: 100 kHz
// Note: I2C settings are specified in the
// Project|Configure|C Compiler|Libraries|I2C menu.
i2c_init();

// DS1307 Real Time Clock initialization
// Square wave output on pin SQW/OUT: Off
// SQW/OUT pin state: 0
rtc_init(0,0,0);

// Alphanumeric LCD initialization
// Connections are specified in the
// Project|Configure|C Compiler|Libraries|Alphanumeric LCD menu:
// RS - PORTD Bit 0
// RD - PORTC Bit 6
// EN - PORTD Bit 1
// D4 - PORTD Bit 2
// D5 - PORTD Bit 3
// D6 - PORTD Bit 4
// D7 - PORTB Bit 6
// Characters/line: 16
lcd_init(16);

lcd_clear();
lcd_gotoxy(0,0);

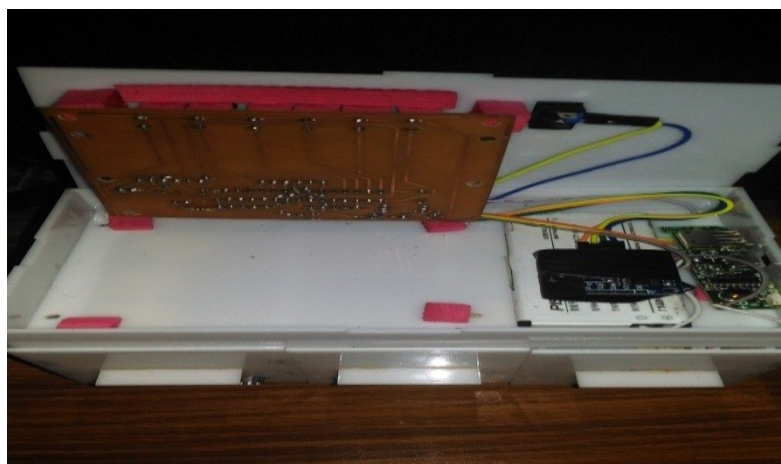
```

```
lcd_putsf("Alarm Peningat");  
lcd_gotoxy(0,1);  
lcd_putsf("Minum Obat");  
delay_ms(1000);
```

```
while (1)  
{  
    // Place your code here  
    // panggil program utama  
    program_utama();  
}
```

```
}
```

**LAMPIRAN FOTO PENELITIAN MEDICINE REMINDER TOOL
ALARAM BASED MICROCONTROLLER ATMEGA 8**



(Gambar Lampiran Saat Pembuatan Dan Penelitian Alat)