LAMPIRAN 1

```
/*****************************
*************************

This program was created by
the

CodeWizardAVR V2.60 Evaluation

Automatic Program Generator

© Copyright 1998-2012 Pavel
Haiduc, HP InfoTech s.r.l.

http://www.hpinfotech.com

Project : Monitoring BPM

Version : 1

Date    : 03/08/2016

Author  : Fajar

Company : UNNOCS

Comments:

Chip type               :
ATmega8535

Program type            :
Application

AVR Core Clock frequency:
12,000000 MHz

Memory model            :
Small

External RAM size       : 0

Data Stack size         : 128

*****************************
************************/

#include <mega16.h>

#include <stdio.h>

#include <stdlib.h>

#include <delay.h>

unsigned char str[16],ksr[16];

unsigned char bpm1[33];

unsigned int suhu,V,Z, bpm;

int detik;

// Alphanumeric LCD functions

#include <alcd.h>

// Declare your global
variables here

// Timer1 overflow interrupt
service routine

interrupt [TIM1_OVF] void
timer1_ovf_isr(void)

{

// Reinitialize Timer1 value

TCNT1H=0x48E5 >> 8;

TCNT1L=0x48E5 & 0xff;

// Place your code here

 detik++;

}

#define ADC_VREF_TYPE
((0<<REFS1) | (0<<REFS0) |
(0<<ADLAR))

// Read the AD conversion
result

unsigned int read_adc(unsigned
char adc_input)

{

ADMUX=adc_input |
ADC_VREF_TYPE;
```

```
// Delay needed for the stabilization of the ADC input voltage

delay_us(10);

// Start the AD conversion
46875     48e5

ADCSRA|=(1<<ADSC);

// Wait for the AD conversion to complete

while ((ADCSRA & (1<<ADIF))==0);

ADCSRA|=(1<<ADIF);

return ADCW;}

void waktu()

{

if(detik<=60)

    {

    sprintf(ksr,"%ds",detik);

    lcd_gotoxy(12,0);

    lcd_puts(ksr);

    }}

void bpm_START()

{TCCR0=0x06;}

void bpm_STOP()

{TCCR0=0x00;}

void indikator_SUHU()

{

lcd_clear();

if(suhu>=(364/10)&&suhu<=(37,6/100))

{PORTD.0=1;PORTD.1=0;   //led hijau aktif led merah off

lcd_gotoxy(12,1);

lcd_putsf("GOOD");

}

else

{PORTD.0=0;PORTD.1=1;

lcd_gotoxy(12,1);

lcd_putsf("BAD");

}}

void indikator_BPM()

{

if(bpm>=60&&bpm<=100)

{

PORTD.3=0;PORTD.4=1;

lcd_gotoxy(12,0);

lcd_putsf("GOOD");

}

else

{

PORTD.3=1;PORTD.4=0;

lcd_gotoxy(12,0);

lcd_putsf("BAD");}}

void suhu_hitung()

{

V=read_adc(0);

Z=V*5/1024;

 suhu=V*100;
```

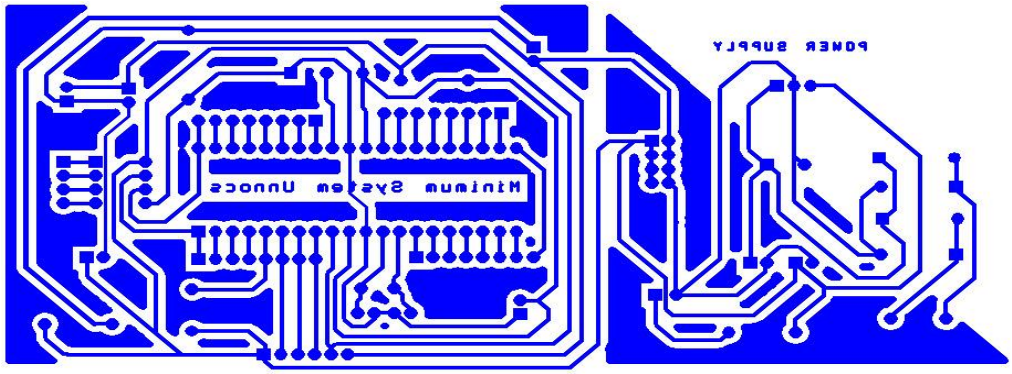
```
//suhu=(float)V/2;
lcd_gotoxy(0,1);
lcd_putsf("SUHU:");
sprintf(str,"%i",suhu);
lcd_gotoxy(5,1);
lcd_puts(str);
lcd_putchar(223);
lcd_putsf("C");
lcd_gotoxy(8,1);
delay_ms(1000);
indikator_SUHU();
 }
void bpm_hitung()
{
waktu();
lcd_gotoxy(0,0);
lcd_putsf("BPM :");
bpm_START();
bpm=TCNT0;
sprintf(bpm1,"%i",bpm);
lcd_gotoxy(5,0);
lcd_puts(bpm1);
lcd_gotoxy(9,0);
lcd_putsf("BM");
if(detik>=60)
{
bpm_STOP();
indikator_BPM();
}}
void main(void)
{
// Declare your local
variables here
// Input/Output Ports
initialization
// Port A initialization
// Function: Bit7=In Bit6=In
Bit5=In Bit4=In Bit3=In
Bit2=In Bit1=In Bit0=In
DDRA=(0<<DDA7) | (0<<DDA6) |
(0<<DDA5) | (0<<DDA4) |
(0<<DDA3) | (0<<DDA2) |
(0<<DDA1) | (0<<DDA0);
// State: Bit7=T Bit6=T Bit5=T
Bit4=T Bit3=T Bit2=T Bit1=T
Bit0=T
PORTA=(0<<PORTA7) |
(0<<PORTA6) | (0<<PORTA5) |
(0<<PORTA4) | (0<<PORTA3) |
(0<<PORTA2) | (0<<PORTA1) |
(0<<PORTA0);
// Port B initialization
// Function: Bit7=In Bit6=In
Bit5=In Bit4=In Bit3=In
Bit2=In Bit1=In Bit0=In
DDRB=(0<<DDB7) | (0<<DDB6) |
(0<<DDB5) | (0<<DDB4) |
(0<<DDB3) | (0<<DDB2) |
(0<<DDB1) | (0<<DDB0);
// State: Bit7=T Bit6=T Bit5=T
Bit4=T Bit3=T Bit2=T Bit1=T
Bit0=P
PORTB=(0<<PORTB7) |
(0<<PORTB6) | (0<<PORTB5) |
(0<<PORTB4) | (0<<PORTB3) |
```

```
(0<<PORTB2) | (0<<PORTB1) |
(1<<PORTB0);

// Port C initialization

// Function: Bit7=In Bit6=In
Bit5=In Bit4=In Bit3=In
Bit2=In Bit1=In Bit0=In

DDRC=(0<<DDC7) | (0<<DDC6) |
(0<<DDC5) | (0<<DDC4) |
(0<<DDC3) | (0<<DDC2) |
(0<<DDC1) | (0<<DDC0);

// State: Bit7=T Bit6=T Bit5=T
Bit4=T Bit3=T Bit2=T Bit1=T
Bit0=T

PORTC=(0<<PORTC7) |
(0<<PORTC6) | (0<<PORTC5) |
(0<<PORTC4) | (0<<PORTC3) |
(0<<PORTC2) | (0<<PORTC1) |
(0<<PORTC0);

// Port D initialization

// Function: Bit7=In Bit6=In
Bit5=In Bit4=Out Bit3=Out
Bit2=In Bit1=Out Bit0=Out

DDRD=(0<<DDD7) | (0<<DDD6) |
(1<<DDD5) | (1<<DDD4) |
(1<<DDD3) | (0<<DDD2) |
(1<<DDD1) | (1<<DDD0);

// State: Bit7=T Bit6=T Bit5=T
Bit4=0 Bit3=0 Bit2=P Bit1=0
Bit0=0

PORTD=(0<<PORTD7) |
(0<<PORTD6) | (0<<PORTD5) |
(0<<PORTD4) | (0<<PORTD3) |
(1<<PORTD2) | (0<<PORTD1) |
(0<<PORTD0);

// Timer/Counter 0
initialization

// Clock source: T0 pin
Falling Edge

// Mode: Normal top=0xFF

// OC0 output: Disconnected

TCCR0=(0<<WGM00) | (0<<COM01)
| (0<<COM00) | (0<<WGM01) |
(0<<CS02) | (0<<CS01) |
(0<<CS00);

TCNT0=0x00;

OCR0=0x00;

// Timer/Counter 1
initialization

// Clock source: System Clock

// Clock value: 46,875 kHz

// Mode: Normal top=0xFFFF

// OC1A output: Disconnected

// OC1B output: Disconnected

// Noise Canceler: Off

// Input Capture on Falling
Edge

// Timer Period: 1 s

// Timer1 Overflow Interrupt:
On

// Input Capture Interrupt:
Off

// Compare A Match Interrupt:
Off

// Compare B Match Interrupt:
Off

TCCR1A=(0<<COM1A1) |
(0<<COM1A0) | (0<<COM1B1) |
(0<<COM1B0) | (0<<WGM11) |
(0<<WGM10);

TCCR1B=(0<<ICNC1) | (0<<ICES1)
| (0<<WGM13) | (0<<WGM12) |
(1<<CS12) | (0<<CS11) |
(0<<CS10);
```

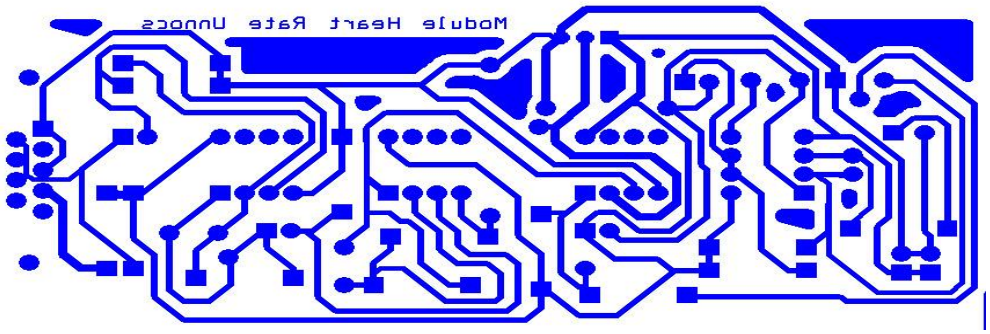
```
TCNT1H=0x48;

TCNT1L=0xE5;

ICR1H=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=0<<AS2;

TCCR2=(0<<WGM20) | (0<<COM21)
| (0<<COM20) | (0<<WGM21) |
(0<<CS22) | (0<<CS21) |
(0<<CS20);

TCNT2=0x00;

OCR2=0x00;



// Timer(s)/Counter(s)
Interrupt(s) initialization

TIMSK=(0<<OCIE2) | (0<<TOIE2)
| (0<<TICIE1) | (0<<OCIE1A) |
(0<<OCIE1B) | (1<<TOIE1) |
(0<<OCIE0) | (0<<TOIE0);

// External Interrupt(s)
initialization

// INT0: Off

// INT1: Off

// INT2: Off

ICR1L=0x00;

OCR1AH=0x00;


MCUCR=(0<<ISC11) | (0<<ISC10)
| (0<<ISC01) | (0<<ISC00);

MCUCSR=(0<<ISC2);


// USART initialization

// USART disabled

UCSRB=(0<<RXCIE) | (0<<TXCIE)
| (0<<UDRIE) | (0<<RXEN) |
(0<<TXEN) | (0<<UCSZ2) |
(0<<RXB8) | (0<<TXB8);

// Analog Comparator
initialization

// Analog Comparator: Off

ACSR=(1<<ACD) | (0<<ACBG) |
(0<<ACO) | (0<<ACI) |
(0<<ACIE) | (0<<ACIC) |
(0<<ACIS1) | (0<<ACIS0);

// ADC initialization

// ADC Clock frequency:
750,000 kHz

// ADC Voltage Reference: AREF
pin

// ADC High Speed Mode: Off

// ADC Auto Trigger Source:
ADC Stopped

ADMUX=ADC_VREF_TYPE;

ADCSRA=(1<<ADEN) | (0<<ADSC) |
(0<<ADATE) | (0<<ADIF) |
(0<<ADIE) | (1<<ADPS2) |
(0<<ADPS1) | (0<<ADPS0);
```

```
SFIOR=(1<<ADHSM) | (0<<ADTS2) | (0<<ADTS1) | (0<<ADTS0);

// SPI initialization

// SPI disabled

SPCR=(0<<SPIE) | (0<<SPE) | (0<<DORD) | (0<<MSTR) | (0<<CPOL) | (0<<CPHA) | (0<<SPR1) | (0<<SPR0);

// TWI initialization

// TWI disabled

TWCR=(0<<TWEA) | (0<<TWSTA) | (0<<TWSTO) | (0<<TWEN) | (0<<TWIE);

// Alphanumeric LCD initialization

// Connections are 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 3

// D5 - PORTC Bit 4

// D6 - PORTC Bit 5

// D7 - PORTC Bit 6

// Characters/line: 16

lcd_init(16);

#asm("sei")

PORTD.0=0;

PORTD.1=0;

PORTD.3=0;

PORTD.4=0;

while (1)

      {

      while(PIND.2==1)

      {

       lcd_gotoxy(0,0);

       lcd_putsf("BPM & SUHU");

       lcd_gotoxy(0,1);

       lcd_putsf("MONITOR ver.01");

       }

       lcd_clear();

      while(1)

       {

       bpm_hitung();

       suhu_hitung();

      }}

}
```

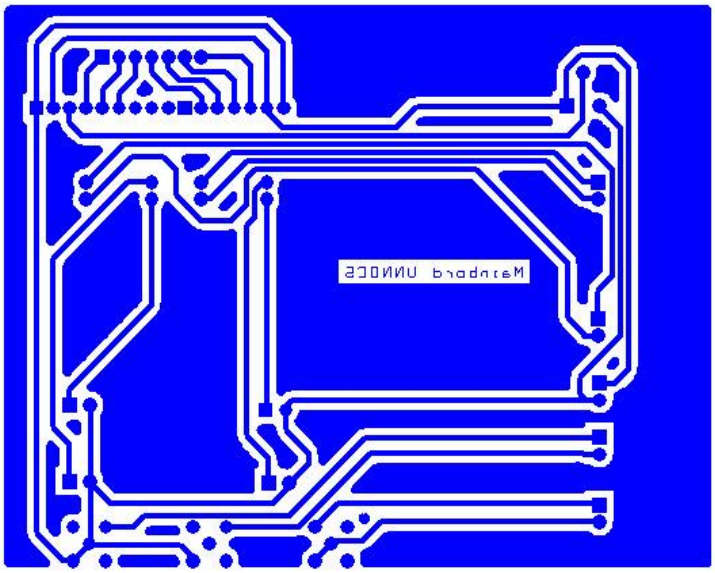
LAMPIRAN 2



Gambar Layout Minsis



Gambar Layout Sensor Heart Rate dan Suhu



Gambar Layout Main Panel