LAMPIRAN

Pembuatan Program *Timer* dan suhu

```
/*******************************************************
This program was created by the

CodeWizardAVR V3.12 Advanced

Automatic Program Generator

© Copyright 1998-2014 Pavel Haiduc, HP InfoTech s.r.l.

http://www.hpinfotech.com



Project :

Version :

Date    : 7/31/2016

Author  :

Company :

Comments:

Chip type               : ATmega16

Program type            : Application
```

```
AVR Core Clock frequency: 1.000000 MHz

Memory model            : Small

External RAM size       : 0

Data Stack size         : 256

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

#include <mega16.h>

#include <delay.h>

#include <stdlib.h>

// Alphanumeric LCD functions

#include <alcd.h>

// Declare your global variables here

bit timerrun=0,timernotend=1;

unsigned char

settingtimer=3,temp[4],detik=59,jam=0,menit=59;
```

```
unsigned int dataadc=0;

float suhu;

// Timer1 overflow interrupt service routine

interrupt [TIM1_OVF] void timer1_ovf_isr(void)

{

// Reinitialize Timer1 value

TCNT1H=0xFC2F >> 8;

TCNT1L=0xFC2F & 0xff;

// Place your code here

if(timerrun==1)

          {

              if(detik==0)

                  {

                  if(menit==0)
```

```
        {

        if(jam==0)

            {

            timerrun=0;timernotend=0;


            }else{

            jam--;menit=59;detik=59;

            }

        }else{

        menit--;detik=59;

        }

    }else{

    detik--;

    }
```

```
            }
        }

// Voltage Reference: AVCC pin

#define ADC_VREF_TYPE ((0<<REFS1) | (1<<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

ADCSRA|=(1<<ADSC);

// Wait for the AD conversion to complete
```

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

ADCSRA|=(1<<ADIF);

return ADCW;

}

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=T

PORTB=(0<<PORTB7) | (0<<PORTB6) | (0<<PORTB5) |

(0<<PORTB4) | (0<<PORTB3) | (0<<PORTB2) | (0<<PORTB1) |

(0<<PORTB0);
```

```
// Port C initialization

// Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=In

Bit2=In Bit1=Out Bit0=Out

DDRC=(0<<DDC7) | (0<<DDC6) | (0<<DDC5) | (0<<DDC4) |

(0<<DDC3) | (0<<DDC2) | (1<<DDC1) | (1<<DDC0);

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

Bit1=0 Bit0=0

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=In Bit3=In

Bit2=In Bit1=In Bit0=In

DDRD=(0<<DDD7) | (0<<DDD6) | (0<<DDD5) | (0<<DDD4) |
```

```
(0<<DDD3) | (0<<DDD2) | (0<<DDD1) | (0<<DDD0);

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

Bit1=P Bit0=P

PORTD=(0<<PORTD7) | (0<<PORTD6) | (0<<PORTD5) |

(0<<PORTD4) | (1<<PORTD3) | (1<<PORTD2) | (1<<PORTD1) |

(1<<PORTD0);

// Timer/Counter 0 initialization

// Clock source: System Clock

// Clock value: Timer 0 Stopped

// 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: 0.977 kHz

// Mode: Normal top=0xFFFF

// OC1A output: Disconnected

// OC1B output: Disconnected

// Noise Canceler: Off

// Input Capture on Falling Edge

// Timer Period: 1.0004 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) | (1<<CS10);

TCNT1H=0xFC;

TCNT1L=0x2F;

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=0<<AS2;

TCCR2=(0<<PWM2) | (0<<COM21) | (0<<COM20) | (0<<CTC2) |

(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

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

// The Analog Comparator's positive input is

// connected to the AIN0 pin

// The Analog Comparator's negative input is

// connected to the AIN1 pin
```

```
ACSR=(1<<ACD) | (0<<ACBG) | (0<<ACO) | (0<<ACI) |

(0<<ACIE) | (0<<ACIC) | (0<<ACIS1) | (0<<ACIS0);

// ADC initialization

// ADC Clock frequency: 500.000 kHz

// ADC Voltage Reference: AVCC pin

// ADC Auto Trigger Source: ADC Stopped

ADMUX=ADC_VREF_TYPE;

ADCSRA=(1<<ADEN) | (0<<ADSC) | (0<<ADATE) | (0<<ADIF) |

(0<<ADIE) | (0<<ADPS2) | (0<<ADPS1) | (1<<ADPS0);

SFIOR=(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 - PORTB Bit 0

// RD - PORTB Bit 1

// EN - PORTB Bit 2

// D4 - PORTB Bit 4

// D5 - PORTB Bit 5

// D6 - PORTB Bit 6
```

```
// D7 - PORTB Bit 7

// Characters/line: 16

lcd_init(16);

// Global enable interrupts

#asm("sei")

lcd_gotoxy(0,0);

lcd_putsf("Welcome");

delay_ms(3000);

while(PIND.3==1)

              {

              if(PIND.0==0)  //Jika PIND.0 ditekan

           {
                  settingtimer=3;delay_ms(500);

                  }else if(PIND.1==0){       //Jika PIND.1 ditekan

                  settingtimer=6;delay_ms(500);
```

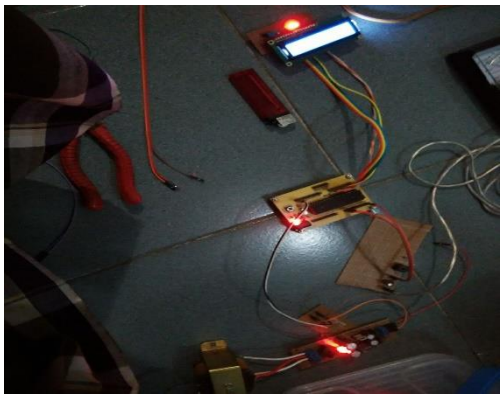
```
                }else if(PIND.2==0){       //Jika PIND.2 ditekan
                settingtimer=9;delay_ms(500);

                }

            lcd_clear();

            itoa(settingtimer,temp);lcd_puts("Setting:

");lcd_puts(temp);lcd_puts(" Jam");      //Tampilkan

Settingan

    delay_ms(200);

    }

delay_ms(500);

jam=settingtimer-1;

timerrun=1;

PORTC.1=1;

          while(timernotend)
```

```
            {

            dataadc=read_adc(0);

            lcd_clear();

            lcd_puts("Timer : ");

if(jam<10){lcd_puts("0");}itoa(jam,temp);lcd_puts(temp);

      lcd_puts(":");

      if(menit<10){lcd_puts("0");}itoa(menit,temp);lcd_puts(te

      mp);lcd_puts(":");

      if(detik<10){lcd_puts("0");}itoa(detik,temp);lcd_puts(te

      mp);

            suhu=dataadc-2;

            suhu=(float)suhu/2;

            lcd_gotoxy(0,1);lcd_puts("Suhu

      :");ftoa(suhu,1,temp);lcd_puts(temp);lcd_putchar(0xdf);l
```

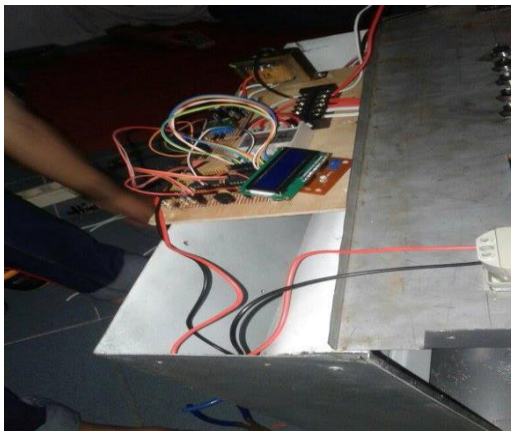
```
cd_puts("C");

    delay_ms(100);

    }

lcd_clear();

lcd_puts("Selesai...");

PORTC.0=1;PORTC.1=0;

while (1)

      {

      // Place your code here

      }

}
```

Proses pemasukan program ke minsis

Pengambilan data tanpa *hourmeter*



Proses pemasangan rangkaian ke *box/chasing*



Proses pengambilan data