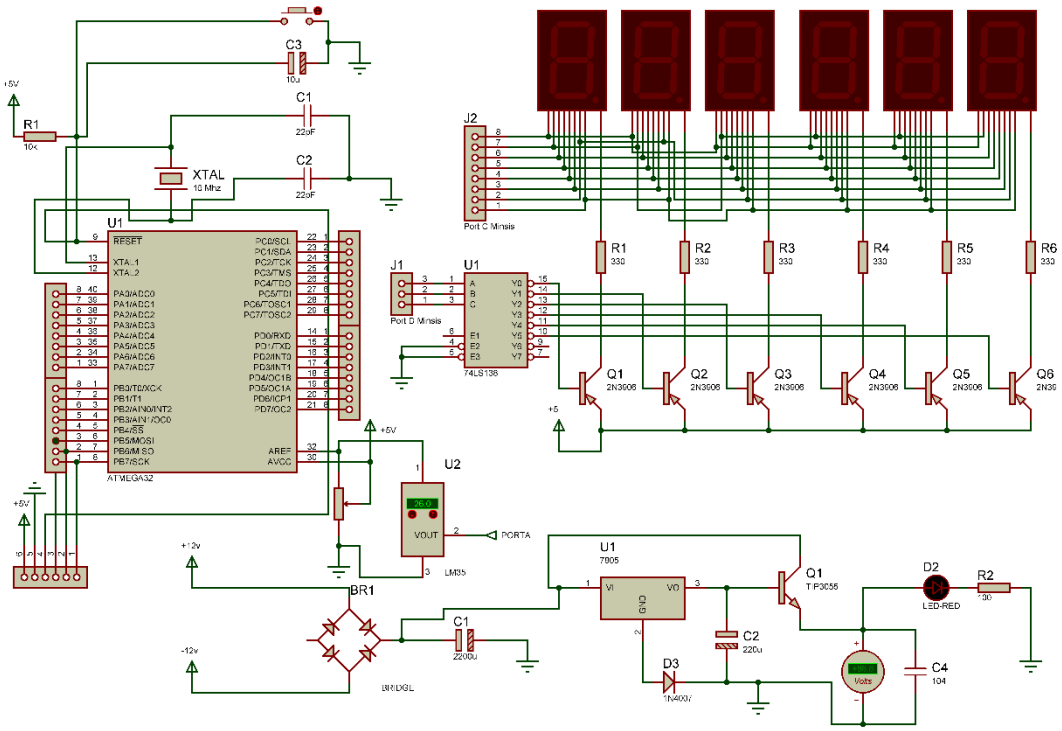
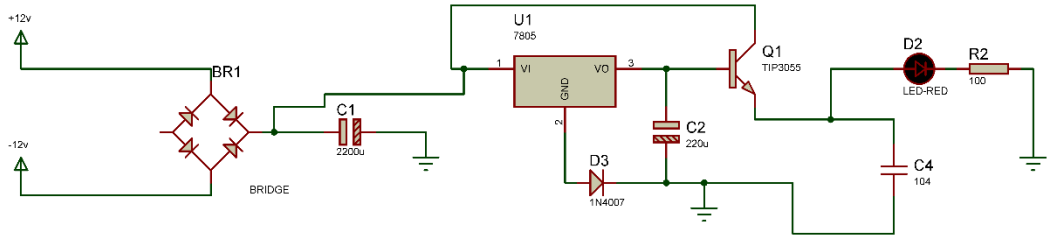
LAMPIRAN

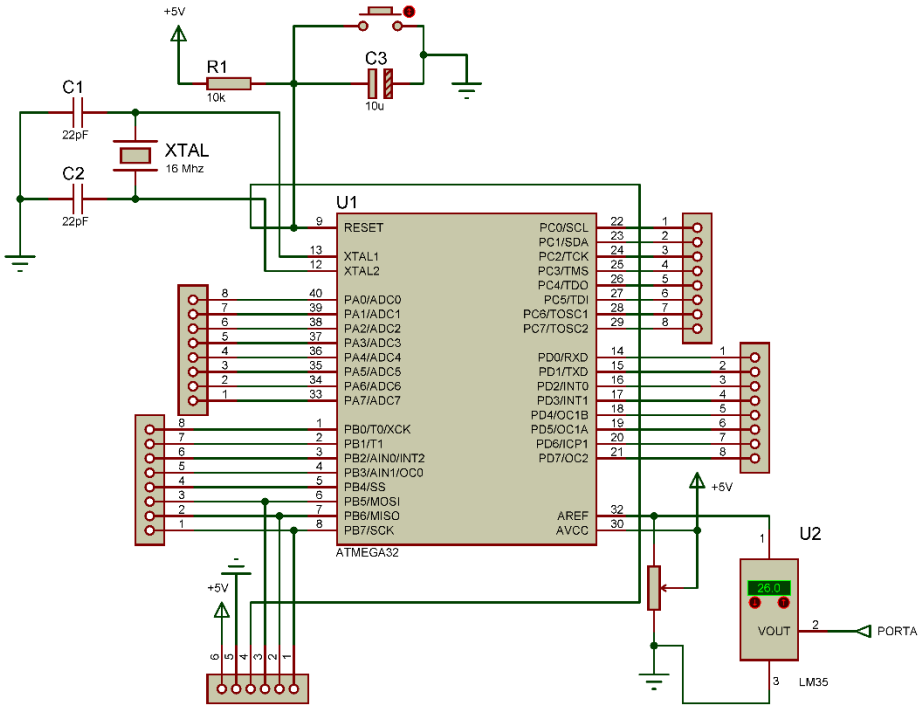
Rangkaian keseluruhan



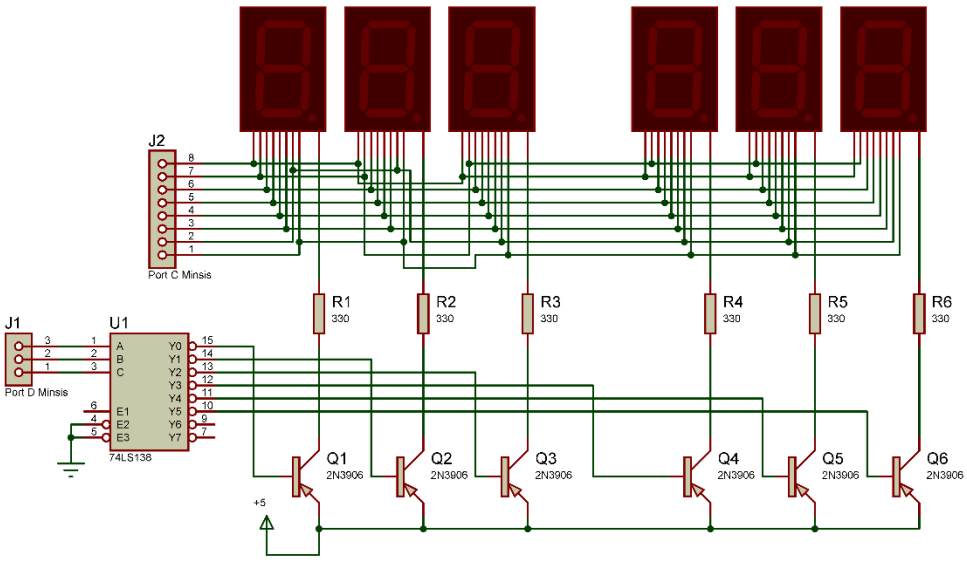
Rangkaian *Power Supply*



Rangkaian *Minimum System*



Rangkaian *Seven segment*



Hasi perhitungan pada pengukuran suhu ruang *infant warmer* 34 ºC

1. **Perhitungan nilai rata-rata**

Display TA

Rata-rata = $\frac{34+33,7+33,7+33,9+34+34+33,9+33,8+33,9+34}{10}$

= 33, 89

Display Fluke

Rata-rata = $\frac{33,8+33,9+33,8+34+33,9+34+34+33,9+33,7+33,8}{10}$

= 33,88

2. **Perhitungan nilai simpangan**

Display TA

Simpangan = 33,88 – 33,89

= -0,01

3. **Perhitungan nilai *error* (%)**

Error = $\frac{-0,01}{33,88} \times 100$

= -0,03 %

SOP

(STANDAR OPERASIOAL PROSEDUR)

1. Siapkan Alat *Infant Warmer*
2. Kunci roda penggerak
3. Hubungkan alat dengan catu daya
4. Hidupkan alat dengan menekan saklar ON/OFF ke posisi ON
5. Tekan tombol pada “control infant” jika ingin menggunakan *system* infant, dan tekan tombol pada “control teraphy” jika ingin menggunakan *system* teraphy
6. Atur suhu yang diinginkan sebelum menggunakan *system* infant
7. Pada saat menggunakan sistem infant, biasakan hidupkan alat 5 – 10 menit sebelum bayi dimasukan agar suhu pada ruang *Infant Warmer* stabil
8. Pada saat menggunakan sistem terapi, pilih mode waktu pemberian sinar phototeraphy dengan menekan tombol 1, 3, dan 6 jam
9. Masukan bayi dan tempelkan suhu sensor suhu *skin* pada bayi
10. Jika proses telah selesai, lepaskan sensor suhu *skin* pada bayi kemudian pindahkan bayi ke bed biasa
11. Tekan saklar ON/OFF ke posisi OFF untuk mematikan alat
12. Buka kunci roda penggerak
13. Rapikan dan simpan alat pada tempatnya
14. SELESAI

Program CVAVR

```
1.    /*****************************************************
2.    This program was produced by the
3.    CodeWizardAVR V2.05.0 Professional
4.    Automatic Program Generator
5.    © Copyright 1998-2010 Pavel Haiduc, HP InfoTech s.r.l.
6.    http://www.hpinfotech.com
7.
8.    Project :
9.    Version :
10.   Date    : 15/07/2018
11.   Author  :
12.   Company :
13.   Comments:
14.
15.
16.   Chip type               : ATmega16
```

```
17.   Program type            : Application
18.   AVR Core Clock frequency: 16,000000 MHz
19.   Memory model            : Small
20.   External RAM size       : 0
21.   Data Stack size         : 256
22.   *****************************************************/
23.
24.   #include <mega16.h>
25.   #include <delay.h>
26.
27.   unsigned char
      angka[10]={0xc0,0xf9,0xa4,0xb0,0x99,0x92,0x82,0xf8,0x80,0x
      90};
28.   int satuan,puluhan,koma,data_temp,x,a,detik,menit,b,jam;
29.   float
      data,suhu,suhu1,suhu2,suhu3,suhu4,suhu5,data2,buff1,buff2;
30.   unsigned char
      detiksat,detikpul,menitsat,menitpul,jamsat,jampul;
31.   bit titik;
32.   int i;
33.
34.   // Timer1 overflow interrupt service routine
35.   interrupt [TIM1_OVF] void timer1_ovf_isr(void)
36.   {
37.   // Reinitialize Timer1 value
38.   TCNT1H=0xBDC >> 8;
39.   TCNT1L=0xBDC & 0xff;
40.   // Place your code here
41.    if (PINA.4==0)
42.    {
43.     if (b==1)
44.     {
45.      detik++;
46.      if (titik==1)
47.      {
48.       PORTA.3=0x00;
49.       titik=0;
50.      }
51.      else
52.      {
53.       PORTA.3=0xff;
54.       titik=1;
55.      }
56.     }
57.
58.    if (b==2)
59.    {
60.     detik++;
61.     if (titik==1)
62.     {
63.      PORTA.3=0x00;
64.      titik=0;
65.     }
66.     else
67.     {
68.      PORTA.3=0xff;
```

```
69.       titik=1;
70.      }
71.     }
72.
73.     if (b==4)
74.     {
75.      detik++;
76.      if (titik==1)
77.      {
78.       PORTA.3=0x00;
79.       titik=0;
80.      }
81.      else
82.      {
83.       PORTA.3=0xff;
84.       titik=1;
85.      }
86.     }
87.    }
88.
89.    if(detik==60)
90.     {
91.      detik=0;
92.      menit++;
93.     }
94.     if(menit==60)
95.     {
96.      menit=0;
97.      jam++;
98.     }
99.     if(jam==24)
100.    {
101.     jam=0;
102.     detik++;
103.    }
104.   }
105.
106.   #define ADC_VREF_TYPE 0x00
107.
108.   // Read the AD conversion result
109.   unsigned int read_adc(unsigned char adc_input)
110.   {
111.   ADMUX=adc_input | (ADC_VREF_TYPE & 0xff);
112.   // Delay needed for the stabilization of the ADC input
       voltage
113.   delay_us(10);
114.   // Start the AD conversion
115.   ADCSRA|=0x40;
116.   // Wait for the AD conversion to complete
117.   while ((ADCSRA & 0x10)==0);
118.   ADCSRA|=0x10;
119.   return ADCW;
120.   }
121.
122.   // Declare your global variables here
123.   void ambil_data()
```

```
{
suhu1=suhu*10;
koma=(int)suhu1;
koma=koma%10;
data_temp=(int)suhu;
satuan=data_temp%10;
puluhan=(data_temp/10)%10;
}

void ambil_data2()
{
suhu3=suhu2*10;
koma=(int)suhu3;
koma=koma%10;
data_temp=(int)suhu2;
satuan=data_temp%10;
puluhan=(data_temp/10)%10;
}

void ambil_data3()
{
suhu4=suhu5*10;
koma=(int)suhu4;
koma=koma%10;
data_temp=(int)suhu5;
satuan=data_temp%10;
puluhan=(data_temp/10)%10;
}

void ambil_data4()
{
    detiksat=detik%10;
    detikpul=(detik/10)%10;
}
void ambil_data5()
{
    menitsat=menit%10;
    menitpul=(menit/10)%10;
}
void ambil_data6()
{
    jamsat=jam%10;
    jampul=(jam/10)%10;
}

void tampilkan_seven_segment()
{
PORTD.2=1;PORTD.3=0;PORTD.4=0;
 PORTC=angka[satuan];
PORTC.7=0;
delay_ms(1);

PORTD.2=0;PORTD.3=0;PORTD.4=0;
PORTC=angka[puluhan];
delay_ms(1);

```

```
180.  PORTD.2=0;PORTD.3=1;PORTD.4=0;
181.  PORTC=angka[koma];
182.  delay_ms(1);
183.
184.  }
185.
186.  void tampilkan_seven_segment2()
187.  {
188.  PORTD.2=0;PORTD.3=0;PORTD.4=1;
189.  PORTC=angka[satuan];
190.  PORTC.7=0;
191.  delay_ms(1);
192.
193.  PORTD.2=1;PORTD.3=1;PORTD.4=0;
194.  PORTC=angka[puluhan];
195.  delay_ms(1);
196.
197.  PORTD.2=1;PORTD.3=0;PORTD.4=1;
198.  PORTC=angka[koma];
199.  delay_ms(1);
200.
201.  PORTD.2=1;PORTD.3=1;PORTD.4=1;
202.  PORTC=angka[koma];
203.  delay_ms(1);
204.  }
205.
206.  void tampilkan_seven_segment3()
207.  {
208.  PORTD.5=0;PORTD.6=0;PORTD.7=0;
209.  PORTC=angka[puluhan];
210.  delay_ms(1);
211.
212.  PORTD.5=0;PORTD.6=0;PORTD.7=1;
213.  PORTC=angka[satuan];
214.  PORTC.7=0;
215.  delay_ms(1);
216.
217.  PORTD.5=0;PORTD.6=1;PORTD.7=0;
218.  PORTC=angka[koma];
219.  delay_ms(1);
220.
221.  PORTD.5=1;PORTD.6=1;PORTD.7=1;
222.  PORTC=0b10100100;
223.  delay_ms(1);
224.  }
225.
226.  void tampil4()
227.  {
228.      PORTD.5=1;
229.      PORTD.6=0;
230.      PORTD.7=0;
231.      PORTC=angka[menitsat];
232.      delay_ms(1);
233.
234.      PORTD.5=0;
235.      PORTD.6=1;
```

```
236.      PORTD.7=1;
237.      PORTC=angka[menitpul];
238.      delay_ms(1);
239.  }
240.  void tampil5()
241.  {
242.      PORTD.5=1;
243.      PORTD.6=1;
244.      PORTD.7=0;
245.      PORTC=angka[jamsat];
246.      delay_ms(1);
247.
248.      PORTD.5=1;
249.      PORTD.6=0;
250.      PORTD.7=1;
251.      PORTC=angka[jampul];
252.      delay_ms(1);
253.
254.      PORTD.5=1;
255.      PORTD.6=1;
256.      PORTD.7=1;
257.      PORTC=angka[jampul];
258.      delay_ms(1);
259.      }
260.
261.  void main(void)
262.  {
263.  // Declare your local variables here
264.  PORTA.1=0;
265.  PORTA.2=0;
266.  PORTD.1=0;
267.  PORTD.2=1;
268.  PORTD.3=1;
269.  PORTD.4=1;
270.  PORTD.5=1;
271.  PORTD.6=1;
272.  PORTD.7=1;
273.  suhu2=34;
274.  // Input/Output Ports initialization
275.  // Port A initialization
276.  // Func7=In Func6=In Func5=Out Func4=Out Func3=Out
      Func2=Out Func1=Out Func0=Out
277.  // State7=T State6=T State5=0 State4=0 State3=0 State2=0
      State1=0 State0=0
278.  PORTA=0x30;
279.  DDRA=0x0F;
280.
281.  // Port B initialization
282.  // Func7=In Func6=Out Func5=Out Func4=Out Func3=In
      Func2=In Func1=In Func0=In
283.  // State7=P State6=1 State5=1 State4=1 State3=P State2=P
      State1=P State0=P
284.  PORTB=0xFF;
285.  DDRB=0x70;
286.
287.  // Port C initialization
```

```
288.  // Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out
      Func2=Out Func1=Out Func0=Out
289.  // State7=0 State6=0 State5=0 State4=0 State3=0 State2=0
      State1=0 State0=0
290.  PORTC=0xFF;
291.  DDRC=0xFF;
292.
293.  // Port D initialization
294.  // Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out
      Func2=Out Func1=Out Func0=Out
295.  // State7=0 State6=0 State5=0 State4=0 State3=0 State2=0
      State1=0 State0=0
296.  DDRD=0xFF;
297.
298.  // Timer/Counter 0 initialization
299.  // Clock source: System Clock
300.  // Clock value: Timer 0 Stopped
301.  // Mode: Normal top=0xFF
302.  // OC0 output: Disconnected
303.  TCCR0=0x00;
304.  TCNT0=0x00;
305.  OCR0=0x00;
306.
307.  // Timer/Counter 1 initialization
308.  // Clock source: System Clock
309.  // Clock value: 62,500 kHz
310.  // Mode: Normal top=0xFFFF
311.  // OC1A output: Discon.
312.  // OC1B output: Discon.
313.  // Noise Canceler: Off
314.  // Input Capture on Falling Edge
315.  // Timer1 Overflow Interrupt: On
316.  // Input Capture Interrupt: Off
317.  // Compare A Match Interrupt: Off
318.  // Compare B Match Interrupt: Off
319.  TCCR1A=0x00;
320.  TCCR1B=0x04;
321.  TCNT1H=0x0B;
322.  TCNT1L=0xDC;
323.  ICR1H=0x00;
324.  ICR1L=0x00;
325.  OCR1AH=0x00;
326.  OCR1AL=0x00;
327.  OCR1BH=0x00;
328.  OCR1BL=0x00;
329.
330.  // Timer/Counter 2 initialization
331.  // Clock source: System Clock
332.  // Clock value: Timer2 Stopped
333.  // Mode: Normal top=0xFF
334.  // OC2 output: Disconnected
335.  ASSR=0x00;
336.  TCCR2=0x00;
337.  TCNT2=0x00;
338.  OCR2=0x00;
339.
```

```
340.  // External Interrupt(s) initialization
341.  // INT0: Off
342.  // INT1: Off
343.  // INT2: Off
344.  MCUCR=0x00;
345.  MCUCSR=0x00;
346.
347.  // Timer(s)/Counter(s) Interrupt(s) initialization
348.  TIMSK=0x04;
349.
350.  // USART initialization
351.  // USART disabled
352.  UCSRB=0x00;
353.
354.  // Analog Comparator initialization
355.  // Analog Comparator: Off
356.  // Analog Comparator Input Capture by Timer/Counter 1: Off
357.  ACSR=0x80;
358.  SFIOR=0x00;
359.
360.  // ADC initialization
361.  // ADC Clock frequency: 500,000 kHz
362.  // ADC Voltage Reference: AREF pin
363.  // ADC Auto Trigger Source: ADC Stopped
364.  ADMUX=ADC_VREF_TYPE & 0xff;
365.  ADCSRA=0x85;
366.
367.  // SPI initialization
368.  // SPI disabled
369.  SPCR=0x00;
370.
371.  // TWI initialization
372.  // TWI disabled
373.  TWCR=0x00;
374.
375.  // Global enable interrupts
376.  #asm("sei")
377.
378.  while (1)
379.   {
380.    if (PINA.4==0)
381.    {
382.    buff1=0;
383.    for(i=0;i<10;i++)
384.    {
385.     data=read_adc(6);
386.     buff1+=data;
387.    }
388.    suhu5=(buff1/10)*121/1023;
389.
390.     if (a==1)
391.     {
392.     buff2=0;
393.     for(i=0;i<10;i++)
394.     {
395.      data2=read_adc(7);
```

```
396.       buff2+=data2;
397.      }
398.      suhu=(buff2/10)*121/1023-3;
399.
400.       PORTA.1=1;
401.
402.       if (suhu>=suhu2)
403.       {
404.        PORTA.1=0;
405.       }
406.
407.       if (suhu<suhu2)
408.       {
409.        PORTA.1=1;
410.       }
411.      }
412.
413.      if (suhu>=suhu2+0.5)
414.      {
415.       PORTD.1=1;
416.       suhu=0;
417.       PORTA.1=0;
418.       suhu2=0;
419.      }
420.
421.      if (a==2)
422.      {
423.       suhu=0;
424.       PORTA.1=0;
425.       PORTD.1=0;
426.      }
427.     }
428.
429.     if (PINA.4==1)
430.     {
431.      suhu=0;
432.      PORTA.1=0;
433.      a=0;
434.      suhu2=34;
435.
436.        b=0;
437.        PORTD.1=0;
438.        detik=0;
439.        menit=0;
440.        jam=0;
441.        PORTA.2=0;
442.        PORTA.3=0x00;
443.        PORTB.4=0xFF;
444.        PORTB.5=0xFF;
445.        PORTB.6=0xFF;
446.     }
447.
448.     for(x=0;x<100;x++)
449.     {
450.      if (PINA.4==0)
451.      {
```

```
452.       if (PINB.1==0) {a=1;}
453.       if (PINB.0==0) {a=2;}
454.       if (PINB.4==0) {b=1;}
455.       if (PINB.5==0) {b=2;}
456.       if (PINB.6==0) {b=4;}
457.
458.       if (PINB.3==0)
459.       {
460.        suhu2=suhu2+0.01;
461.        if (suhu2>37.6)
462.        {
463.         suhu2=34;
464.        }
465.       }
466.
467.       if (PINB.2==0)
468.       {
469.        suhu2=suhu2-0.01;
470.        if (suhu2<34)
471.        {
472.         suhu2=37.5;
473.        }
474.       }
475.
476.       if (PINB.7==0)
477.       {
478.        b=0;
479.        PORTD.1=0;
480.        detik=0;
481.        menit=0;
482.        jam=0;
483.        PORTA.2=0;
484.        PORTA.3=0x00;
485.        PORTB.4=0xFF;
486.        PORTB.5=0xFF;
487.        PORTB.6=0xFF;
488.       }
489.
490.        if (b==1)
491.        {
492.         PORTA.2=1;
493.         PORTB.4=0x00;
494.         PORTB.5=0XFF;
495.         PORTB.6=0XFF;
496.
497.         if (jam==1)
498.         {
499.          jam=1;
500.          detik=0;
501.          PORTA.3=0x00;
502.          PORTA.2=0;
503.          PORTD.1=1;
504.         }
505.        }
506.
507.       if (b==2)
```

```
508.       {
509.        PORTB.5=0x00;
510.        PORTB.6=0XFF;
511.        PORTB.4=0XFF;
512.        PORTA.2=1;
513.
514.        if (jam==3)
515.        {
516.         menit=0;
517.         detik=0;
518.         jam=3;
519.         PORTA.3=0x00;
520.         PORTA.2=0;
521.         PORTD.1=1;
522.        }
523.       }
524.
525.       if (b==4)
526.       {
527.        PORTB.6=0x00;
528.        PORTB.4=0XFF;
529.        PORTB.5=0XFF;
530.        PORTA.2=1;
531.
532.        if (jam==6)
533.        {
534.         menit=0;
535.         detik=0;
536.         jam=6;
537.         PORTA.3=0x00;
538.         PORTA.2=0;
539.         PORTD.1=1;
540.        }
541.       }
542.
543.     ambil_data();
544.     tampilkan_seven_segment();
545.     ambil_data2();
546.     tampilkan_seven_segment2();
547.     ambil_data3();
548.     tampilkan_seven_segment3();
549.     ambil_data4();
550.     tampil4();
551.     ambil_data5();
552.     tampil5();
553.     ambil_data6();
554.      }
555.     }
556.    }
557.   }
```