LAMPIRAN

- Listing Program pada mikrokontroler (device_program.c)

```
/*****************************************************
This program was produced by the
CodeWizardAVR V2.05.3 Standard
Automatic Program Generator
© Copyright 1998-2011 Pavel Haiduc, HP InfoTech s.r.l.
http://www.hpinfotech.com

Project :
Version :
Date    : 4/30/2015
Author  :
Company : Widya Mandala Catholic University
Comments:


Chip type               : ATmega164PA
Program type            : Application
AVR Core Clock frequency: 18.432000 MHz
Memory model            : Small
External RAM size       : 0
Data Stack size         : 256
*****************************************************/

#include <mega164a.h>
#include <delay.h>
#include <string.h>
#include <stdlib.h>

#define GPGGA          1
#define GPRMC          6
#define GPVTG          7

//#define relay1          PORTD.5
//#define relay2          PORTD.6
//#define charger         PORTD.7
//#define EX_batt_stat    PINB.1
//#define bt_emergency    PINB.6
//#define LED_STAT        PORTB.0
//#define StatusSIM908    PINC.3
// yang terbaru
#define relay1          PORTD.5
#define relay2          PORTD.6
//#define charger         PORTC.4     //PORTB.4

#define EX_batt_stat    PINC.3  // CEK STATUS ADA AKI ATAU
TIDAK
//#define bt_emergency    PINB.6
#define LED_STAT        PORTD.7  // d 7
```

```
#define StatusSIM908    PINB.0
//#define POWEREN         PORTB.4  // Portb.4
#define PWRKEY_P        PORTB.1

#define OK      1
#define ERROR   0

#define ON      1
#define OFF     0

// Declare your global variables here

unsigned char sCommand,header_ID,;
unsigned int
selection=0,validate,header,count=0,net_stat=1,charger;
unsigned char EXTBAT[1],INTBATCAP[4],mode;
bit Flag_restart=0;
flash unsigned char ID[6]="DV002";

#ifndef RXB8
#define RXB8 1
#endif

#ifndef TXB8
#define TXB8 0
#endif

#ifndef UPE
#define UPE 2
#endif

#ifndef DOR
#define DOR 3
#endif

#ifndef FE
#define FE 4
#endif

#ifndef UDRE
#define UDRE 5
#endif

#ifndef RXC
#define RXC 7
#endif

#define FRAMING_ERROR (1<<FE)
#define PARITY_ERROR (1<<UPE)
#define DATA_OVERRUN (1<<DOR)
#define DATA_REGISTER_EMPTY (1<<UDRE)
#define RX_COMPLETE (1<<RXC)
```

```
// USART0 Receiver buffer
#define RX_BUFFER_SIZE0 127
char rx_buffer0[RX_BUFFER_SIZE0];

#if RX_BUFFER_SIZE0 <= 256
unsigned char rx_wr_index0,rx_rd_index0,rx_counter0;
#else
unsigned int rx_wr_index0,rx_rd_index0,rx_counter0;
#endif

// This flag is set on USART0 Receiver buffer overflow
bit rx_buffer_overflow0;

// USART0 Receiver interrupt service routine
interrupt [USART0_RXC] void usart0_rx_isr(void)
{
char status,data;
status=UCSR0A;
data=UDR0;
if ((status & (FRAMING_ERROR | PARITY_ERROR | DATA_OVERRUN))==0)
   {
   rx_buffer0[rx_wr_index0++]=data;
#if RX_BUFFER_SIZE0 == 256
   // special case for receiver buffer size=256
   if (++rx_counter0 == 0) rx_buffer_overflow0=1;
#else
   if (rx_wr_index0 == RX_BUFFER_SIZE0) rx_wr_index0=0;
   if (++rx_counter0 == RX_BUFFER_SIZE0)
      {
      rx_counter0=0;
      rx_buffer_overflow0=1;
      }
#endif
   }
   //#asm("wdr")
}

// Timer2 overflow interrupt service routine
bit FLAG_TOV=0;
unsigned int timer_count;
unsigned int terhold_count=0;
unsigned int timeout_ms=0;
interrupt [TIM2_OVF] void timer2_ovf_isr(void)
{
        // Place your code here
        timeout_ms++;
        if (timeout_ms>=100)
        {
            timer_count++;
            timeout_ms=0;
            //PORTB.0=~PORTB.0;
```

```
        }
        if (timer_count>=terhold_count)
        {
            TCCR2B=0x00;
            FLAG_TOV=1;
            //PORTB.0=0;
        }
        else
        {
            TCNT2=0x4B;
        }
}
void start_timer2(unsigned int sec)
{
    terhold_count=sec;
    TCCR2B=0x07;
    TCNT2=0x4B;
    FLAG_TOV=0;
    timer_count=0;

}
void clear_timer()
{
    TCCR2B=0x00;
    TCNT2=0x00;
    FLAG_TOV=0;
    timer_count=0;
}
#ifndef _DEBUG_TERMINAL_IO_
// Get a character from the USART0 Receiver buffer
#define _ALTERNATE_GETCHAR_
#pragma used+

char getchar(void)
{
    char data;
    ////////////////////////start timer/////////////////
    start_timer2(10);
    while ((rx_counter0==0) && (FLAG_TOV==0));
    if (FLAG_TOV==1)
    {
        return (0);
    }
    else
    {
        clear_timer();
    }
data=rx_buffer0[rx_rd_index0++];
#if RX_BUFFER_SIZE0 != 256
if (rx_rd_index0 == RX_BUFFER_SIZE0) rx_rd_index0=0;
#endif
#asm("cli")
```

```
--rx_counter0;
#asm("sei")
return data;
}
#pragma used-
#endif

// USART0 Transmitter buffer
#define TX_BUFFER_SIZE0 64
char tx_buffer0[TX_BUFFER_SIZE0];

#if TX_BUFFER_SIZE0 <= 256
unsigned char tx_wr_index0,tx_rd_index0,tx_counter0;
#else
unsigned int tx_wr_index0,tx_rd_index0,tx_counter0;
#endif

// USART0 Transmitter interrupt service routine
interrupt [USART0_TXC] void usart0_tx_isr(void)
{
if (tx_counter0)
   {
   --tx_counter0;
   UDR0=tx_buffer0[tx_rd_index0++];
#if TX_BUFFER_SIZE0 != 256
   if (tx_rd_index0 == TX_BUFFER_SIZE0) tx_rd_index0=0;
#endif
   }
   //#asm("wdr")
}

#ifndef _DEBUG_TERMINAL_IO_
// Write a character to the USART0 Transmitter buffer
#define _ALTERNATE_PUTCHAR_
#pragma used+
void putchar(char c)
{
while (tx_counter0 == TX_BUFFER_SIZE0);
#asm("cli")
if (tx_counter0 || ((UCSR0A & DATA_REGISTER_EMPTY)==0))
   {
   tx_buffer0[tx_wr_index0++]=c;
#if TX_BUFFER_SIZE0 != 256
   if (tx_wr_index0 == TX_BUFFER_SIZE0) tx_wr_index0=0;
#endif
   ++tx_counter0;
   }
else
   UDR0=c;
#asm("sei")
}
#pragma used-
```

```
#endif

// USART1 Receiver buffer
#define RX_BUFFER_SIZE1 128
char rx_buffer1[RX_BUFFER_SIZE1];

#if RX_BUFFER_SIZE1 <= 256
unsigned char rx_wr_index1,rx_rd_index1,rx_counter1;
#else
unsigned int rx_wr_index1,rx_rd_index1,rx_counter1;
#endif

// This flag is set on USART1 Receiver buffer overflow
bit rx_buffer_overflow1;

// USART1 Receiver interrupt service routine
interrupt [USART1_RXC] void usart1_rx_isr(void)
{
char status,data;
status=UCSR1A;
data=UDR1;
if ((status & (FRAMING_ERROR | PARITY_ERROR | DATA_OVERRUN))==0)
   {
   rx_buffer1[rx_wr_index1++]=data;
#if RX_BUFFER_SIZE1 == 256
   // special case for receiver buffer size=256
   if (++rx_counter1 == 0) rx_buffer_overflow1=1;
#else
   if (rx_wr_index1 == RX_BUFFER_SIZE1) rx_wr_index1=0;
   if (++rx_counter1 == RX_BUFFER_SIZE1)
      {
      rx_counter1=0;
      rx_buffer_overflow1=1;
      }
#endif
   }
}

// Get a character from the USART1 Receiver buffer
#pragma used+
char getchar1(void)
{
char data;
while (rx_counter1==0);
data=rx_buffer1[rx_rd_index1++];
#if RX_BUFFER_SIZE1 != 256
if (rx_rd_index1 == RX_BUFFER_SIZE1) rx_rd_index1=0;
#endif
#asm("cli")
--rx_counter1;
#asm("sei")
return data;
```

```
}
#pragma used-
// Standard Input/Output functions
#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;
}
unsigned int led,thigh,ton;
// Timer 0 overflow interrupt service routine
interrupt [TIM0_OVF] void timer0_ovf_isr(void)
{
// Place your code here
        led++;
        if (led>=ton )
        {
                //PORTB.0=1;
                LED_STAT=0;
        }
        else {
        //PORTB.0=0;
        LED_STAT=1;
        }
        if (led>=(ton+thigh))
        {
                led=0;
        }
}

// Pin change 8-15 interrupt service routine
bit PINEMG=0;
bit flag_tbut_run=0;
bit Busy=1;
unsigned char BTEMGSTAT='0';
unsigned char count_bt=0;
interrupt [PC_INT1] void pin_change_isr1(void)
{
// Place your code here
    if (flag_tbut_run==0)
    {
```

```
        TCCR1B=0x05;
        TCNT1H=0xB1;
        TCNT1L=0xDF;
        flag_tbut_run=1;
    }
    if (PINEMG==0)
    {
        PINEMG=1;
        count_bt++;
        delay_ms(100);
    }
    else
    {
        PINEMG=0;
    }
}
// External Interrupt 2 service routine
interrupt [EXT_INT2] void ext_int2_isr(void)
{
// Place your code here

}
// Timer1 overflow interrupt service routine
void  get_server_command(void);
interrupt [TIM1_OVF] void timer1_ovf_isr(void)
{
// Place your code here
    TCCR1B=0x00;
    TCNT1H=0x00;
    TCNT1L=0x00;
    if (count_bt>2)
    {
       BTEMGSTAT='1';
       count_bt=0;
    }
    else
    {
        count_bt=0;
    }
    TCCR1B=0x00;
    TCNT1H=0x00;
    TCNT1L=0x00;
    //PORTB.0=1;
    LED_STAT=0;
    delay_ms(50);
    //PORTB.0=0;
    LED_STAT=1;
    flag_tbut_run=0;
}

#define MAX_QUEUE_SIZE 2
struct GPS
```

```
{
    unsigned char
UTC[12],LAT[11],LNG[12],NS[2],EW[2],GPSU[3],SOG[5];
};
typedef struct GPS GPS_data;
GPS_data NMEA;

void initialize_Queue( int* front, int* length )
{
  *front = *length = 0;
}
int queuefull( int length )/* Check queue full */
{
  return length == MAX_QUEUE_SIZE;
}
int queueempty( int length )
{
  return length == 0;
}
void enqueue( int front, int* length, GPS_data queue[],GPS_data
item )
{
      int where;
      if ( !queuefull( *length ) )
      {
           where = front + *length;
           queue[ where % MAX_QUEUE_SIZE ] = item;
           (*length)++;
      }
}

void led_blink(unsigned char control, unsigned int
pulse,unsigned int T_high)
{
    ton=pulse;
    thigh=T_high;
    if (control==ON)
    {
        TCCR0B=0x05;
    }
    else if (control==OFF)
    {
        TCCR0B=0x00;
    }
}



char wait_string(char *stringof)
{
//////////////////////////// menunggu string hingga
timeout//////////////////////////////
```

```
        char tmp;
        unsigned int indx=0;
        while(indx<strlen(stringof))
        {
                tmp=getchar();
                if (tmp!=0)
                {
                        if (tmp == stringof[indx])
                        {
                                indx++;
                        }
                        else
                        {
                                indx=0;
                        }
                }
                else
                {
                    return (0);
                }
        }
        return (1);
}
void send_string(unsigned char *strofarr)
{
        unsigned int i=0;
        unsigned int arrlength=strlen(strofarr);
        for (i=0; i< arrlength; i++)
        {
                putchar(strofarr[i]);
        }
}
void clear_serial()
{
    unsigned char recycle;
    while(rx_counter0>0)
    {
       recycle=getchar();
    }
}

void batt_management(int secT)
{
        //dec 905 >>> 8.4V Vrdif 4.44/4.43 vref = 5.01
//430>>880 // full
        //6.6 >> 3.5883V >> dec 732 >> v habis
        //drop
        //5.59v
        //unsigned long ADC_bat=0;

        //float bat_cap=0;
        //charger=0;
```

```
if (secT>=40)
{
    printf("AT+CBC");
    putchar(13);
if (wait_string("+CBC"))
{
    wait_string(": ");
    mode=getchar();
    wait_string(",");
    INTBATCAP[0]=getchar();
    INTBATCAP[1]=getchar();
    INTBATCAP[2]=getchar();
}
putchar(13);
putchar(13);
wait_string("OK");

    if (EX_batt_stat==0)
    {
        EXTBAT[0]='0';
        charger=0;
    }
    else
    {
        EXTBAT[0]='1';
        if (mode==1)
        {
            charger=1;
        }
        else if (mode==0)
        {
            charger=0;
        }
    }
    printf("mode :");
    putchar(mode);
    printf("kapasitas : ");
    putchar(INTBATCAP[0]);
    putchar(INTBATCAP[1]);
    if (INTBATCAP[2]!=',')
    {
            putchar(INTBATCAP[2]);
    }
    putchar(13);

    printf("exbat: ");
    putchar(EXTBAT[0]);
    printf(" charging: ");
    putchar(charger+0x30);
}
else if (secT==36)
{
```

```
            if (charger)
            {
            charger=0;
            }
        }
}

void initGPRS()
{
        delay_ms(1000);
        printf("AT+CGDCONT=1,");
        putchar('"');
        printf("IP");
        putchar('"');
        putchar(',');
        putchar('"');
        printf("internet");
        putchar('"');
        putchar(13);
        wait_string("OK");
        delay_ms(5000);
        printf("AT+CSTT=");
        putchar('"');
        printf("internet");
        putchar('"');
        putchar(',');
        putchar('"');
        printf("");
        putchar('"');
        putchar(',');
        putchar('"');
        printf("");
        putchar('"');
        putchar(13);
        while(getchar()!='O') ;
}

char start_GPS()
{
        printf("AT+CGPSIPR=115200");
        putchar(13);
        wait_string("OK");
        printf("AT+CGPSPWR=1");
        delay_ms(1000);
        putchar(13);
        wait_string("OK");
        printf("AT+CGPSRST=1");
        delay_ms(1000);
        putchar(13);
        wait_string("OK");
        delay_ms(4000);
        if (rx_counter1>0)
```

```
        {
            return  (1);
        }
        else
        {
            return (0);
        }
}

char start_SIM908()
{
        LED_STAT=0;
        if(StatusSIM908==1)
        {
            LED_STAT=1;
            return (7);
        }
        else
        {
            //PORTD.4=0;
            PWRKEY_P=1;
            LED_STAT=1;
            delay_ms(2100);
            //PORTD.4=1;
            //TERBARU
            PWRKEY_P=0;
            LED_STAT=0;
            delay_ms(20000);
            putchar(13);

            if (!start_GPS())
            {
                putchar(13);
                send_string("GPS NOT CONNECT");
                putchar(13);
                LED_STAT=1;
                return (6);
            }
            else
            {
                LED_STAT=1;
                return (7);
            }
        }
}

void led_blinking(int value)
{
    unsigned int i=0;
    if (value>0)
    {
        for (i=0;i<value;i++)
```

```
        {
        LED_STAT=0;
        delay_ms(100);
        LED_STAT=1;
        delay_ms(400);
        }
        delay_ms(2000);
    }
    else
    {
        LED_STAT=0;
        delay_ms(100);
        LED_STAT=1;
        delay_ms(100);
    }
}
void TurnOff_SIM908()
{
    char stat_off=0;
    while(stat_off==0)
    {
        clear_serial();
        if(StatusSIM908==1)
        {

            //PORTD.4=0;
            //Terbaru
            PWRKEY_P=0;
            delay_ms(2000);
            //PORTD.4=1;
            PWRKEY_P=1;
            delay_ms(500);
            if(wait_string("NORMAL POWER DOWN"))
            {
               stat_off=1;
            }
        }
        else
        {
          stat_off=1;
        }
    }
}

void booting_SIM908()
{
    char status_boot=0;
    int kt=0;
    while(1)
    {
        putchar(13);
        status_boot=start_SIM908();
```

```
        if (status_boot>1)
        {
            while((status_boot!=7)&&(kt<10))
            {
                led_blinking(status_boot);
                kt++;
            }
        }
        else
        {
            led_blink(ON,20,2);
            delay_ms(20000);
            led_blink(OFF,40,2);

        }
        if (status_boot!=7)
        {
            TurnOff_SIM908();
            status_boot=0;
            kt=0;
        }
        else
        {
            break;
        }
    }
}

void restart_SIM908()
{
    /// turn off modul SIM908////
    TurnOff_SIM908();
    delay_ms(2000);
    booting_SIM908();
    initGPRS();
    //PORTB.0=0;
    LED_STAT=1;
    Busy=0;
}

void cek_header()
{
        if ((sCommand=='$') && (validate==0))
        {
                validate=1;
                selection=0;
        }
        else if ((sCommand=='G') && (validate==1))
        {
                validate=2;
        }
        else if ((sCommand=='P') && (validate==2))
```

```
        {
                validate=3;
        }
        else if ((sCommand=='G') && (validate==3))
        {
                validate=4;
        }
        else if ((sCommand=='G') && (validate==4))
        {
                validate=5;
        }
        else if ((sCommand=='A') && (validate==5))
        {
                header=1;
                header_ID=GPGGA;
                validate=0;
        }
        else if ((sCommand=='V') && (validate==3))
        {
                validate=4;
        }
        else if ((sCommand=='T') && (validate==4))
        {
                validate=5;
        }
        else if ((sCommand=='G') && (validate==5))
        {
                header=1;
                header_ID=GPVTG;
                validate=0;
        }
        else if ((sCommand=='R') && (validate==3))
        {
                validate=4;
        }
        else if ((sCommand=='M') && (validate==4))
        {
                validate=5;
        }
        else if ((sCommand=='C') && (validate==5))
        {
                header=1;
                header_ID=GPRMC;
                validate=0;
        }
        else
        {
                header=0;
                validate=0;
        }
}
```

```
void ambil_data_GPVTG(int select)
{
        if (select==7)
        {
                NMEA.SOG[count]=sCommand;
        }
}
void ambil_data_GPGGA(int select)
{
        if (select==1)
        {
                NMEA.UTC[count]=sCommand;
        }
        else if (select==2)
        {
                NMEA.LAT[count]=sCommand;
        }
        else if (select==3)
        {
                NMEA.NS[count]=sCommand;
        }
        else if (select==4)
        {
                NMEA.LNG[count]=sCommand;
        }
        else if (select==5)
        {
                NMEA.EW[count]=sCommand;
        }
        else if (select==6)
        {
                NMEA.GPSU[0]='';
                NMEA.GPSU[1]='';
                NMEA.GPSU[2]='';
        }
        else if (select==7)
        {
                NMEA.GPSU[count]=sCommand;
        }
}
void cekSerial(void)
{
        if (rx_counter1>0)
        {       //ada data serial masuk
                sCommand = getchar1();
                if (header==0)
                {
                        cek_header();
                }
                else if((header==1) && (sCommand ==','))
                {
                        count=0;
```

```
                        selection++;
                }
                else if ((selection>0) && (header==1))
                {
                       if (header_ID==GPGGA)
                       {
                                ambil_data_GPGGA(selection);
                       }
                       else if (header_ID==GPVTG)
                       {
                                ambil_data_GPVTG(selection);
                       }
                       count++;
                }
                if ((sCommand==13) || (sCommand==10))
                {
                        selection=0;
                        header=0;
                        header_ID='';
                        //putchar (13);
                }
                //<CR>: ASCII character 13
                //<LF>: ASCII character 10
        }
}
char gsm_response()
{
        unsigned char result=0,temp;
        unsigned int state_ok=0,state_err=0;
        delay_ms(2000);
        do
        {
                temp=getchar();
                if ((temp=='O') && (state_ok==0))
                {
                        state_ok=1;
                }
                else if ((temp=='K') && (state_ok==1))
                {
                        result=OK;
                }

                if ((temp=='E') && (state_err==0))
                {
                        state_err=1;
                }
                else if ((temp=='R') && (state_err==1))
                {
                        state_err=2;
                }
                else if ((temp=='R') && (state_err==2))
                {
```

```
                        state_err=3;
                }
                else if ((temp=='O') && (state_err==3))
                {
                        state_err=4;
                }
                else if ((temp=='R') && (state_err==4))
                {
                        result=ERROR;
                }
                if ((sCommand==13) || (sCommand==10))
                {
                        selection=0;
                        header=0;
                        header_ID='';
                        //putchar (13);
                }
        }while(rx_counter0>0);
        return result;
}

unsigned int TUPDATE=5;
void Get_command()
{
        char t_update[5];
        int temp=0;
        if (wait_string("HTTP/1.1 200 OK"))
        {
            wait_string("$");
            relay1=getchar()-0x30;
            wait_string(",");
            relay2=getchar()-0x30;
            wait_string(",");
            t_update[0]=getchar();
            t_update[1]=getchar();
            t_update[2]=getchar();
            t_update[3]=getchar();
            wait_string(",");
            BTEMGSTAT=getchar();
            temp=atof(t_update);
            TUPDATE=temp;
        }
        putchar(13);
        putchar(13);
        putchar('>');
        putchar(TUPDATE/1000%10  +0x30);
        putchar(TUPDATE/100%10  +0x30);
        putchar(TUPDATE/10%10  +0x30);
        putchar(TUPDATE%10  +0x30);
}
void get_server_command()
{
```

```
Busy=1;
clear_serial();
printf("AT+CIPSHUT=?") ;
putchar(13);
if (!wait_string("OK"))
{
        Flag_restart=1;
        printf("exit");
        putchar(13);
        goto exit;
}
printf("AT+CGATT=1");
putchar(13);
if (!wait_string("OK"))
{
        printf("exit");
        putchar(13);
        goto exit;
}

printf("AT+CIICR");
putchar(13);
while(getchar()!='O') ;
printf("AT+CIFSR");
putchar(13);
delay_ms(2000);
printf("AT+CIPSTATUS");
putchar(13);
if (!wait_string("STATE"))
{
        printf("exit");
        putchar(13);
        goto exit;
}
printf("AT+CIPSTART=");
putchar('"');
printf("TCP");
putchar('"');
putchar(',');
putchar('"');
printf("182.253.236.11");
putchar('"');
putchar(',');
printf("80");
putchar(13);
if(wait_string("OK"))
{
        if (!wait_string("CONNECT OK"))
        {
            printf("restart");
            putchar(13);
            Flag_restart=1;
```

```
                goto exit;
            }
        }
        //led_blink(ON,7,1);
        printf("AT+CIPSEND");
        putchar(13);
        if (!wait_string(">"))
        {
                printf("restart");
                putchar(13);
                Flag_restart=1;
                goto exit;
        }
        printf("GET /command.php?ID=");
        printf(ID);
        printf("&BTEMG=");
        putchar(BTEMGSTAT);
        printf(" HTTP/1.1");
        putchar(13);
        putchar(10);
        printf("Host: gpsfence.web.id");
        putchar(13);
        putchar(10);
        putchar(13);
        putchar(10);
        putchar(26);
        wait_string("SEND OK");
        putchar(13);
        Get_command();
        putchar(13);
        printf("AT+CIPCLOSE");
        putchar(13);
        wait_string("O");
        clear_serial();
exit:
        putchar(13);
        printf("AT+CIPCLOSE");
        putchar(13);
        wait_string("O");
        //led_blink(OFF,5,1);
        Busy=0;
        clear_serial();
}
char sendingToDb(GPS_data item)
{
        char stat=0;
        clear_serial();

        Busy=1;
        //led_blink(ON,40,2);
        if(item.UTC[6]!='.')
        {
```

```
    goto skip;
}
printf("AT+CIPSHUT=?");
putchar(13);
if (!gsm_response())
{
        printf("exit");
        putchar(13);
        Flag_restart=1;
        goto exit;
}
printf("AT+CGATT=1");
putchar(13);
if (!wait_string("OK"))
{
        printf("exit");
        putchar(13);
        goto exit;
}
printf("AT+CIICR");
putchar(13);
while(getchar()!='O') ;
printf("AT+CIFSR");
putchar(13);
delay_ms(2000);
printf("AT+CIPSTATUS");
putchar(13);
wait_string("STATE");
printf("AT+CIPSTART=");
putchar('"');
printf("TCP");
putchar('"');
putchar(',');
putchar('"');
printf("182.253.236.11");
putchar('"');
putchar(',');
printf("80");
putchar(13);
if(wait_string("OK"))
{
        if (!wait_string("CONNECT OK"))
        {
            printf("restart");
            putchar(13);
            Flag_restart=1;
            goto exit;
        }
}
//led_blink(ON,7,1);
printf("AT+CIPSEND");
putchar(13);
```

```
if (!wait_string(">"))
{
        printf("restart");
        putchar(13);
        Flag_restart=1;
        goto exit;
}
printf("GET /upload2.php?ID=");
printf(ID);
printf("&utc=");
send_string(item.UTC);
printf("&latd=");
putchar(item.LAT[0]);
putchar(item.LAT[1]);
printf("&latm=");
putchar(item.LAT[2]);
putchar(item.LAT[3]);
putchar(item.LAT[4]);
putchar(item.LAT[5]);
putchar(item.LAT[6]);
putchar(item.LAT[7]);
putchar(item.LAT[8]);
printf("&ns=");
putchar(item.NS[0]);
printf("&lngd=");
putchar(item.LNG[0]);
putchar(item.LNG[1]);
putchar(item.LNG[2]);
printf("&lngm=");
putchar(item.LNG[3]);
putchar(item.LNG[4]);
putchar(item.LNG[5]);
putchar(item.LNG[6]);
putchar(item.LNG[7]);
putchar(item.LNG[8]);
putchar(item.LNG[9]);
printf("&ew=");
putchar(item.EW[0]);
printf("&gpsu=");
send_string(item.GPSU);
printf("&speed=");
putchar(item.SOG[0]);
putchar(item.SOG[1]);
putchar(item.SOG[2]);
putchar(item.SOG[3]);
putchar(item.SOG[4]);
printf("&intbatcap=");// baterry capacity
putchar(INTBATCAP[0]);
putchar(INTBATCAP[1]);
if (INTBATCAP[2]!=',')
{
           putchar(INTBATCAP[2]);
```

```
        }
        putchar('%');
        printf("&intbatstat=");// charging or not charging
        putchar(charger+0x30);
        printf("&extbat="); // external battery available or
not
        putchar(EXTBAT[0]);
        printf("&relay=");
        putchar(relay1+0x30);
        putchar('!');
        putchar(relay2+0x30);
        printf("&BTEMG=");
        putchar(BTEMGSTAT);
        printf(" HTTP/1.1");
        putchar(13);
        putchar(10);
        printf("Host: gpsfence.web.id");
        putchar(13);
        putchar(10);
        putchar(13);
        putchar(10);
        putchar(26);
        wait_string("SEND OK");
        putchar(13);
        Get_command();
        putchar(13);
        clear_serial();
        net_stat=3;
        printf("AT+CIPCLOSE");
        putchar(13);
        wait_string("O");
        skip:
        stat=1;


        exit:
        clear_serial();
        //led_blink(OFF,5,1);
        Busy=0;
        return stat;
}
void dequeue(int* front, int* length, GPS_data queue[])
{
     int where;
     if ( !queueempty( *length ) )
     {
         where = *front;
         *front = (where+1) % MAX_QUEUE_SIZE;
         if (sendingToDb(queue[where]))
         {
            (*length)--;
         }
```

```
     }
}
void Init_hardware()
{
    // Crystal Oscillator division factor: 1
    #pragma optsize-
    CLKPR=0x80;
    CLKPR=0x00;
    #ifdef _OPTIMIZE_SIZE_
    #pragma optsize+
    #endif

    // Declare your local variables here

    // Input/Output Ports initialization
    PORTA=0x00;
    DDRA=0x00;

    PORTB=0b00000100;
    DDRB=0b00011010;

    PORTC=0x00;
    DDRC=0b00001000;

    PORTD=0b11100000;
    DDRD=0b11100000;

    // Timer/Counter 0 initialization
    // Clock source: System Clock
    // Clock value: 18.000 kHz
    // Mode: Normal top=0xFF
    // OC0A output: Disconnected
    // OC0B output: Disconnected
    TCCR0A=0x00;
    TCCR0B=0x00;//TCCR0B=0x05;
    TCNT0=0x00;
    OCR0A=0x00;
    OCR0B=0x00;

    // Timer/Counter 1 initialization
    // Clock source: System Clock
    // Clock value: 18.000 kHz
    // Mode: Normal top=0xFFFF
    // OC1A output: Discon.
    // OC1B output: Discon.
    // Noise Canceler: Off
    // Input Capture on Falling Edge
    // Timer1 Overflow Interrupt: On
    // Input Capture Interrupt: Off
    // Compare A Match Interrupt: Off
    // Compare B Match Interrupt: Off
    //0.05ms  =20000 b1df
```

```
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: 18.000 kHz
// Mode: Normal top=0xFF
// OC2A output: Disconnected
// OC2B output: Disconnected
ASSR=0x00;
TCCR2A=0x00;
TCCR2B=0x00;
TCNT2=0x00;
OCR2A=0x00;
OCR2B=0x00;

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: On
// INT2 Mode: Falling Edge
// Interrupt on any change on pins PCINT0-7: Off
// Interrupt on any change on pins PCINT8-15: On
// Interrupt on any change on pins PCINT16-23: Off
// Interrupt on any change on pins PCINT24-31: Off
EICRA=0x20;
EIMSK=0x04;
EIFR=0x04;
PCMSK1=0x04;
PCICR=0x02;
PCIFR=0x02;

// Timer/Counter 0 Interrupt(s) initialization
TIMSK0=0x01;

// Timer/Counter 1 Interrupt(s) initialization
TIMSK1=0x01;

// Timer/Counter 2 Interrupt(s) initialization
TIMSK2=0x01;

// USART0 initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART0 Receiver: On
```

```
    // USART0 Transmitter: On
    // USART0 Mode: Asynchronous
    // USART0 Baud Rate: 115200
    UCSR0A=0x00;
    UCSR0B=0xD8;
    UCSR0C=0x06;
    UBRR0H=0x00;
    UBRR0L=0x09;

    // USART1 initialization
    // Communication Parameters: 8 Data, 1 Stop, No Parity
    // USART1 Receiver: On
    // USART1 Transmitter: Off
    // USART1 Mode: Asynchronous
    // USART1 Baud Rate: 115200
    UCSR1A=0x00;
    UCSR1B=0x90;
    UCSR1C=0x06;
    UBRR1H=0x00;
    UBRR1L=0x09;
//    UCSR1A=0x00;
//    UCSR1B=0x00;
//    UCSR1C=0x00;
//    UBRR1H=0x00;
//    UBRR1L=0x00;

    // Analog Comparator initialization
    // Analog Comparator: Off
    // Analog Comparator Input Capture by Timer/Counter 1: Off
    ACSR=0x80;
    ADCSRB=0x00;
    DIDR1=0x00;

    // ADC initialization
    // ADC Clock frequency: 144.000 kHz
    // ADC Voltage Reference: AVCC pin
    // ADC Auto Trigger Source: ADC Stopped
    // Digital input buffers on ADC0: On, ADC1: On, ADC2: On,
ADC3: On
    // ADC4: On, ADC5: On, ADC6: On, ADC7: On
    DIDR0=0x00;
    ADMUX=ADC_VREF_TYPE & 0xff;
    ADCSRA=0x87;

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

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

```
    // Watchdog Timer initialization
    // Watchdog Timer Prescaler: OSC/1024k
    // Watchdog Timer interrupt: Off
    //#pragma optsize-
    //#asm("wdr")
    //WDTCSR=0x39;
    //WDTCSR=0x29;
    //#ifdef _OPTIMIZE_SIZE_
    //#pragma optsize+
    //#endif
}
void main(void)
{
// Declare your local variables here
unsigned int sec_tick=0;
unsigned int min_tick=0,sat=0;
unsigned char get_sec_tick='',get_min_tick='';
int length_queue,front_queue;
unsigned long count_respons=0;

GPS_data queue[MAX_QUEUE_SIZE];
initialize_Queue(&front_queue, &length_queue);

Init_hardware();
delay_ms(2000);
PWRKEY_P=1;
delay_ms(2100);
PWRKEY_P=0;
booting_SIM908();
initGPRS();
LED_STAT=1;
Busy=0;
get_min_tick=NMEA.UTC[3];
#asm("sei")// Global enable interrupts

        while (1)
        {
// Place your code here
cekSerial();
count_respons++;
if(count_respons>10000000){Flag_restart=1;}
if ((Flag_restart==1) || (StatusSIM908==0))
{
restart_SIM908();
Flag_restart=0;
count_respons=0;
net_stat=1;
}
if (header==0)
{
if (get_sec_tick != NMEA.UTC[5])
{
```

```
count_respons=0;
get_sec_tick = NMEA.UTC[5];
sec_tick++;
if ((sec_tick%net_stat)==0)
{
//PORTB.0=1;
LED_STAT=0;
delay_ms(100);
//PORTB.0=0;
LED_STAT=1;
}
batt_management(sec_tick);
dequeue(&front_queue, &length_queue, queue);
if (sec_tick>=40)
{
//PORTB.0=0;
LED_STAT=1;
clear_serial();
printf("TN");
putchar(min_tick/100%10    +0x30);
putchar(min_tick/10%10    +0x30);
putchar(min_tick%10    +0x30);
putchar(13);
printf("TU");
putchar(TUPDATE/1000%10  +0x30);
putchar(TUPDATE/100%10  +0x30);
putchar(TUPDATE/10%10  +0x30);
putchar(TUPDATE%10  +0x30);
putchar(13);
printf("BT:");
putchar(BTEMGSTAT);
putchar('>');
send_string(NMEA.GPSU);
putchar(13);
wait_string("ERROR");
printf("AT");
putchar(13);
wait_string("OK");
sec_tick=0;

}}
if (NMEA.UTC[3]!=get_min_tick)
{
get_min_tick=NMEA.UTC[3];
min_tick++;
if (min_tick>=TUPDATE)
{
sec_tick=0;
sat=atoi(NMEA.GPSU);
//if (atoi(NMEA.LNG[0]!='')
if (sat>0)
{
```

```
//sendingToDb(NMEA);
enqueue(front_queue,&length_queue,queue,NMEA);
}
else
{
printf("no sattelites");
putchar(13);
wait_string("ERROR");
get_server_command();

}
min_tick=0;
}
else
{
get_server_command();
sec_tick=0;
}
LED_STAT=1;
}}}}
```

- **Listing Program Halaman WEB**
  - **Viewdevice.php**

```
<!DOCTYPE html>
<html>
<head>
<?php    echo    '<meta    http-equiv="refresh"    content="10;
URL=viewdevice.php?DVID='.$_GET["DVID"]. '">'; ?>

<?php
        $rly1=$_GET["RL1"];
        $rly2=$_GET["RL2"];
        $getTupdt=$_GET["tupdt"];
        $BT_EMG=$_GET["BTEMG"];
        $dv=$_GET["DVID"];

$con1                                                          =
mysql_connect("gpsfence.web.id","u5889786","tugasakhir2011");
if (!$con1)
 {
  die('Could not connect: ' . mysql_error());
  }
                              mysql_select_db("u5889786_dvadm",
$con1);
                              if ($rly1!='')
                              {
                                      $sql1="UPDATE  Device_list
SET Relay1='$rly1' WHERE Device_ID='$dv'";

                                      if
(!mysql_query($sql1,$con1))
```

```
                                        {
                                                die('Error:   '   .
mysql_error());
                                        }
                                }
                                else if ($rly2!='')
                                {
                                        $sql1="UPDATE  Device_list
SET Relay2='$rly2' WHERE Device_ID='$dv'";
                                        if
(!mysql_query($sql1,$con1))
                                        {
                                                die('Error:   '   .
mysql_error());
                                        }
                                }
                                else if ($getTupdt!='')
                                {
                                        $sql1="UPDATE  Device_list
SET Update_interval='$getTupdt' WHERE Device_ID='$dv'";
                                        if
(!mysql_query($sql1,$con1))
                                        {
                                                die('Error:   '   .
mysql_error());
                                        }
                                }
                                else if ($BT_EMG!='')
                                {
                                        $sql1="UPDATE  Device_list
SET Button_Emg='0' WHERE Device_ID='$dv'";
                                        if
(!mysql_query($sql1,$con1))
                                        {
                                                die('Error:   '   .
mysql_error());
                                        }
                                }
                                mysql_close($con1);

                $dv_ID=$_GET["DVID"];
                $DVID;
                $DVIMEI;
                $RELAY1;
                $RELAY2;
                $T_UPDATE;
                $STATEMG;
                $con                                                    =
mysql_connect("gpsfence.web.id","u5889786","tugasakhir2011");
                if (!$con)
                {
```

```
                        die('Could    not    connect:    '    .
mysql_error());
                }
                mysql_select_db("u5889786_dvadm", $con);
                $result = mysql_query("SELECT * FROM Device_list
WHERE Device_ID='$dv_ID'");
                while($row = mysql_fetch_array($result))
                {
                                $DVID=$row['Device_ID'];
                                $DVIMEI=$row['IMEI'];
                                $RELAY1=$row['Relay1'];
                                $RELAY2=$row['Relay2'];
                                $T_UPDATE=$row['Update_interval'];
                                $STATEMG=$row['Button_Emg'];

                }
                mysql_close($con);
                if($RELAY1=="OFF"){$notRELAY1="ON";}
                else{$notRELAY1="OFF";}
                if($RELAY2=="OFF"){$notRELAY2="ON";}
                else{$notRELAY2="OFF";}
                ?>

        <script type="text/javascript">
                        function validate_text(evt)
                        {
                                var    theEvent    =    evt    ||
window.event;
                                var  key  =  theEvent.keyCode  ||
theEvent.which;
                                if (!((key>=48) && (key<=57)))
                                {
                                        theEvent.returnValue     =
false;
                                if(theEvent.preventDefault)
theEvent.preventDefault();
                                }
                        }
                        function validate()
                        {
                                var
txt_inpt=document.getElementById("inptext");
                                var
txtlength=txt_inpt.value.length;
                                if        ((txtlength>0)        &&
((txt_inpt.value)>0))
                                {

        document.getElementById("btn").disabled=false;
                                }
                                else
                                {
```

```
        document.getElementById("btn").disabled=true;
                                }
                        }
        </script>

    </head>

<body>
                DEVICE ID :<br>
                <?php echo $DVID;?><br>
                IMEI :<br>
        <?php echo $DVIMEI;?><br>
                Relay1:<br>
        <?php                           echo                            "<a
href=\"http://gpsfence.web.id/viewdevice.php?DVID=".$DVID."&RL1
=".$notRELAY1."&RL2=&BTEMG=\">".$RELAY1."</a>"  ?>
        (Click to edit)<br>
                Relay2 :<br>
        <?php                           echo                            "<a
href=\"http://gpsfence.web.id/viewdevice.php?DVID=".$DVID."&RL1
=&RL2=".$notRELAY2."&BTEMG="."\">".$RELAY2."</a>"  ?>
        (Click to edit)<br>
        Status : <?php if($dv!='')
                                                {
                                                        if
($STATEMG=='0')
                                                        {

        echo " Normal";
                                                        }else
                                                        {

        echo " Emergency ";

        echo                                                            "<a
href=\"http://gpsfence.web.id/viewdevice.php?DVID=".$DVID."&RL1
=&RL2=&BTEMG=0"."\">"."Clear"."</a>";
                                                        }
                                                }
                                ?>
        <br>
        Time for Update :
        <?php echo $T_UPDATE." Min"; ?>
                <br>
        <form method="get" action="viewdevice.php?">
                <input  type="hidden"  name="DVID"  value="<?php
echo $DVID;?>">
        <input type="hidden" name="RL1" value="">
        <input type="hidden" name="RL2" value="">
```

```
        <input type="text" name="tupdt" size="4" maxlength="4"
onKeyPress="validate_text(event)"          onKeyUp="validate()"
id="inptext" > Min
        <input     type="submit"     value="Update"     id="btn"
disabled=true>
        </form>
</body>
</html>
```

- Upload2.php

```
<?php
date_default_timezone_set('Asia/Jakarta');   /*waktu   mengikuti
waktu di jakarta*/

/*data menyesuaikan dengan tabel di database*/
$deviceID = $_GET["ID"]."_".date("ymd");
$utc =  $_GET["utc"];
$latd =  $_GET["latd"];
$latm =  $_GET["latm"];
$ns =  $_GET["ns"];
$lngd =  $_GET["lngd"];
$lngm =  $_GET["lngm"];
$ew =  $_GET["ew"];
$gpsu =  $_GET["gpsu"];
$speed =  $_GET["speed"];
$intbatcap =  $_GET["intbatcap"];
$intbatstat =  $_GET["intbatstat"];
$extbat =  $_GET["extbat"];
$relay =  $_GET["relay"];
$btemg =  $_GET["betmg"];
$STATEMG;

/*Tampilan Waktu*/
$hour = $utc[0].$utc[1];
$minute = $utc[2].$utc[3];
$Sec = $utc[4].$utc[5].$utc[6].$utc[7].$utc[8].$utc[9];
$hour = $hour + 7;
if ($hour>24)
        {
                $hour = $hour - 24;
        }
$utc = $hour.":".$minute.":".$Sec;

/*Tampilan arah*/
$latitude = ($latd + ($latm/60));
$longitude = ($lngd + ($lngm/60));
if ($ns=="S")
        {
                $latitude = $latitude * -1;
        }
if ($ew == "W")
```

```
	{
		$longitude = $longitude * -1;
	}
if ($speed == "")
	{
		$speed = "-";
	}
$con                                                              =
mysql_connect("gpsfence.web.id","u5889786","tugasakhir2011");
	if(!$con)
    {
	die('Could not connect: ' . mysql_error());
    }
mysql_select_db("u5889786_dvdat",$con);

$sql               =               "INSERT               INTO
$deviceID(UTC,LATITUDE,LONGITUDE,GPSU,SPEED,INTBATCAP,INTBATSTA
T,EXTBAT,RELAY)  VALUES  ('$UTC',  '$latitude',  '$longitude',
'$gpsu',  '$speed',  '$intbatcap',  '$intbatstat',  '$extbat',
'$relay')";



if(!mysql_query($sql,$con))
	{
		$sql = "CREATE TABLE $deviceID(Data_ID INT NOT
NULL  AUTO_INCREMENT,    PRIMARY  KEY(Data_ID),UTC  varchar(15),
LATITUDE varchar(25) , LONGITUDE varchar(25), GPSU varchar(3),
SPEED varchar(10), INTBATCAP varchar(6), INTBATSTAT varchar(20),
EXTBAT varchar(20), RELAY varchar(200))";

if(!mysql_query($sql,$con))
	{
		die('Error :  ' . mysql_error());
	}
		else
	{
		$sql        ="INSERT        INTO        $deviceID
(UTC,LATITUDE,LONGITUDE,GPSU,SPEED,INTBAT_CAP,INTBAT,STAT,EXTBA
T,RELAY)  VALUES  ('$UTC',  '$latitude',  '$longitude',  '$gpsu',
'$speed', '$intbatcap', '$intbatstat', '$extbat', '$relay')";

if(!mysql_query($sql,$con))
	{
		die('Error :  ' . mysql_error());
	}
		else
	{
		echo"create table and 1 record added";
	}}}
		else
	{
```

```
                echo"1 record added";
        }
$mysql_close($con);

$dv_ID=$_GET["ID"];
$con1                                                           =
mysql_connect("gpsfence.web.id","u5889786","tugasakhir2011");

if(!$con1)
        {
                die('could not connect: ' . mysql_error());
        } mysql_select_db("u5889786_dvadm",$con1);
$result1   =   mysql_query("SELECT   *   FROM   Device_list   WHERE
Device_ID = '$dv_ID'");
while($row1 = mysql_fetch_array($result1))
        {
                if($row1['Relay1']=="ON")
        {
                echo "$0";
        }
                else
        {
                 echo "$1";
         }
                if($row1['Relay2']=="ON")
        {
                echo "$0";
        }
                else
        {
                 echo ",1";
         }
$updt_int=$row1['Update_interval'];
$STATEMG = $row1['Button_EMG'];
        }
if(($btemg==1) && ($STATEMG!=1))
        {
$commandSQL="UPDATE   DEVICE_list   SET   BUTTON_Emg='1'   WHERE
Device_ID='$dv_ID'";


 mysql_query($commandSQL,$con1);
        }
$length_number=strlen($updt_int);
if($length_number==1) {echo ",000".$updt_int;}
else if($length_number==2) {echo ",00".$updt_int;}
else if($length_number==3) {echo ",0".$updt_int;}
else if($length_number==4) {echo ",".$updt_int;}
echo",0";
mysql_close($con1);
echo $deviceID. "time:  ".$utc;
```

```
?>
```

- **Collect.php**

```
<?php
$deviceID = $_GET["ID"];
$data = array(array());
$id = 0;
$temp = 0;

$ct = 0;
$con                        =                          mysql_connect
("gpsfence.web.id","u5889786","tugasakhir2011");
if(!$con)
       {
       die('Could not connect: ' .mysql_error());
       }
mysql_select_db("u5889786_dvdat",$con);
$result = mysql_query("SELECT * FROM $deviceID ORDER BY Data_ID
ASC");

//database 8 data
while($row = mysql_fetch_array($result))
       {
       $data[$ct][0]=$row['Data_ID'];
       $data[$ct][1]=$row['UTC'];
       $data[$ct][2]=$row['LATITUDE'];
       $data[$ct][3]=$row['LONGITUDE'];
       $data[$ct][4]=$row['GPSU'];
       $data[$ct][5]=$row['SPEED'];
       $data[$ct][6]=$row['INTBAT_CAP'];
        $data[$ct][7]=$row['EXTBAT'];
       $data[$ct][8]=$row['RELAY'];
       $ct++;
       }
mysql_close($con);

for($a=0;$a<$ct;$a++)
       {
        if ($data[$a][7]==1)
       {
       $data[$a][7]="Plugged in";
       }
       else
       {
       $data[$a][7]="Unplug";
       }
       $relayparse=$data[$a][8];
       if ($relayparse[0]==1){$rly1="OFF";}else{$rly1="ON";}
       if ($relayparse[2]==1){$rly2="OFF";}else{$rly2="ON";}
       $data[$a][8]="RL1:".$rly1." || RL2:".$rly2;
       echo
$data[$a][0]."$".$data[$a][1]."$".$data[$a][2]."$".$data[$a][3]
```

```
."$".$data[$a][4]."$".$data[$a][5]."$".$data[$a][6]."$".$data[$
a][7]."$".$data[$a][8]."@";
        }
echo "!"$sct;
?>
```

- **Index.html**

```
<!DOCTYPE html>
<html>
<head>
<?php
$list_table="";
$cdn=0;
$cdd=0;
$device_name=array();
$device_date=array(array());
$dbname='u487078170_dvdat';
$device_view=$_GET['DeviceID']."_".$_GET['DATE'];
if(!mysql_connect("gpsfence.web.id","u487078170",
"tugasakhir2011"))
        {
        exit;
        }
   $sql = "SHOW TABLES FROM $dbname";
   $Result = mysql_query($sql);
if(!$result)
        {
         exit;
        }
while($row = mysql_fetch_row($result))
        {
    $list_table=$list_table.$row[0]." ";
    $device_date[$cdn][$cdd]=$temp[1];
    $device_date[$cdn][$cdd]=$temp[1];
    $device_date[$cdn][$cdd]=$temp[1];
    }
mysql_free_result($result);
?>
<link rel="stylesheet" href="mapstyle.css" />
    <meta     http-equiv="Content-Type"     content="text/html;
charset=utf-8" />

   <script type="text/javascript"

src="http://maps.googleapis.com/maps/api/js?AIzaSyAJn8mfzvALY2k
bgyOS0tSksxx4gNPXpD0&sensor=false">

</script>
    <script type="text/javascript">
        // Use JQuery to trigger the loading of the Map
var getstr;
var passing = new Array();
```

```
var data_ID = new Array();
var data_gos = new Array();
var lintang = new Array();
var bujur = new Array();
var gpsu = new Array();
var speed = new Array();
var intbatcap = new Array();
var time = new Array();
var jumlah_data = 0;
var temp = new Array();
var ct=0;
var map;
var markersArray = [];
var device_view = "<?php echo $device_view; ?>";
var list_table = "<?php echo $list_table; ?>"
var device_name = new Array();
var device_date = new Array();

var boundaryColor = '#ED1B24'; // initialize color of polyline
var polyCoordinates =[]; // initialize an array where we store
latitude and longitude pair
var count=0;

var Fence;
function get_data()
    {
    if(window.XMLHttpRequest)
    {
    //code for IE7+, Firefox, Chrome, Opera, Safari xmlhttp =
new XMLHttpRequest();
    }
    else
    {
    //code     for     IE6,     IE5     xmlhttp     =     new
ActiveXObject("Microsoft.XMLHTTP")
    }
xmlhttp.onereadystatechange = function()
        {
    if(xmlhttp.readyState==4 && xmlhttp.status==200)
    {
    getstr = xmlhttp.responseText;
    extract_data(getstr);
    }}
xmlhttp.open("GET","http://gpsfence.web.id/collect.php?ID="+
device_view ,true);
xmlhttp.send();
       }

function extract_data(data_string)
       {
    passing = data_string.split("!");
    data_gps = passing[0].split("@");
```

```
    jumlah_data = passing[1];
    for(var ii=0; ii<jumlah_data;ii++)
        {
        tempt = data_gps[ii].split("$");
         data_ID[ii] = temp[0];
         time[ii] = temp[1];
         lintang[ii] = temp[2];
         bujur[ii] = temp[3];
         gpsu[ii] = temp[4];
         speed[ii] = temp[5];
         intbatcap[ii] = temp[6];
        }
    create_marker();
    }
 function initialize()
        {
    var    surabaya    =    new    google.maps.LatLng(-7.2609401,
112.7741939);
    var  aaa             =  new  google.maps.LatLng(-7.2609401,
112.7751939);
    var  mapOptions  =  {zoom:  15,center:  surabaya,mapTypeId:
google.maps.MapTypeId.ROADMAP,
         mapTypeControl: false};
    var map = new
    google.maps.Map(document.getElementById("Gmap"),mapOptions);

var                pinImage                =                new
google.maps.MarkerImage("http://chart.apis.google.com/chart?chs
t=d_map_pin_letter&chld=%E2%80%A2|" + pinColor,
new google.maps.Size(21, 34),
    new google.maps.Point(0,0),
    new google.maps.Point(10, 34));

var pinColor = "FE7569";
var infowindow = new google.maps.InfoWindow();
var marker, i;

 var locations = [
      ['Vehicle A', -7.2619401, 112.7741949],
      ['Vehicle A', -7.2639401, 112.7731949],
      ['Vehicle A', -7.2639401, 112.7561949],
      ['Vehicle A', -7.2649401, 112.7721949],
      ['Vehicle A', -7.2629801, 112.7881949],
      ['Vehicle A', -7.2549101, 112.7891949],
    ];
var  iconURLPrefix  =  'http://maps.google.com/mapfiles/ms/icons/';
   var icons = [
      iconURLPrefix + 'red-dot.png',
      iconURLPrefix + 'red-dot.png',
      iconURLPrefix + 'red-dot.png',
      iconURLPrefix + 'red-dot.png',
      iconURLPrefix + 'blue-dot.png',
```

```
      iconURLPrefix + 'blue-dot.png'
    ]
    var iconsLength = icons.length;

 var triangleCoords = [
    new google.maps.LatLng(-7.2599401, 112.7831939),
    new google.maps.LatLng(-7.2659401, 112.7831939),
    new google.maps.LatLng(-7.2649401, 112.7521939),
    new google.maps.LatLng(-7.2579401, 112.7541939)
  ];

 Fence = new google.maps.Polygon({
    paths: triangleCoords,
    strokeColor: '#FF0000',
    strokeOpacity: 0.8,
    strokeWeight: 2,
    fillColor: '#FF',
    fillOpacity: 0.5,
    editable: true,
    draggable: true
  });
  Fence.setMap(map);

for( i = 0; i < locations.length; i++ ) {
var marker = new google.maps.Marker({
            position:  new  google.maps.LatLng(locations[i][1],
locations[i][2]),
            map:   map,   icon:   icons[iconCounter],animation:
google.maps.Animation.DROP
        });
 iconCounter++;
      if(iconCounter >= iconsLength) {
        iconCounter = 0;
      }
google.maps.event.addListener(marker, 'click', (function(marker,
i) {
        return function() {
          infowindow.setContent(locations[i][0]);
          infowindow.open(map, marker);

 if
(google.maps.geometry.poly.containsLocation(this.getPosition(),
Fence) == false) {
        alert("Out Of Geofence");

marker.setIcon('http://maps.google.com/mapfiles/ms/icons/blue-
dot.png');}
        else{alert("Inside Geofence");

marker.setIcon('http://maps.google.com/mapfiles/ms/icons/red-
dot.png');};
       }
```

```
     })(marker, i));
}
google.maps.event.addListener(Fence,    'dragend',    function()
{alert('Marker Change');})
google.maps.event.addDomListener(window, 'load', initialize);
}

function StartTime()
        {
    get_data();
    initialize();
    select_device();
    looptimer();
        }
function looptimer()
        {
    get_data();
    t=setTimeout('looptimer()',20000);
        }
function select_device()
        {
        var temp_lst_dvc=list_table.split(" ");
        var cdn=0;
        var cdd=0;
        var ct=1;
        var dvc_lst;

        while (ct<(temp_lst_dvc.length))
        {
                dvc_lst=temp_lst_dvc[ct].split("_");
            if(ct>1)
            {
               {
               device_date[cdn].push(dvc_lst[1]);
               }
             else
               {
               cdn++;
               device_name[cdn] = dvc_lst[0];
               device_date[cdn]= new Array();
               device_date[cdn].push(dvc_lst[1]);
               }
        }
      else
      {
        device_date[cdn]= new Array();
        device_name[cdn]=dvc_lst[0];
      device_date[cdn].push(dvc_lst[1]);
        }
                ct++;
        }
for(var dev=0;dev<device_date.length-1;dev++)
```

```
{
var dvc=document.getElementById("DeviceID");
var option=document.createElement("option");
option.text=device_name[dev];
option.id=device_name[dev];
try
{
dvc.add(option,dvc.options[null]);
}
catch (e)
{
dvc.add(option,null);
}
}
<?php
$deviceslc=$_GET['DeviceID'];
?>
var curent_dev="<?php echo $deviceslc;  ?>";
if (curent_dev)
{
document.getElementById(curent_dev).selected=true;
}
else
{
document.getElementById(device_name[0]).selected=true;
}
select_date();
}
function select_date()
{
var dev_sel=document.getElementById("DeviceID").selectedIndex;
var
dev_name=document.getElementsByTagName("option")[dev_sel].value;
var index_device=device_name.indexOf(dev_name);
document.getElementById("DATE").innerHTML = "";
for (var i=0;i<device_date[index_device].length;i++)
{
var dev_date=document.getElementById("DATE");
var option=document.createElement("option");
option.text=device_date[index_device][i];
option.id=device_date[index_device][i];
try
{
dev_date.add(option,dev_date.options[null]);
}
catch (e)
{                                       dev_date.add(option,null);
}
}
<?php
$dateslc=$_GET['DATE'];
?>
```

```
var curent_date="<?php echo $dateslc;  ?>";
document.getElementById(curent_date).selected=true;
}

    </script>
</head>

<body onload="initialize()" >
<div id="Gmap" style="width:100%; height:600px"></div>
<div id="kontainer">
<div id="menu">
<embed       src=http://flash-clocks.com/free-flash-clocks-blog-
topics/free-flash-clock-121.swf                id="digitalclock"
wmode=transparent type=application/x-shockwave-flash></embed>

<br>
<br>
<div id="SelectDevice">
<form name="form1" method="get" action="index.html">
<i><center>Search Device</center></i>
<br>
<label><strong>Select    Device    :    <select    name="DeviceID"
id="DeviceID" onChange="select_date()">
</select>
<br>
select Date :
</strong></label>
<strong>
<label>
<select name="DATE" id="DATE">
</select>
</label>
<br>
</strong>
<strong>
<input type="submit" value="Go">
</strong>
</form>
</div>
<?php                          echo                          "<iframe
src=".'"',"viewdevice.php?DVID=".$deviceslc.'"'."id=".'"'."Show
Dev".'"'."></iframe>
<br>
<br>
<a href="adddevice.php" > Add Device</a>
</div>
<div id="Gmap"></div>
</div>

</body>
</html>
```

Vehicle Security System using Geofence on Google Map

Egber Pangaliela[1)], Hartono Pranjoto, Ph.D[2)]
Email : egberpangaliela@yahoo.com

ABSTRAK

Global Positioning System (GPS) receiver dipasang di kendaraan bermotor telah digunakan untuk melacak kendaraan. Posisi kendaraan ditransmisikan melalui jaringan nirkabel menggunakan jaringan telepon seluler dikenal sebagai GSM (Global System untuk komunikasi Mobile). Sebuah kendaraan yang memiliki receiver GPS yang terpasang onboard, terhubung dengan modem GPRS dan terhubung ke sistem komputer di internet yang dapat dipantau dan memberikan peringatan ketika perjalanan di luar area yang telah ditetapkan. Daerah ini sangat penting bagi banyak situasi seperti kota sewa mobil, perusahaan truk untuk mengirim barang dari satu kota ke kota lain dan perusahaan logistik dengan banyak armada.

Sistem yang dirancang di sini adalah modul dengan GPS dan GPRS sudah terintegrasi dalam satu modul. Output dari penerima GPS terhubung ke mikrokontroler. Mikrokontroler menentukan data yang dikumpulkan melalui penerima GPS dan kemudian mengirimkan informasi tersebut ke sistem komputer melalui modem GPRS. Mikrokontroler juga menerima perintah dari sistem komputer melalui koneksi GPRS dan kemudian dapat ACCT sesuai, seperti perubahan frekuensi geo-koordinat atau mengubah mesin kendaraan off jika diperlukan. Perangkat ini akan membantu pengguna untuk melacak kendaraan yang melalui GoogleMap dengan GPS koordinat data yang dikirim ke server database setiap beberapa waktu.

perangkat ini akan memungkinkan operator untuk mematikan mesin kendaraan dan satu perangkat lain onboard, kendaraan jika diperlukan. Dengan fitur Geofence pada server Web menggunakan HTML5, pagar virtual telah dibangun di sekitar GoogleMap dan ketika kendaraan bergerak di luar pagar pengguna dapat diberitahu baik melalui email atau perubahan warna pada halaman web. Perangkat ini telah diuji dan terbukti bekerja dengan semua dikondisikan yang disebutkan di atas. Sistem komputer yang menampilkan halaman web bersama dengan geofence yang telah dikembangkan dan terbukti bekerja dengan baik seperti yang ditunjukkan.

Kata Kunci: vehicle geofence, Google map, fleet management, GPS assisted vehicle tracking

PENDAHULUAN

Sistem pelacakan kendaraan menggunakan Global Positioning System (GPS) adalah sistem yang menggunakan GPS untuk menemukan koordinat geografis kendaraan yang telah dikembangkan sebelumnya. Perangkat itu sendiri pada dasarnya adalah penerima radio disetel ke frekuensi frekuensi transmisi dari satelit GPS yang memungkinkan penerima untuk menghitung koordinat geografis. Ada beberapa set data yang dapat diperoleh dari satelit GPS seperti posisi yang akurat dari penerima dalam radius tertentu, jumlah satelit yang diterima, kecepatan bergerak GPS penerima, dan tanggal / waktu yang akurat berdasarkan waktu *Universal Coordinated*. jarak antara satelit transmisi dan penerima GPS ditentukan dengan menggunakan selang waktu antara satelit yang menggunakan jam atom yang sangat akurat dan penerima menggunakan Kristal kuarsa yang kurang akurat.

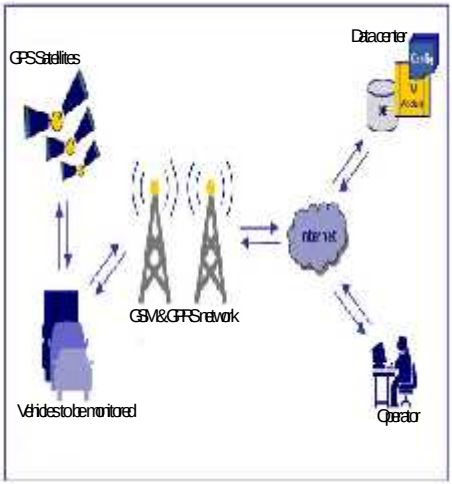
[1)] Mahasiswa di Jurusan Teknik Elektro Fakultas Teknik Universitas Katolik Widya Mandala Surabaya
[2),]Staff Pengajar di Jurusan Teknik Elektro Fakultas Teknik Elektro Universitas Katolik Widya Mandala Surabaya

Meskipun jam Kristal dan kritstal kuarsa memiliki waktu yang kurang akurat, tetapi hasil dari jarak masih sangat di percaya sampai resolusi satu meter atau kurang. Sinyal yang dikirim oleh satelit termasuk *timestamp* sinyal yang akan mengirim dan menerima serta menentukan jarak dengan mengukur waktu untuk perjalanan ke penerima.

Menggunakan metode triangulasi berdasarkan jarak dari satelit GPS, posisi penerima GPS akan diketahui secara tepat dalam beberapa meter. Hal ini dapat dicapai karena posisi yang tepat dari satelit GPS yang sangat baik dan dapat diandalkan. Ketidakpastian jarak dapat timbul karena beberapa fenomena fisik seperti gradien suhu pada atmosfer, bounce sinyal karena benda, dan kekuatan sinyal satelit yang diterima oleh penerima GPS. Data lain juga diperoleh dari penerima GPS seperti kecepatan bergerak kendaraan, menuju pergerakan penerima GPS, akurasi posisi, jumlah sinyal satelit yang diterima, dan kekuatan sinyal yang diterima.

Data dikirim dari penerima GPS biasanya sudah dalam format digital dan di kirim ke komputer atau mikrokontroler menggunakan koneksi serial dengan sinyal amplitudo 0-5 Volts. Format data yang sudah distandarisasi menggunakan format NMEA-0183 dengan standar terbaru yang versi 4.10. Berdasarkan standar NMEA-0183, data tingkat serial penerima GPS adalah 4800 bit per detik dengan 8 bit data dan satu stop bit (4800 bps 8N1). Selain mengirim data melalui koneksi serial ini, pengaturan dari penerima GPS dapat dilakukan melalui koneksi serial ini. Pengaturan termasuk format data, unit pengukuran, dan informasi waktu.

Data GPS yang diperoleh dari kendaraan yang bergerak ditransmisikan menggunakan jaringan data nirkabel dari bagian dari GSM (Global System for Mobile Communication) yang juga dikenal sebagai telepon seluler. Komponen Data bagian dari GSM disebut General Packet Radio Service (GPRS) dapat terhubung ke komputer host dengan kecepatan hingga 128 kb / s yang cukup cepat untuk aplikasi ini. Untuk menggunakan GPRS bagian dari jaringan GSM, GPRS modem diperlukan untuk menghubungkan bagian data dari jaringan nirkabel. Modem GPRS terhubung ke mikrokontroler melalui koneksi serial simiar untuk koneksi ke GPS disebutkan sebelumnya.



Gambar 1. Vehicle monitoring/ management system using GPS via GPRS on network.

Gambar 1 menggambarkan sistem yang digunakan untuk keamanan kendaraan dan sistem manajemen pada GPS dibantu kendaraan menggunakan *geofence* dan peta sistem Google. Ada satelit GPS untuk membantu GPS receiver dipasang pada kendaraan yang akan dimonitor untuk menemukan koordinat geografis mereka. Koordinat kendaraan kirim via modem GPRS menggunakan jaringan GSM ke Internet untuk sistem komputer biasanya di sebuah pusat data dengan server database dan web server sudah terinstal pada sistem. Seorang operator juga terhubung ke Internet mengakses server melalui web browser untuk mengakses data koordinat dari kendaraan yang akan dimonitor.

Web server dari sistem komputer dengan kemampuan HTML5 dapat memberikan geolocation kendaraan yang akan dipantau pada halaman web overlay dengan peta lokasi menggunakan *Google Map* dimana dapat di bentuk *Geofence*.

Operator memiliki beberapa keistimewaan monitoring dan kontrol kendaraan. Operator dapat melacak kendaraan dan tempat *geofence* di sekitar kendaraan dan beberapa fitur lainnya dibahas kemudian. Dengan fitur ini, kendaraan yang akan melampaui geo-pagar atau pagar virtual dapat ditampilkan pada layar dan operator dapat visual waspada dan kemudian dapat mengambil tindakan yang sesuai.

TINJAUAN PUSTAKA

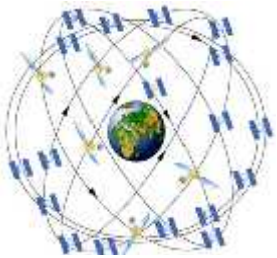
GPS *(Global Positioning System)*

GPS adalah singkatan dari *Global Positioning System* yang merupakan sistem untuk menentukan posisi dan navigasi secara global dengan menggunakan satelit dan metode Triangulasi. Sistem tersebut merupakan sistem yang pertama kali dikembangkan oleh Departemen Pertahanan Amerika yang awalnya diperuntukan bagi kepentingan militer. NAVSTAR GPS (*Navigation Satellite Timing and Ranging Global Positioning System*) adalah nama asli dari Sistem GPS, yang mempunyai tiga segmen yaitu: satelit (*Space Segment*), pengendali (*Control Segment*), dan penerima/pengguna (*User Segment*).

Satelit GPS yang mengorbit bumi seluruhnya berjumlah 24 buah, 21 buah aktif bekerja dan 3 buah sisanya adalah cadangan. Satelit ini bertugas untuk menerima dan menyimpan data yang ditransmisikan oleh stasiun-stasiun pengendali, menyimpan dan menjaga informasi waktu berketelitian tinggi (jam atom di satelit), dan memancarkan sinyal serta informasi secara kontinyu ke perangkat penerima (*receiver*). Segmen pengendali bertugas untuk mengendalikan satelit dari bumi yaitu untuk melihat keadaan satelit, penentuan serta prediksi orbit, sinkronisasi waktu antar satelit, dan mengirimkan data ke satelit. Sedangkan segmen penerima bertugas menerima data dari satelit dan memprosesnya untuk menentukan posisi, arah, jarak dan waktu yang diperlukan oleh pengguna. Pada skripsi ini, digunakan GPS komersial dengan tingkat akurasi posisi sebesar + 10 meter yang berfungsi untuk menentukan posisi alat tersebut berada agar dapat ditampilkan pada peta google maps.

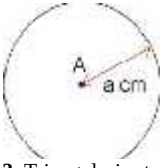
- **Cara Kerja GPS**

Teknologi GPS memerlukan 24 satelit buatan (mengorbit pada ketinggian 20.200 km), yang disebut juga *space segment* agar semua titik di permukaan bumi dapat terpantau. Gambar 2 menggambarkan 6 bidang orbit satelit yang masing-masing bidang berjarak 60^o (6 bidang agar memenuhi 360^o), dan tiap bidang orbit terdapat 4 satelit. Dengan susunan seperti ini, diharapkan semua titik di permukaan bumi dapat dipantau oleh 5-10 satelit dalam waktu bersamaan sehingga dapat menyediakan data dan informasi yang sangat akurat.



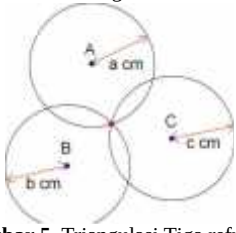
Gambar 2. Sistem satelit GPS

Jumlah minimal yang dibutuhkan untuk dapat menentukan lokasi (koordinat) obyek yang diamati adalah 4 satelit. Hal ini berhubungan dengan konsep Triangulasi. Triangulasi dapat dianalogikan sebagai berikut: Suatu titik A berada pada jarak a cm dari pengamat X. Dari informasi ini dapat diketahui bahwa X dapat terletak di mana saja sepanjang keliling lingkaran dengan radius a cm (Gambar 3). Titik B diketahui berada pada jarak b cm dari X (Gambar 4). Dari data kedua ini dapat ditentukan dua kemungkinan posisi X (titik merah), yaitu di kedua titik perpotongan kedua lingkaran. Kemudian titik C diketahui berada pada jarak c cm (Gambar 5) dari posisi X. Dengan data terakhir ini bisa dengan tepat dipastikan letak X.



Gambar 3. Triangulasi satu refrensi

Gambar 4. Triangulasi dua refrensi



Gambar 5. Triangulasi Tiga refrensi

Namun masih terdapat kelemahan dalam konsep ini, yaitu lokasi tetap tidak bisa ditentukan walaupun sudah memiliki 3 titik refrensi jika ketiga titik. Konsep ini adalah inti utama teknologi GPS.

Google maps

Google Maps adalah dasar pemetaan web dan teknologi aplikasi layanan yang disediakan oleh Google, gratis (untuk non-komersial). Di dalam Google Maps menawarkan peta jalan, sebuah rute rencana untuk bepergian dengan berjalan kaki, mobil, atau angkutan umum dan pemantau bisnis di perkotaan untuk beberapa negara di sekitar dunia. Menurut salah satu pencipta (Las Rasmussen), Google Maps adalah suatu cara untuk mengorganisasikan informasi di dunia secara geografis. Seperti banyak aplikasi web Google lainya, Google Maps menggunakan JavaScript secara ekstensif.

Google maps menyediakan "API key" sebagai sarana untuk dapat menampilkan peta Google maps pada halaman web yang telah dibuat. *API key* tersebut disisipkan pada program halaman utama seperti pada Gambar 5 yang mana serangkaian kode tersebut
"key=AIzaSyD8NEDFqGvY-V6yu aUywiNv9ZId0iUKUo"
merupakan API key yang digunakan pada skripsi ini. Terdapat syarat untuk mendapatkan *API key* ini yaitu harus memiliki akun gmail setelah itu dapat menggunjungi https://developers.google.com/maps/licensing untuk mendapatkan API key tersebut.

Gambar 6. API Key Google Maps

Mikrokontroler ATMEGA164PA

Mikrokontroler ATMEGA164PA merupakan suatu chip yang berfungsi mengontrol suatu rangkaian elektronik agar dapat bekerja sesuai dengan fungsi yang diinginkan. IC ini dapat diprogram secara *In-System Programing* (ISP) dan mampu diprogram secara berulang-ulang sebanyak 10.000 kali tulis/hapus (*write*/*erase*). Gambar 7 merupakan bentuk fisik dari mikrokontroler ATMEGA164PA.

Gambar 7. ATMEGA164PA

Berikut adalah fitur-fitur yang dimiliki ATMEGA164PA :

1. Memiliki Tegangan kerja antara 1,8V-5,5V.
2. Memiliki 16 KB *Flash Memory* dan terdapat 8 kanal ADC dengan resolusi 10 bit.
3. Memiliki 2 kanal UART RX0, TX0 dan RX1, TX1.
4. Memiliki 512 Bytes EEPROM dan 1K Bytes *Internal SRAM*.
5. Memiliki *Input/Output* (I/O) sebanyak 32 pin yang terbagi menjadi 4 PORT, yaitu PORT A, PORT B, PORTC, dan PORTD.
6. Frekuensi maksimum 20 MHz.

Geofence

Untuk mengetahui kendaraan yang menyeleweng keluar dari area kerja yang telah ditentukan, maka diperlukan fitur geofencing. Geofencing (pembatasan lokasi) digunakan untuk menganalisa posisi kendaraan secara otomatis dan melaporkan kapan kendaraan keluar atau masuk area geofence yang sebelumya telah ditentukan oleh pemakai. Area geofence tersebut merupakan area virtual yang membatasi lokasi tertentu. Keluar atau masuknya area virtual ditentukan oleh LBS (*Location-based service)*. Data disimpan pada web server yang berfungsi sebagai *GPS Tracking Server*. Komputer pemantau akan melakukan koneksi ke alamat web server untuk dapat mematau posisi kendaraan bergerak.

Aplikasi dibangun menggunakan tampilan peta digital yang diambil dari Google Map. Peta Google Map tersebut kita program melalui API (Application Programming Interface) yang tersedia untuk menampilkanGambar 8 menunjukan contoh *geofence* yang dapat dibuat pada html 5 menggunakan google maps

Gambar 8. Contoh Geofence pada Maps

Modul GPS GSM/GPRS SIM908

SIM908EVB merupakan sebuah modul GPS dan modul GSM/GPRS yang dikemas dalam satu modul. Bentuk fisik sim908 dapat dilihat pada Gambar 9. Berikut adalah spesifikasi modul tersebut:

Gambar 9. Modul SIM908

GPRS:

1. Berbasis SIM908, modul GSM *Quad-Band* 850/900/18001900MHz
2. Mendukung kartu SIM dengan tegangan 1.8V & 3V.
3. Mendukung GPRS multi-slot kelas 10, fitur SIM Application toolkit, fitur Fax, dan protokol TCP/IP.
4. Tersedia 1 chanel ADC dengan range tegangan 0 – 2.4 V yang diwakili oleh nilai 0 -2400 (1 nilai = 0.001V)
5. *Baudrate* 4800, 9600, 19200, 38400, 57600, atau 115200 bps (*autobaud*), 8 data bit, 1 *stop* bit, *no parity*.
6. *Protocol* AT+COMMAND

GPS

1. Frekuensi L1, 1575.42 MHz C/A code
2. Penerima 42-channel
3. Konsumsi daya 77mA
4. *Cold start* 30s
5. *Hot start* 1s
6. *Baudrate* @4800 8N1

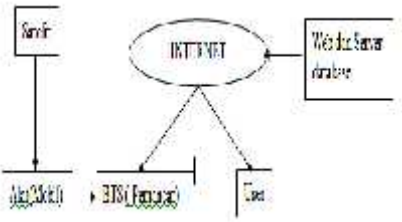
Protokol NMEA-0183

modul ini juga memiliki manajemen daya yang digunakan untuk mengelola baterai isi ulang internal. Baterai ini adalah kekuatan cadangan jika kekuatan eksternal dari baterai mobil dicabut dan cadangan di tertanam di dalam unit. Unit manajemen biaya termasuk kekuatan mengisi ulang penuh jika baterai cadangan rendah daya, mengisi baterai cadangan untuk jangka waktu tertentu ketika penuh, dan kemudian berhenti pengisian saat baterai cadangan benar-benar penuh.

Gambar 8 juga menunjukkan bahwa modul memiliki penerima GPS lengkap, sambungan data paket GPRS lengkap bersama dengan telepon GSM dengan koneksi kartu SIM eksternal untuk GSM dengan *Mini SIM*. GPS, GSM dan GPRS Unit dikendalikan oleh dua pelabuhan UART yang berbeda, satu UART untuk GPS dan UART lain untuk unit GSM / GPRS. Modul GSM adalah Quad-band dengan daya ke kartu SIM (1,8-3 Volt) kompatibel dengan penyedia di Indonesia. Serial port koneksi dukungan data rate 4800-115200 bit / detik (bps). Antena untuk GSM dan GPS modul yang terhubung melalui dua koneksi yang berbeda.

METODE PENELITIAN

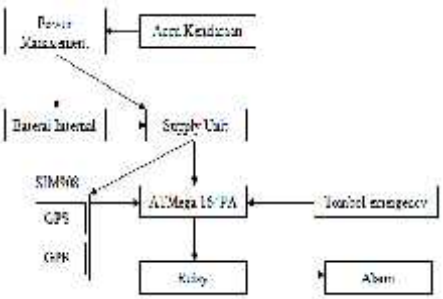
Pada pembuatan alat ini, terdapat dua bagian yang saling berhubungan seperti terlihat pada Gambar 10, yaitu perancangan pada sisi *device* dan perancangan pada sisi aplikasi *server*. Perancangan pada sisi *device* terdiri dari perancangan *hardware* dan *software.*pada sisi aplikasi *server* terdiri dari perancangan halaman web serta perancangan penyimpanan data pada *database.*



Gambar 10. Keseluruhan Sistem

Perancangan Alat Sisi *Device*

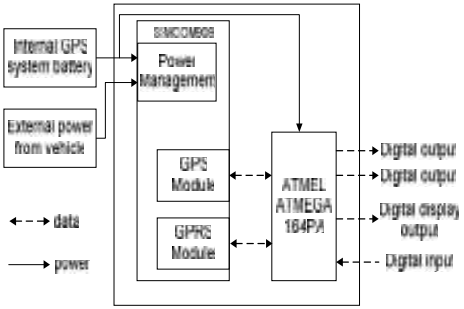
Diagram blok pada sisi hardware dapat dilihat pada Gambar 11. Modul GPS menentukan lokasi koordinat dengan bantuan beberapa satelit. Interface modul ini menggunakan komunikasi serial dengan protokol NMEA-0183. Data-data keluaran GPS tersebut berupa kalimat (*string*) yang tiap karakternya merupakan kode ASCII 8 bit. Data tersebut dikirimkan kepada mikrokontroler melalui komunikasi serial dengan baudrate sesuai dengan interface GPS device. Data yang didapat oleh mikrokontroler, diproses untuk memperoleh koordinat geografi (*latitude*/ lintang, *longitude*/ bujur), waktu dan kecepatan. Data tersebut dikirimkan ke *server* melalui perangkat GPRS yang dilakukan oleh modul GSM dengan menggunakan protocol HTTP. Pada proses ini, modul GSM juga dapat menerima perintah dari *server* untuk melakukan tugas tertentu seperti menyalakan alarm kendaraan. Jika tombol emergency ditekan lebih dari 2 kali dalam waktu 1 detik, maka mikrokontroler akan memerintahkan perangkat GPRS agar mengirimkan pesan darurat ke *server* untuk memberitahu pemantau tentang keadaan darurat.



Gambar 11. Diagram Blok sisi *Hardware*

Baterai internal berfungsi sebagai catu daya cadangan ketika catu daya utama (*accu*) dilepas. Blok *supply unit* berfungsi untuk menurunkan tegangan dan mensuplai tegangan pada sistem. *Power management* dirancang agar dapat berfungsi sebagai indikator terpasangnya *accu* dan dapat mengetahui kapasitas dari baterai *internal* tersebut. Disamping itu dapat juga berfungsi sebagai *charger internal* yang dapat berfungsi untuk mengisi baterai secara otomatis ketika baterai dalam keadaan *low level*. Mikrokontroler membaca kapasitas baterai *internal* dengan menggunakan rangkaian pembagi tegangan yang kemudian diproses oleh ADC (*Analog to Digital Converter*) yang terdapat pada mikrokontroler.

Desain sistem keamanan kendaraan dan sistem manajemen dalam diagram blok yang ditunjukkan pada Gambar 12. Ada modul GPS dan modul GPRS terintegrasi dalam satu modul yang lebih besar di SIMCOM 908. Sistem ini juga memiliki modul manajemen daya untuk mengontrol pengisian baterai lithium cadangan yang akan digunakan untuk daya sistem jika ada kegagalan daya sistem listrik utama. Masing-masing modul (GPS dan GPRS) memiliki seri input / output UART (Universal Asynchronous Receiver Transmitter) di mana perangkat akan mengirim atau menerima data. Serial input / output terhubung ke ATMEGA164PA mikrokontroler yang memiliki dua masukan serial / output. Semua modul yang didukung menggunakan listrik yang sama yang 3.3V yang dapat diperoleh dari daya eksternal atau dari baterai lithium internal.



Gambar 12. Block diagram of vehicle monitoring/ management system using GPS via GPRS network.

Sistem ini memiliki satu baris untuk input digital digunakan sebagai tombol darurat. Dalam hal ini jika pengemudi kendaraan menemukan sebuah kondisi darurat seperti pembajakan mobil (mengambil alih kendaraan dengan kekerasan) atau mengambil muatan dari truk dengan kekerasan, pengemudi cukup menekan tombol darurat selama lebih dari tiga detik. Setelah periode tiga kedua, modem GPRS akan mengirimkan sinyal darurat ke operator di Internet untuk peringatan sehingga operator dapat mengambil tindakan yang tepat.

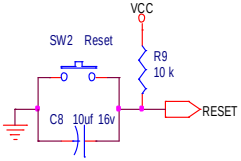
Ada dua saluran output digital dapat dimanfaatkan untuk mengontrol kendaraan dengan menggunakan relay switch. Satu output dari perangkat akan terbuka dan saklar lain akan tertutup. Penggunaan ini dapat digunakan untuk menyimpan data seperti untuk menonaktifkan kendaraan dengan memotong pasokan bahan bakar (mesin diesel) atau dengan memotong pasokan daya listrik ke mesin (mesin bensin). Output digital lainnya dapat digunakan sebagai kontrol sekunder untuk kendaraan seperti AC atau kenyamanan lain di dalam kendaraan dan juga dengan menyalakan alarm eksternal. Digital output port display digunakan dalam hubungannya dengan layar LCD yang digunakan untuk fitur baru ditambahkan ke sistem kemudian.

- **Perancangan Hardware Minimum System ATMega164-PA**

Pada sisi alat (hardware), alat ini menggunakan mikrokontroler ATMega 164PA sebagai pengontrol utama alat tersebut. ATMega164PA dipilih karena mempunyai fitur khusus UART 2 kanal sehubung dengan interface dengan modul SIM908. Selain itu pada mikrokontroler ini juga memiliki fitur ADC internal yang nantinya dapat digunakan untuk mengukur tegangan baterai. Agar mikrokontroler ini dapat

bekerja, maka diperlukan rangkaian pendukung diantaranya rangkaian reset dan rangkaian external crystal oscillator.

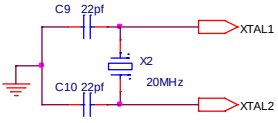
Rangkaian reset berfungsi untuk me-reset program counter dan mengembalikan system dalam kondisi awal. Reset dilakukan secara manual yaitu dengan menekan switch yang akan memberikan logika low pada mikrokontroler. Rangkaian reset dapat dilihat pada Gambar 13.



Gambar 13. Rangkaian Reset

Cara kerja rangkaian reset ini, jika switch SW2 tidak ditekan maka arus yang mengalir dari VCC ke resistor pull-up R9 akan diteruskan kepada pin RESET sehingga mendapatkan sinyal high dan mikrokontroler akan aktif bekerja. Sebaliknya apabila switch SW2 ditekan maka arus yang mengalir dari VCC ke resistor pull-up R9 akan akan langsung mengalir menuju ground sehingga pin RESET mendapatkan sinyal low. Sinyal low ini digunakan oleh mikrokontroler untuk me-reset program counter kembali pada kondisi awal. Kapasitor C8 berfungsi agar tidak terjadi bouncing pada saat switch SW2 ditekan maupun dilepas.

Rangkaian *external crystal oscillator* berfungsi untuk membangkitkan *clock* yang nantinya digunakan oleh mikrokontroler sebagai acuan untuk menjalankan instruksi. Pin XTAL1 dan XTAL2 pada mikokontroler dihubungkan dengan rangkaian seperti pada Gambar 14. Rangkaian oscillator tersebut terdiri dari sebuah kapasitor Kristal dengan frekuensi 20 MHz yang dirangkai dengan 2 buah kapasitor (C9 dan C10) yang nilainya sama besar 22 pF. Besar nilai kapasitor (C9 dan C10) merupakan nilai yang telah direkomendasikan oleh pihak produsen yang tertulis pada datasheet.



Gambar 14. Rangkaian External Crystal Oscillator

Eksekusi instruksi dilakukan dalam sebuah clock cycle, maka satu machine cycle dikerjakan selama satu periode *oscillator*. Persamaan matematis untuk perhitungan waktu satu machine cycle adalah sebagai berikut.

$$1\ mac\ ine\ cycle = 1\ x\ periode\ oscillator$$

Dan $T = 1/f$, maka:

$$Tmac\ ine\ cycle = 1 / f_{crystal}$$

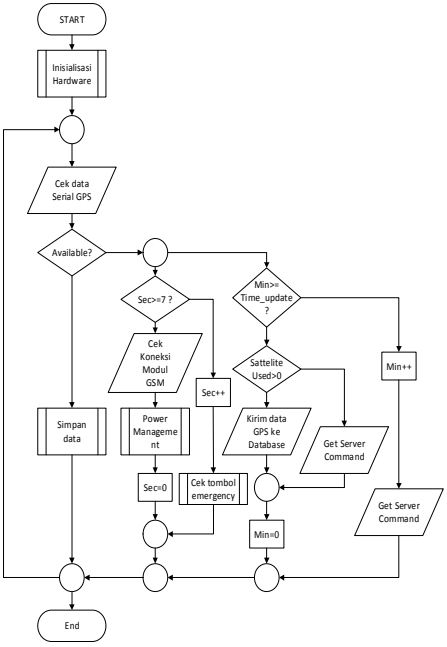
Berdasarkan persamaanmaka diperoleh nilai $T_{machine\ cycle}$ Sebagai berikut:

$$T_{machine\ cycle} = \frac{1}{18.432000\ MHz} = 0.0542\ \mu s$$

Maka mikrokontroler mampu melakukan 1 kali instruksi selama 0.0542 us.

- **Perancangan Software**

Perancangan perangkat lunak pada alat ini bertujuan agar mikrokontroler dapat menjalankan tugasnya dan memperoleh kerja yang maksimal. Oleh karena itu perlu diperhatikan perancangan perangkat lunak serta algoritma *scheduler* agar mikrokontroler dapat mengirim dan menerima data dari modul SIM908 dengan sangat baik. Tugas utama dari mikrokontroler adalah mengambil data GPS dari modul SIM908 serta mengirimkan data GPS tersebut ke database menggunakan komunikasi GPRS secara berkala.



Gambar 15. Flowchart Pada Alat

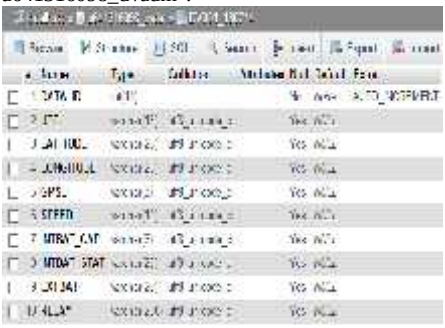
Algoritma keseluruhan kerja mikrokontroler dapat dilihat pada Gambar 15. Mikrokontroler melakukan inisialisasi sitem diantaranya inisialisasi PORT input/ output, pengaturan baudrate GPS dan GSM, serta pengaturan APN, User name, password yang nantinya dibutuhkan untuk melakukan komunikasi menggunakan GPRS. Pengaturan PORT meliputi pengaturan PIN input/ output yang digunakan untuk menyalakan modul SIM908, pengaturan untuk pembacaan nilai ADC tegangan baterai, serta pengaturan PIN untuk menyalakan relay. Pengaturan baudrate dilakukan agar mikrokontroler dapat berkomunikasi dengan modul baik GSM maupun GPS. Pengaturan komunikasi serial untuk GPS menggunakan baudrate 115200 bps agar waktu yang dibutuhkan mikrokontroler untuk mengambil data lebih sedikit. Baudrate untuk modul GSM bersifat autobaud sehingga modul tersebut dapat menyesuaikan baudrate yang dimiliki mikrokontroler. Pada mikrokontroler baudrate yang dipilih adalah 115200 bps sehingga dapat dikatakan kedua perangkat tersebut berkomunikasi menggunakan baudrate 115200 bps. Baudrate tersebut dipilih guna mengantisipasi lama penggunaan waktu yang dibutuhkan untuk mengirim dan menerima data dari modul GSM. Pengaturan APN, Username, dan Pasword mengikuti pengaturan kartu provider yang digunakan namun pada skripsi ini menggunakan provider XL AXIATA sehingga pengaturan APN, Username, dan Pasword menjadi "www.xlgprs.net", "xlgprs", dan "proxl".

Mikrokontroler akan membaca data GPS melalui komunikasi serial menggunakan protocol NMEA0183. Jika protocol NMEA0183 dengan header data $GPGGA dan $GPVTG tersedia maka Mikrokontroler akan mengambil dan menyimpan data waktu (UTC), koordinat, kecepatan, serta satelit yang sedang digunakan. Seperti yang dijelaskan pada bab II protocol NMEA0183 dengan header $GPGGA hanya menyimpan data diantaranya waktu, koordinat, serta satelit yang digunakan sedangkan untuk data kecepatan memerlukan protocol NMEA0183 dengan header $GPVTG. Pada saat data GPS tidak tersedia, maka mikrokontroler melakukan koneksi dengan modul GSM guna memeriksa konektivitas dengan modul tersebut. Selain itu mikrokontroler juga akan memeriksa kapasitas baterai internal serta mengisi baterai tersebut jika kapasitas baterai lebih kecil dari 70%. Tombol emergency juga akan dipantau oleh mikrokontroler sehingga apabila tombol tersebut ditekan, maka mikrokontroler akan mengirimkan pesan darurat ke *server*.

Pengiriman data menuju database dilakukan secara berkala dengan satuan interval waktu menit. Tiap menit mikrokontroler akan memerintahkan modul SIM908 agar terhubung dengan *server* melalui komunikasi GPRS. Komunikasi tersebut bertujuan untuk mengambil perintah dari *server* yaitu menyalakan atau mematikan relay yang terhubung dengan alarm. Apabila menit tick sudah melebihi interval waktu update maka mikrokontroler akan memeriksa apakah data GPS tersebut valid atau tidak. Untuk memeriksa kebenaran data koordinat dapat diperiksa melalui jumlah satelit yang digunakan. Secara teori triangulasi yang mana dijelaskan pada bab II, posisi GPS dapat ditentukan jika menggunakan lebih dari sama dengan 2 satelit. Apabila data GPS telah valid maka mikrokontroler akan mengirimkan data tersebut ke database namun apabila data GPS masih belum valid, maka mikrokontroler akan memerintah modul GSM untuk mengambil perintah dari *server* saja.

Perancangan Database

Perancangan ini bertujuan agar data dapat disimpan sesuai dengan kebutuhan. Terdapat data yang ingin dilihat berdasarkan hari (data GPS) dan adapula data yang dapat di-update sewaktu-waktu (data administrasi device). Dengan demikian dibuat database yang berbeda antara data GPS dengan data yang menyimpan data administrasi device. Data GPS disimpan dalam database dengan nama "u641316098_dvdat" sedangkan untuk data administrasi disimpan dalam database "u641316098_dvadm".



Gambar 16. Gambar Perancangan Database Data Device

Struktur data untuk penyimpanan data GPS dapat dilihat pada Gambar 16. Terdapat 10 kolom untuk penyimpanan data GPS diantaranya DATA_ID, UTC, LATITUDE, LONGITUDE, GPSU, SPEED, INBAT_CAP, INBAT_STAT, EXTBAT dan RELAY. Kolom dengan nama DATA_ID mempunyai properti tipe data int dengan panjang digit sampai 11 dan mempunyai fitur auto increment.

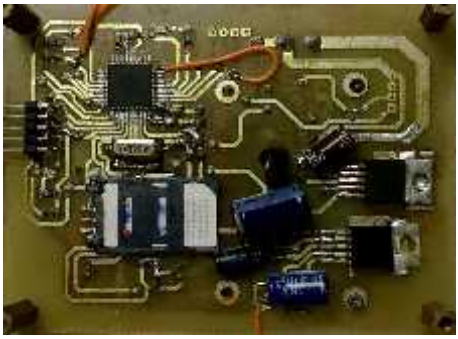
Kolom ini digunakan sebagai penanda data GPS yang pertama kali masuk dan yang paling terakhir masuk sehingga data mudah diurutkan serta lokasi mobil dengan data terakhir mudah diketahui. Data-data yang dikirm oleh device disimpan pada kolom dengan label UTC sampai dengan RELAY. Berikut penjelasan masing-masing kolom :
UTC : waktu device
LATITUDE dan LONGITUDE : koordinat geografi bumi
GPSU : jumlah satelit yang digunakan
INTBAT_CAP : kapasitas baterai internal
INTBAT_STAT : status baterai internal (charging)
EXBAT : eksternal baterai (terpasang/tidak)
RELAY : relay sedang aktif atau mati

HASIL PENELITIAN DAN PEMBAHASAN

Sistem pelacakan kendaraan menggunakan GPS dan GPRS untuk koneksi data yang telah dirancang dan dibangun menggunakan semua subsistem yang disebutkan di atas. Sistem ini menggunakan papan ganda bersisi untuk menghemat ruang dengan SIM908 di satu sisi bersama dengan beberapa bagian eksternal seperti output driver dan regulator. Bagian atas sistem dengan modul SIM908 ditunjukkan pada Gambar 17. Tampil juga dalam gambar ini pigtail untuk antena, bagian atas adalah konektor antena GPS sedangkan koneksi bawah adalah untuk antena GPRS.

Gambar 17. Sisi atas alat

Gambar 18 adalah sisi sebaliknya dari papan sirkuit. Di sisi ini ada mikroprosesor dalam bentuk *SMT quadpak* bersama-sama dengan kristal dan pemegang kartu SIM, dan dua regulator dan di sisi kiri papan beserta adanya 8-pin header untuk pemrograman mikroprosesor melalui port ISP dari mikrokontroler.



Gambar 18. Sisi bawah alat

Informasi lokasi kendaraan yang diperoleh dari GPS disimpan dalam file database (MySQL) dengan data yang ditunjukkan pada Tabel 1. Selain data geografis, waktu, kondisi baterai dan status kendaraan juga disimpan dalam database dan dapat dipanggil dari web dengan mengklik penanda kendaraan. Database dan juga antarmuka web dapat diakses melalui http://www.gpsfence.web.id.

Table 1. Parameters padadatabase yang di dapat dari informasi kendaraan

Field	Parameter saved
UTC	*Time data are taken based on the Universal Time Coordinated time zone*
Longitude	*Longitude coordinate of the vehicle in signed degree format (DDD.dddd) from -180 East to +180 West*
Latitude	*Latitude coordinate of the vehicle in signed degree (DD.dddd) from -90 North to +90 South*
GPSU	*Number of satellites visible to the GPS receiver and used to calculate the position*
Speed	*Speed of the vehicle in kilometer per hour*
Internal battery	*Internal battery capacity in percent*
External power	*Indicator of external battery connection*
Relay 1	*ON/ OFF condition (usually for engine, normally open)*
Relay 2	*On/ OFF condition usually for external alarm or others (normally closed)*
Emergency	*On/ OFF condition of emergency button, normally off/ open*

Transmisi data dari GPS ke database dilakukan melalui koneksi nirkabel GPRS yang telah dibahas sebelumnya. Koneksi data telah diuji dengan semua jaringan GSM prabayar di Indonesia dan telah terbukti untuk bekerja dengan baik. Untuk pekerjaan ini, sambungan data menggunakan Telkomsel penyedia GSM dengan Access Point Name (APN) kode internal ke mikrokontroler. Kode untuk pekerjaan telah dibahas di tempat lain.

Halaman web yang tercantum pada alamat di atas adalah HTML5 mampu dan oleh karena itu dapat menampung geofence dan halaman pertama yang ditampilkan adalah sama dengan Gambar 19.

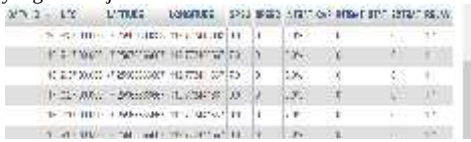
Gambar 19. Kendaraan masih di dalam *Geofence*

Ketika lokasi kendaraan dalam geofence yang (menandai dengan warna abu-abu dari quad-angle) penanda ditampilkan merah dan ketika lokasi di luar pagar, maka penanda ditampilkan sebagai warna biru. Cloking di pasar akan menunjukkan bahwa lokasi dalam atau di luar geofence tersebut. Dalam gambar ini kendaraan dipantau menggunakan GPS dan terbukti dalam pagar geo (empat lokasi) dan dua lokasi di luar pagar. Halaman ini juga akan menunjukkan bahwa kendaraan berada di luar pagar ketika mouse diklik. Ketika kendaraan dalam pagar diklik, maka jendela akan menunjukkan bahwa kendaraan dalam pagar seperti yang ditunjukkan pada Gambar 20

Gambar 20. Kendaraan sudah di luar *Geofence*

Pengujian alat secara keseluruhan bertujuan untuk memastikan apakah alat dapat bekerja dengan baik sesuai dengan rancangan yang telah dibuat. Alat diletakkan pada mobil dan dihidupkan. Antena GPS,GSM dan catu daya eksternal dihubungkan agar alat dapat memperoleh sinyal serta catu daya internal alat dapat terisi ketika catu daya eksternal tersedia.

Data yang dikirimkan oleh alat akan disimpan pada database yang telah dirancang. Hasil proses penyimpanan dapat dilihat pada Gambar 21. dari gambar tersebut terlihat bahwa alat dapat berhasil memperoleh sinyal dan mengirimkan data tersebut kedalam *database*. Pada *database* tersebut dapat dilihat koordinat, satelit yang digunakan, status *internal* baterai serta kapasitas, dan hubungan dengan catudaya eksternal sesuai dengan DATA_ID yang menunjukan data terbaru.



Gambar 21. Data yang Dikirim Oleh Device

Sebuah kendaraan yang dilengkapi dengan penerima GPS dapat memperoleh nya geografis berkoordinasi dengan mudah dan akurat dan data dapat dikirim ke server di Internet melalui jaringan nirkabel GPRS. Data kemudian dapat ditampilkan / *overlay* menggunakan peta untuk menunjukkan lokasi kendaraan dan daripada yang dapat ditingkatkan lebih lanjut dengan menampilkan pagar virtual yang disebut geofence. Dengan geofence ini, kendaraan dapat tertutup berada di daerah tertentu dari operasi dan ketika pindah luar posisi tertutup yang ditentukan operator dapat disiagakan. Dengan sistem peringatan ini, kendaraan dapat lebih dijamin terhadap setiap perbuatan yang salah atau niat buruk lainnya dan dengan demikian keamanan kendaraan ditingkatkan lebih jauh.

Gambar 22. Kondisi Marker setelah perubahan *Geofence*

KESIMPULAN

Dari hasil perancangan, pembuatan dan pengujian yang telah dilakukan, maka dapat diambil kesimpulan sebagai berikut :

- Mikrokontroler dapat mengambil data GPS pada modul SIM908 menggunakan komunikasi serial TTL 115200b/s 8N1. Serta Mikrokontroler dapat terhubung dengan *server* menggunakan komunikasi GPRS melalui perantara modul GSM SIM908.
- Alat dapat bekerja menggunakan baterai internal ketika catu daya di lepas.
- Lokasi koordinat dapat dilihat pada halaman web yang telah dibuat dalam bentuk peta dan marker.
- Halaman web dapat menampilkan Geofence dan mem.
- Jangkauan sinyal dapat mempengaruhi terkirimnya data ke server.

DAFTAR PUSTAKA

[1] Pranjoto, H., Agustine, L, Susilo, Y.S., Tehuayo, R., "GPS Based Vehicle Tracking over GPRS for Fleet Management and Passenger/ Payload/ Vehicle Security", ARPN Journal of Engineering and Applied Sciences, Vol. 9, No. 11.

[2] Grewal, M.S., Weill, L.R., Andrews, A.P., Global Positioning System, Inertial Navigation and Integration, Wiley, New York, 2001.

[3] Halonen, T., Romero, J., Melero, J. (eds), GSM, GPRS, and edge performance: evolution towards 3G/UMTS 2nd Ed., Wiley, New York, 2003.

[4] Kaplan, E., Hegarty, C., Understanding GPS: Principles and Applications 2nd Ed., Artech House, Norwood, Massachussettes, 2006.

[5] Kingsley-Hughes, K., Hacking GPS, Wiley, New York, 2005.

[6] Maral, G., Bousquet, M., Satellite Communications Systems, Systems, Techniques and Technology 5th Ed., Wiley, West Sussex, 2009.

[7] Meyer, E., Ahmed, I., Benefit-Cost Assessment of Automatic Vehicle Location (AVL) in Highway Maintenance, Proceedings of the 2003 Mid-Continent Transportation Research Symposium, Ames, Iowa, August 2003.

[8] Peng, Z. R., Beimborn, E.A., Octania, S., Zygowics, R.J., Evaluation of the Benefits of Automated Vehicle Location Systems in Small and Medium Sized Transit Agencies, Center For Urban Transportation Studies, Milwaukee, Wisconsin, 1999.

[9] Portillo, D., Automated Vehicle Location using Global Positioning Systems for First Responders, Institute for Information Technology Applications Technical Report Series, Colorado, 2008.

[10] National Marine Electronic Association, NMEA 0183 Version 4.10 Electronic, National Marine Electronic Association, Severna Park, Maryland, 2012.

[11] Sirf Technology, NMEA Reference Manual, SIRF Techology, San Jose, 2007.

[12] Seurre, E., Savelli, P. Pietri, J-P., GPRS for Mobile Internet, Artech House, Norwood, Massachussettes, 2003.

[13] Eberspächer , J., Vögel, HJ., Bettstetter, C., Hartmann, C., GSM – Architecture, Protocols and Services 3rd Ed., Wiley, Stuttgart, 2009.

[14] European Telecomunications Standard Institure, Technical Specification AT command set for User Equipment 3GPP TS27.007 version 11.5.0 Release 11 (2013-01), Januari 2013.

[15] Sim Tech, SIM 908 Hardware Design V. 2.00, 2012.