



Perancangan peralatan *data logger* pada truk angkutan barang

Tampilan Data Perekaman
Data Logger



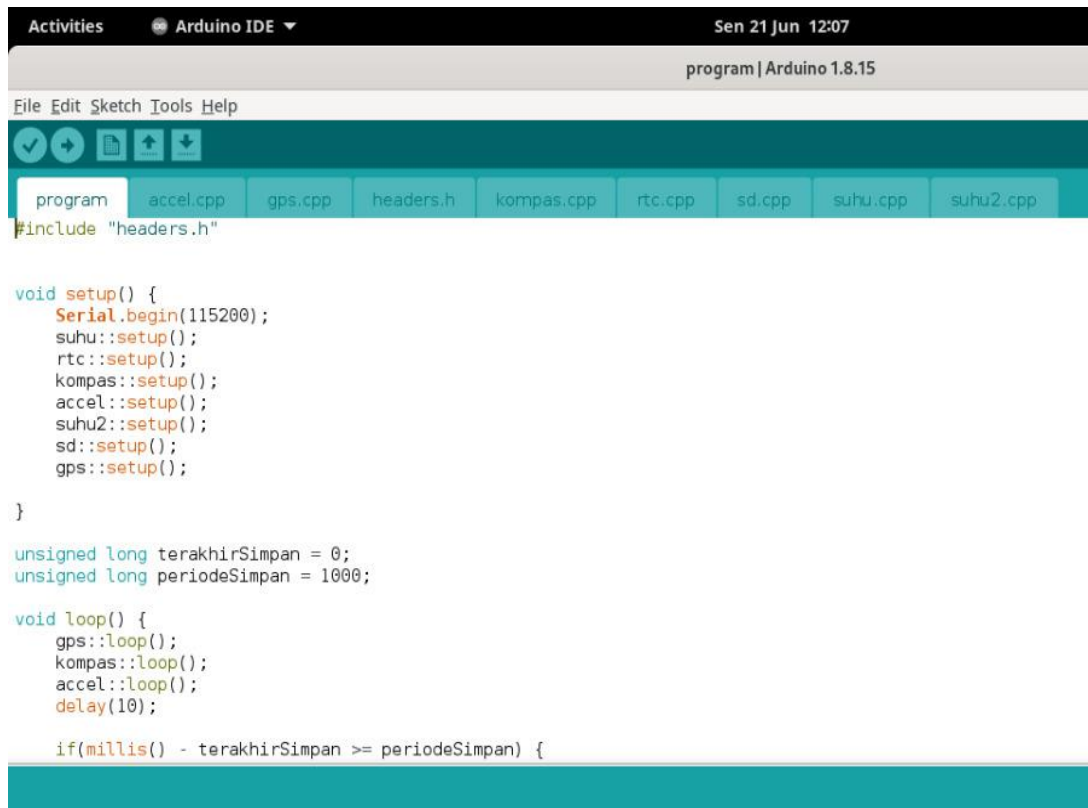
NO	RTC	Sensor GPS			Compass	Akselero Gyroscope			Suhu Luar Kabin	Suhu Dalam Kabin	Suhu Ban A	Suhu Ban B	Suhu Ban C	Suhu Ban D	Tekanan Ban A	Tekanan Ban B	Tekanan Ban C	Tekanan Ban D	Suhu Rem C	Suhu Rem D	Suhu Rem B	Suhu Mesin	Suhu Rem A
		Lat	Long	v		X	Y	Z															
1	7/19/2021 19:35	-6.29	107.07	0.13	47.00	0.09	-0.02	0.98	29.38	26.88	29.63	29.13	28.38	28.13	49.68	38.05	48.77	34.17	28.99	29.09	31.13	50.45	29.45
2	7/19/2021 19:35	-6.29	107.07	0.13	47.00	0.03	-0.05	1.11	29.38	26.88	29.56	29.13	28.44	28.13	49.68	38.05	48.77	34.17	28.99	29.15	31.25	50.49	29.55
3	7/19/2021 19:35	-6.29	107.07	0.13	47.00	-0.01	-0.04	1.15	29.38	26.94	29.63	29.06	28.44	28.13	49.68	38.05	48.77	34.17	29.03	29.09	31.29	50.49	29.49
4	7/19/2021 19:35	-6.29	107.07	0.20	47.00	0.11	-0.04	0.93	29.38	26.88	29.56	29.13	28.38	28.13	49.68	38.11	48.77	34.17	29.11	29.11	31.33	50.55	29.51
5	7/19/2021 19:35	-6.29	107.07	0.30	47.00	0.12	-0.04	0.93	29.38	26.94	29.56	29.13	28.38	28.13	49.68	38.11	48.77	34.17	29.19	29.15	31.29	50.61	29.45
6	7/19/2021 19:35	-6.29	107.07	0.30	47.00	-0.06	-0.01	1.16	29.38	26.94	29.63	29.13	28.38	28.13	49.68	38.11	48.77	34.17	29.11	29.09	31.39	50.51	29.45
7	7/19/2021 19:35	-6.29	107.07	0.30	47.00	0.08	-0.05	1.02	29.38	26.88	29.63	29.13	28.38	28.13	49.68	38.11	48.77	34.17	29.09	29.17	31.21	50.63	29.49
8	7/19/2021 19:35	-6.29	107.07	0.11	47.00	0.05	-0.05	1.04	29.31	26.88	29.56	29.13	28.44	28.13	49.68	38.05	48.77	34.11	29.17	29.15	31.25	50.79	29.49
9	7/19/2021 19:36	-6.29	107.07	0.26	46.00	0.12	-0.03	0.96	29.38	26.94	29.56	29.13	28.38	28.13	49.68	38.11	48.77	34.17	29.15	28.99	31.19	50.73	29.45
10	7/19/2021 19:36	-6.29	107.07	0.26	47.00	0.13	-0.05	0.94	29.38	26.94	29.56	29.13	28.44	28.13	49.68	38.11	48.77	34.24	29.17	29.17	31.27	50.75	29.51
11	7/19/2021 19:36	-6.29	107.07	0.26	47.00	-0.02	-0.01	1.16	29.31	26.88	29.56	29.13	28.44	28.13	49.68	38.11	48.77	34.24	29.17	29.17	31.21	50.91	29.49
12	7/19/2021 19:36	-6.29	107.07	0.09	46.00	0.09	-0.05	1.02	29.31	26.94	29.56	29.13	28.44	28.13	49.68	37.98	48.77	34.11	29.11	29.11	31.19	50.77	29.55

**FOTO DOKUMENTASI UJI COBA DI LINGKUNGAN BALAI PENGUJIAN
LAIK JALAN DAN SERTIFIKASI KENDARAAN BERMOTOR (BPLJ-SKB)
BEKASI**



**FOTO DOKUMENTASI UJI COBA DI LINGKUNGAN BALAI PENGUJIAN
LAIK JALAN DAN SERTIFIKASI KENDARAAN BERMOTOR (BPLJ-SKB)
BEKASI (2)**





The screenshot shows the Arduino IDE interface with the 'program' tab selected. The code includes headers for various libraries and defines setup and loop functions. The libraries used are Serial, rtc, kompas, accel, suhu, suhu2, sd, and gps.

```
#include "headers.h"

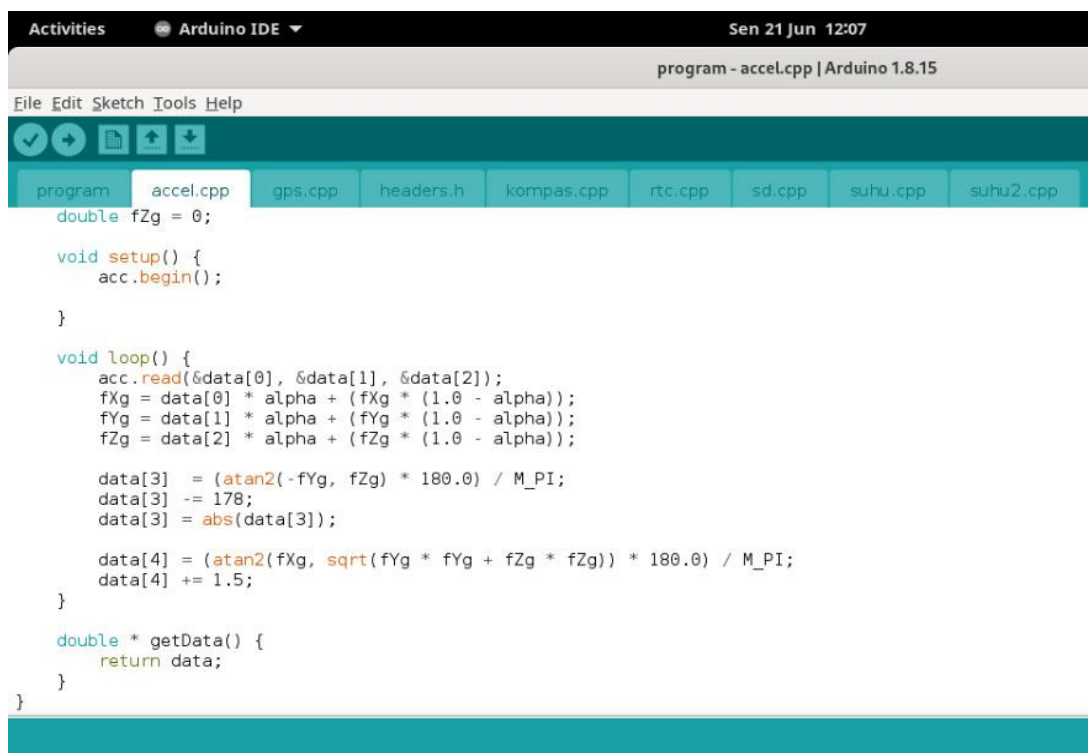
void setup() {
  Serial.begin(115200);
  suhu::setup();
  rtc::setup();
  kompas::setup();
  accel::setup();
  suhu2::setup();
  sd::setup();
  gps::setup();
}

unsigned long terakhirSimpan = 0;
unsigned long periodeSimpan = 1000;

void loop() {
  gps::loop();
  kompas::loop();
  accel::loop();
  delay(10);

  if(millis() - terakhirSimpan >= periodeSimpan) {
```

Pemrograman Keseluruhan sensor



The screenshot shows the Arduino IDE interface with the 'program - accel.cpp' tab selected. The code defines a variable fZg, initializes the accelerometer, and implements a loop function that reads data from the accelerometer and calculates the angle of inclination.

```
double fZg = 0;

void setup() {
  acc.begin();
}

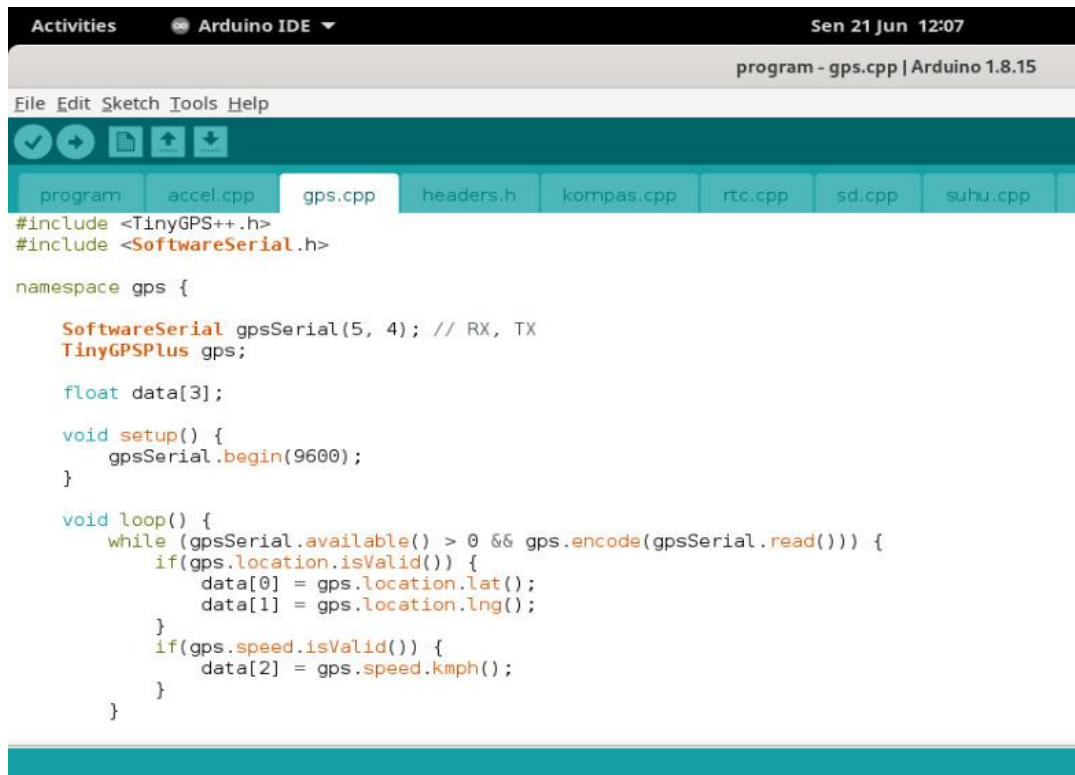
void loop() {
  acc.read(&data[0], &data[1], &data[2]);
  fXg = data[0] * alpha + (fXg * (1.0 - alpha));
  fYg = data[1] * alpha + (fYg * (1.0 - alpha));
  fZg = data[2] * alpha + (fZg * (1.0 - alpha));

  data[3] = (atan2(-fYg, fZg) * 180.0) / M_PI;
  data[3] -= 178;
  data[3] = abs(data[3]);

  data[4] = (atan2(fXg, sqrt(fYg * fYg + fZg * fZg)) * 180.0) / M_PI;
  data[4] += 1.5;
}

double * getData() {
  return data;
}
```

Pemrograman Accelerometer gyroscope

The screenshot shows the Arduino IDE interface with the 'gps.cpp' file open. The title bar indicates 'Sen 21 Jun 12:07' and the file name 'program - gps.cpp | Arduino 1.8.15'. The menu bar includes 'File', 'Edit', 'Sketch', 'Tools', and 'Help'. The toolbar contains icons for saving, running, and other IDE functions. The file explorer shows several files: 'program', 'accel.cpp', 'gps.cpp' (selected), 'headers.h', 'kompas.cpp', 'rtc.cpp', 'sd.cpp', and 'suhu.cpp'. The code in 'gps.cpp' includes 'TinyGPS++.h' and 'SoftwareSerial.h'. It defines a 'gps' namespace with a 'SoftwareSerial' object 'gpsSerial' at pins 5 and 4, a 'TinyGPSPlus' object 'gps', a float array 'data[3]', and functions 'setup()' and 'loop()'. The 'loop()' function checks for available data and updates 'data' with location and speed information.

```
File Edit Sketch Tools Help
program accel.cpp gps.cpp headers.h kompas.cpp rtc.cpp sd.cpp suhu.cpp
#include <TinyGPS++.h>
#include <SoftwareSerial.h>

namespace gps {

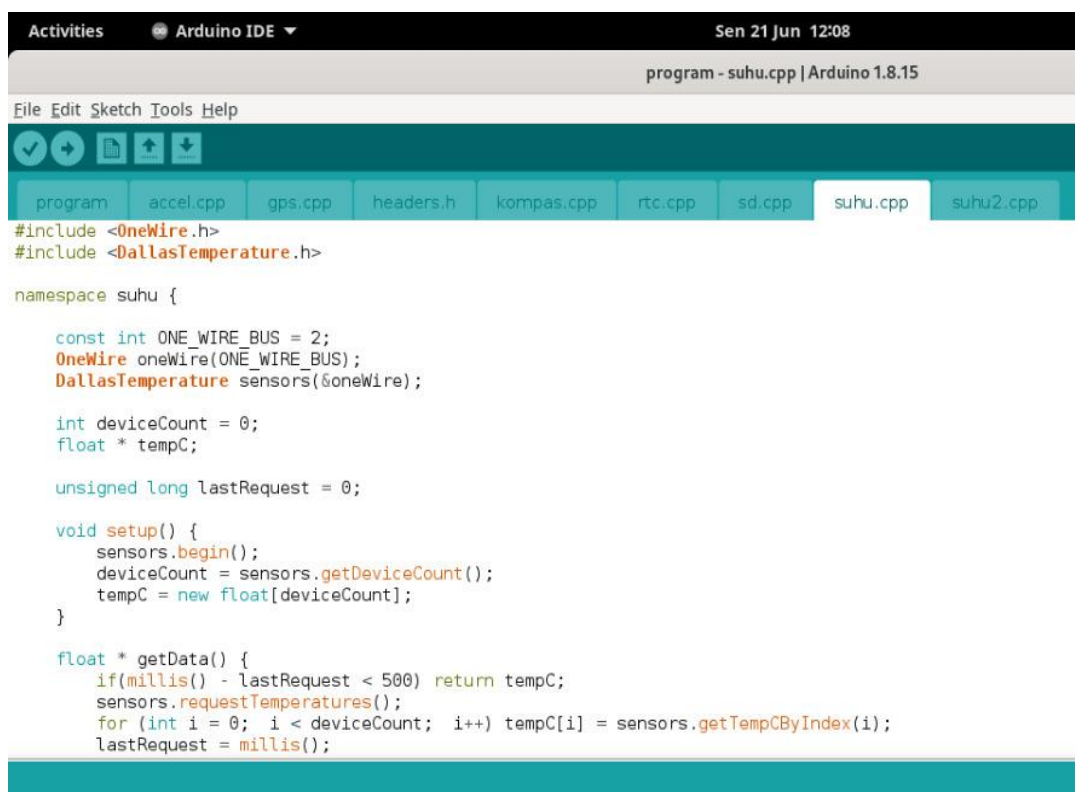
    SoftwareSerial gpsSerial(5, 4); // RX, TX
    TinyGPSPlus gps;

    float data[3];

    void setup() {
        gpsSerial.begin(9600);
    }

    void loop() {
        while (gpsSerial.available() > 0 && gps.encode(gpsSerial.read())) {
            if(gps.location.isValid()) {
                data[0] = gps.location.lat();
                data[1] = gps.location.lng();
            }
            if(gps.speed.isValid()) {
                data[2] = gps.speed.kmph();
            }
        }
    }
}
```

Pemrograman GPS

The screenshot shows the Arduino IDE interface with the 'suhu.cpp' file open. The title bar indicates 'Sen 21 Jun 12:08' and the file name 'program - suhu.cpp | Arduino 1.8.15'. The menu bar includes 'File', 'Edit', 'Sketch', 'Tools', and 'Help'. The toolbar contains icons for saving, running, and other IDE functions. The file explorer shows several files: 'program', 'accel.cpp', 'gps.cpp', 'headers.h', 'kompas.cpp', 'rtc.cpp', 'sd.cpp', 'suhu.cpp' (selected), and 'suhu2.cpp'. The code in 'suhu.cpp' includes 'OneWire.h' and 'DallasTemperature.h'. It defines a 'suhu' namespace with a 'const int' 'ONE_WIRE_BUS' set to 2, a 'OneWire' object 'oneWire', a 'DallasTemperature' object 'sensors', an 'int' 'deviceCount', a 'float' array 'tempC', and an 'unsigned long' 'lastRequest'. It also defines 'setup()' and 'getData()' functions. The 'setup()' function initializes the sensors and allocates memory for 'tempC'. The 'getData()' function returns 'tempC' if a request has been made within the last 500ms, otherwise it requests temperatures from the sensors and updates 'tempC' and 'lastRequest'.

```
Activities Arduino IDE Sen 21 Jun 12:08
program - suhu.cpp | Arduino 1.8.15
File Edit Sketch Tools Help
program accel.cpp gps.cpp headers.h kompas.cpp rtc.cpp sd.cpp suhu.cpp suhu2.cpp
#include <OneWire.h>
#include <DallasTemperature.h>

namespace suhu {

    const int ONE_WIRE_BUS = 2;
    OneWire oneWire(ONE_WIRE_BUS);
    DallasTemperature sensors(&oneWire);

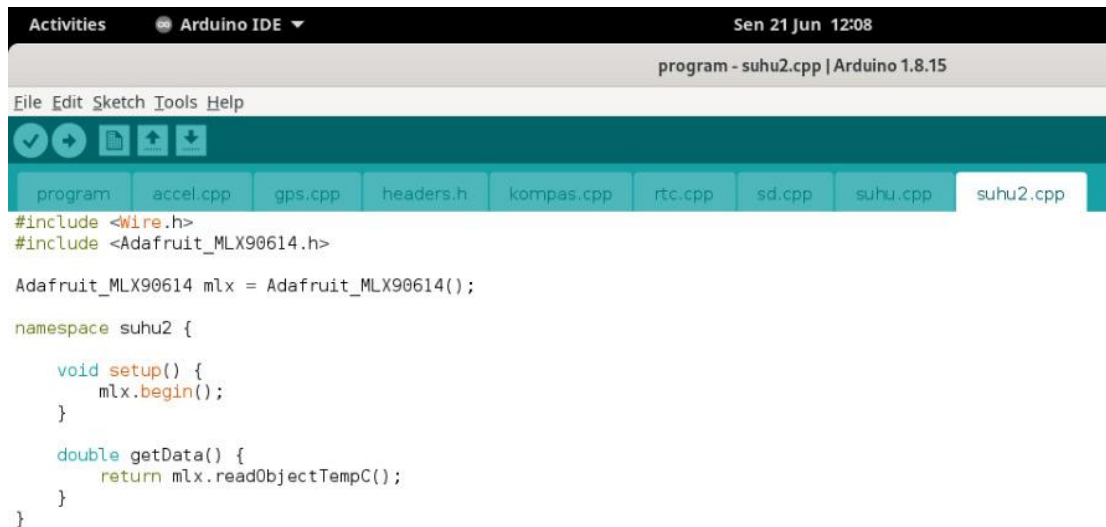
    int deviceCount = 0;
    float * tempC;

    unsigned long lastRequest = 0;

    void setup() {
        sensors.begin();
        deviceCount = sensors.getDeviceCount();
        tempC = new float[deviceCount];
    }

    float * getData() {
        if(millis() - lastRequest < 500) return tempC;
        sensors.requestTemperatures();
        for (int i = 0; i < deviceCount; i++) tempC[i] = sensors.getTempCByIndex(i);
        lastRequest = millis();
    }
}
```

Pemrograman Sensor Suhu (kabel) DS1820B



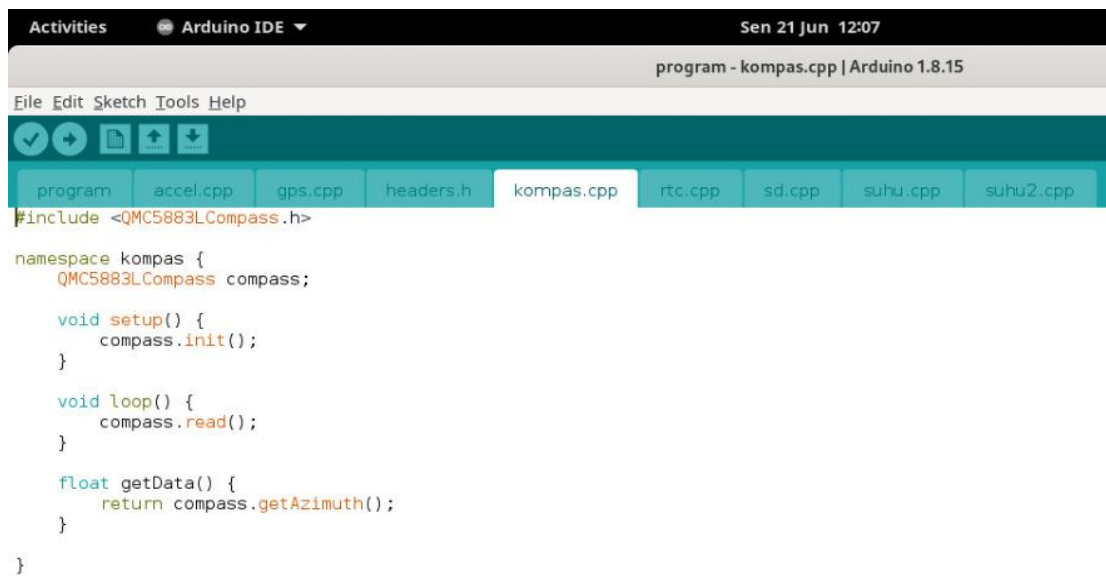
```
Activities Arduino IDE Sen 21 Jun 12:08
program - suhu2.cpp | Arduino 1.8.15
File Edit Sketch Tools Help
[Icons]
program accel.cpp gps.cpp headers.h kompas.cpp rtc.cpp sd.cpp suhu.cpp suhu2.cpp
#include <Wire.h>
#include <Adafruit_MLX90614.h>

Adafruit_MLX90614 mlx = Adafruit_MLX90614();

namespace suhu2 {
    void setup() {
        mlx.begin();
    }

    double getData() {
        return mlx.readObjectTempC();
    }
}
```

Pemrograman sensor suhu (*wireless*) MLX90614



```
Activities Arduino IDE Sen 21 Jun 12:07
program - kompas.cpp | Arduino 1.8.15
File Edit Sketch Tools Help
[Icons]
program accel.cpp gps.cpp headers.h kompas.cpp rtc.cpp sd.cpp suhu.cpp suhu2.cpp
#include <QMC5883LCompass.h>

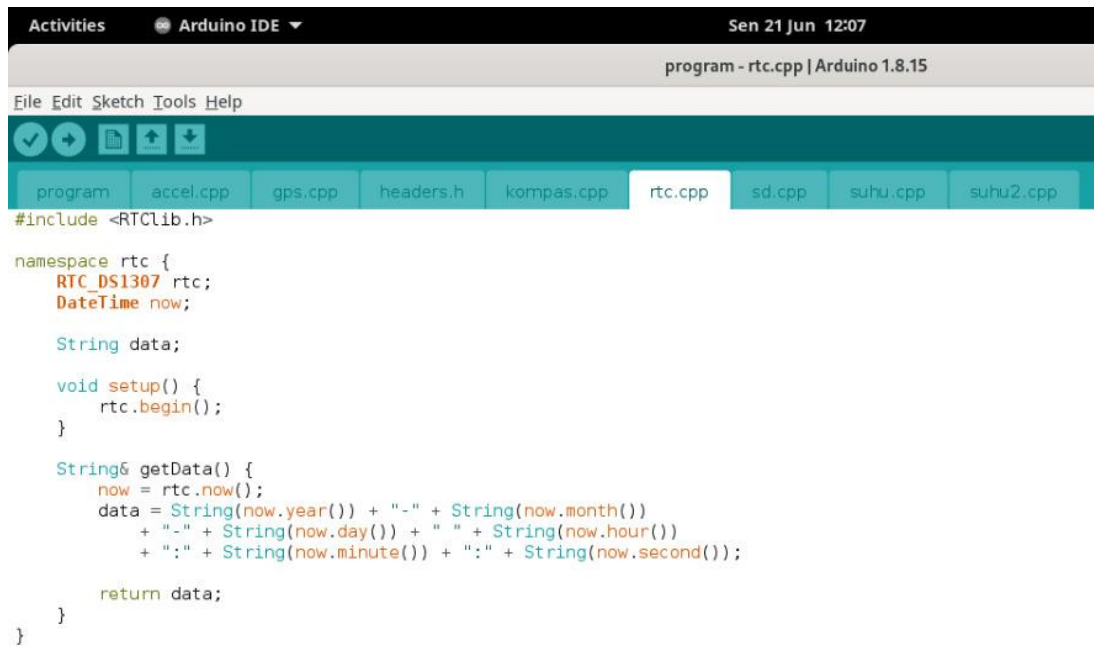
namespace kompas {
    QMC5883LCompass compass;

    void setup() {
        compass.init();
    }

    void loop() {
        compass.read();
    }

    float getData() {
        return compass.getAzimuth();
    }
}
```

Pemrograman sensor kompas HMC5883L



```
Activities Arduino IDE Sen 21 Jun 12:07
program - rtc.cpp | Arduino 1.8.15
File Edit Sketch Tools Help
program accel.cpp gps.cpp headers.h kompas.cpp rtc.cpp sd.cpp suhu.cpp suhu2.cpp
#include <RTCLib.h>

namespace rtc {
    RTC_DS1307 rtc;
    DateTime now;

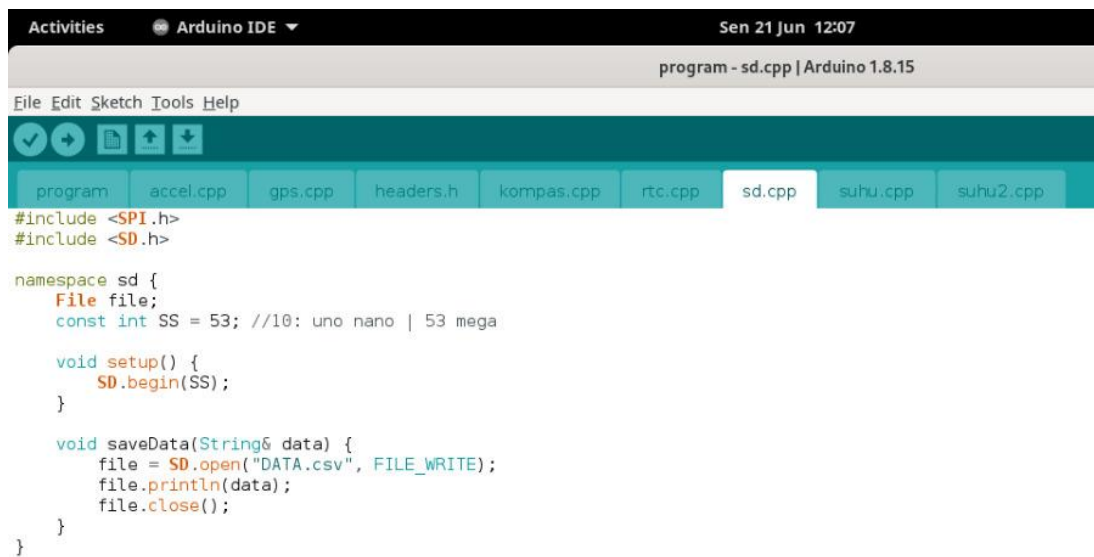
    String data;

    void setup() {
        rtc.begin();
    }

    String& getData() {
        now = rtc.now();
        data = String(now.year()) + "-" + String(now.month())
            + "-" + String(now.day()) + " " + String(now.hour())
            + ":" + String(now.minute()) + ":" + String(now.second());

        return data;
    }
}
```

Pemrograman RTC DS1307 (Sensor waktu)



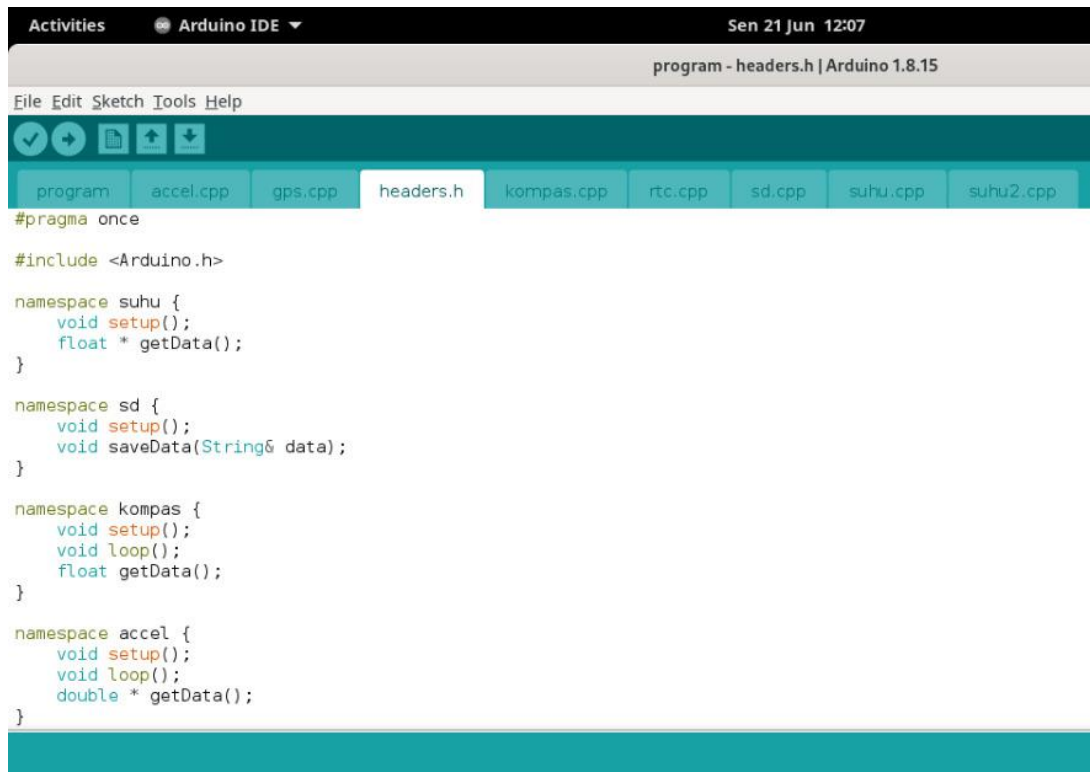
```
Activities Arduino IDE Sen 21 Jun 12:07
program - sd.cpp | Arduino 1.8.15
File Edit Sketch Tools Help
program accel.cpp gps.cpp headers.h kompas.cpp rtc.cpp sd.cpp suhu.cpp suhu2.cpp
#include <SPI.h>
#include <SD.h>

namespace sd {
    File file;
    const int SS = 53; //10: uno nano | 53 mega

    void setup() {
        SD.begin(SS);
    }

    void saveData(String& data) {
        file = SD.open("DATA.csv", FILE_WRITE);
        file.println(data);
        file.close();
    }
}
```

Pemrograman Modul *SD Card*

The screenshot shows the Arduino IDE interface with the 'headers.h' file open. The code defines namespaces for 'suhu', 'sd', 'kompas', and 'accel', each with its own 'setup()' and 'loop()' functions and a 'getData()' function. The 'suhu' namespace returns a float, 'sd' returns a String, 'kompas' returns a float, and 'accel' returns a double.

```
#pragma once

#include <Arduino.h>

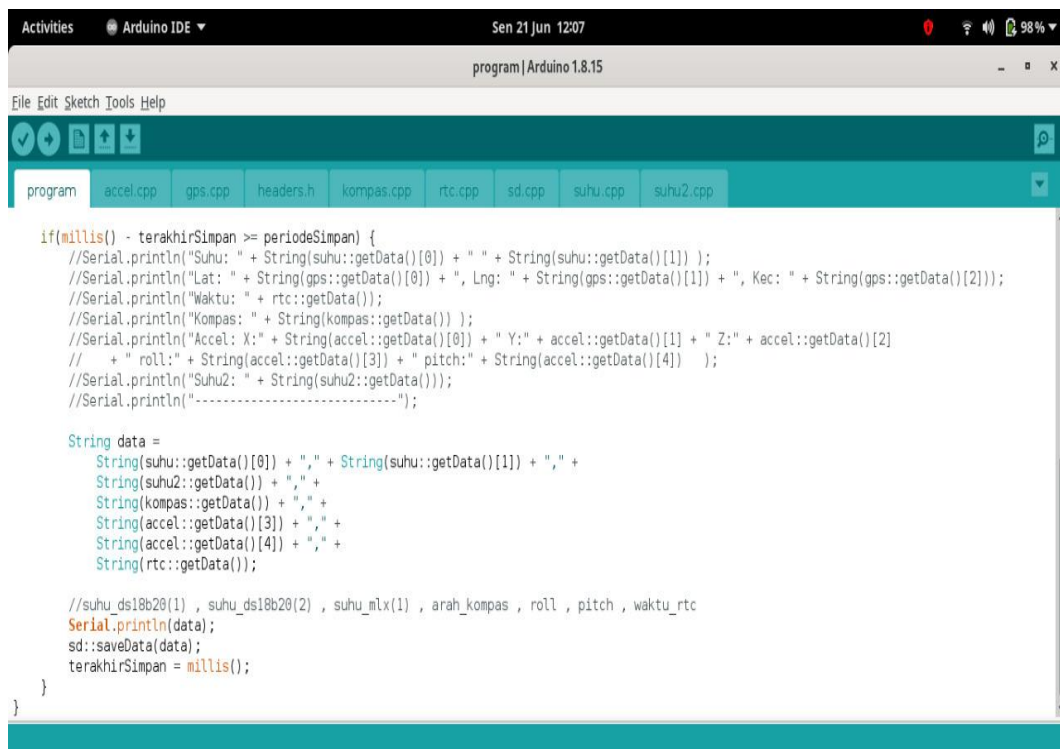
namespace suhu {
    void setup();
    float * getData();
}

namespace sd {
    void setup();
    void saveData(String& data);
}

namespace kompas {
    void setup();
    void loop();
    float getData();
}

namespace accel {
    void setup();
    void loop();
    double * getData();
}
```

Pemrograman *Headers* untuk penampilan dalam tabel hasil perekaman *data logger*

The screenshot shows the main program code in the Arduino IDE. It features a loop that checks if a certain time interval has passed. If so, it prints sensor data (Suhu, Waktu, Kompas, Accel) to the serial monitor and formats it into a single String 'data'. This string is then printed, saved to an SD card, and the timer is reset.

```
if(millis() - terakhirSimpan >= periodeSimpan) {
    //Serial.println("Suhu: " + String(suhu::getData()[0]) + " " + String(suhu::getData()[1]) );
    //Serial.println("Lat: " + String(gps::getData()[0]) + " Lng: " + String(gps::getData()[1]) + " , Kec: " + String(gps::getData()[2]));
    //Serial.println("Waktu: " + rtc::getData());
    //Serial.println("Kompas: " + String(kompas::getData()) );
    //Serial.println("Accel: X:" + String(accel::getData()[0]) + " Y:" + accel::getData()[1] + " Z:" + accel::getData()[2]
    //    + " roll:" + String(accel::getData()[3]) + " pitch:" + String(accel::getData()[4]) );
    //Serial.println("Suhu2: " + String(suhu2::getData()));
    //Serial.println("-----");

    String data =
        String(suhu::getData()[0]) + "," + String(suhu::getData()[1]) + "," +
        String(suhu2::getData()) + "," +
        String(kompas::getData()) + "," +
        String(accel::getData()[3]) + "," +
        String(accel::getData()[4]) + "," +
        String(rtc::getData());

    //suhu_ds18b20(1) , suhu_ds18b20(2) , suhu_mlx(1) , arah_kompas , roll , pitch , waktu_rtc
    Serial.println(data);
    sd::saveData(data);
    terakhirSimpan = millis();
}
}
```

Pemrograman akhir untuk *string data*

FOTO PROSES INSTALASI PADA OBJEK EKSPERIMEN

