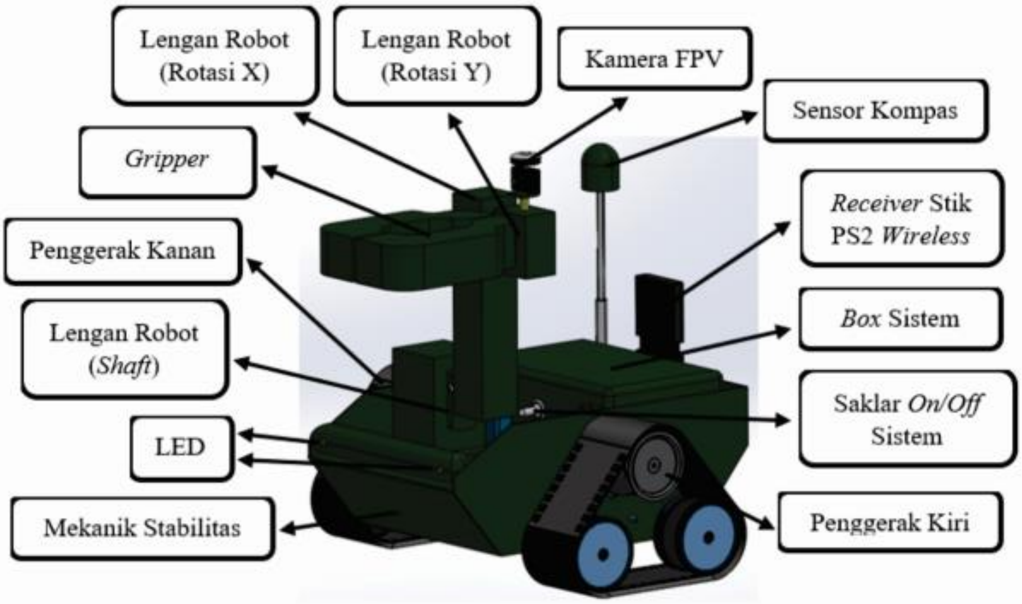
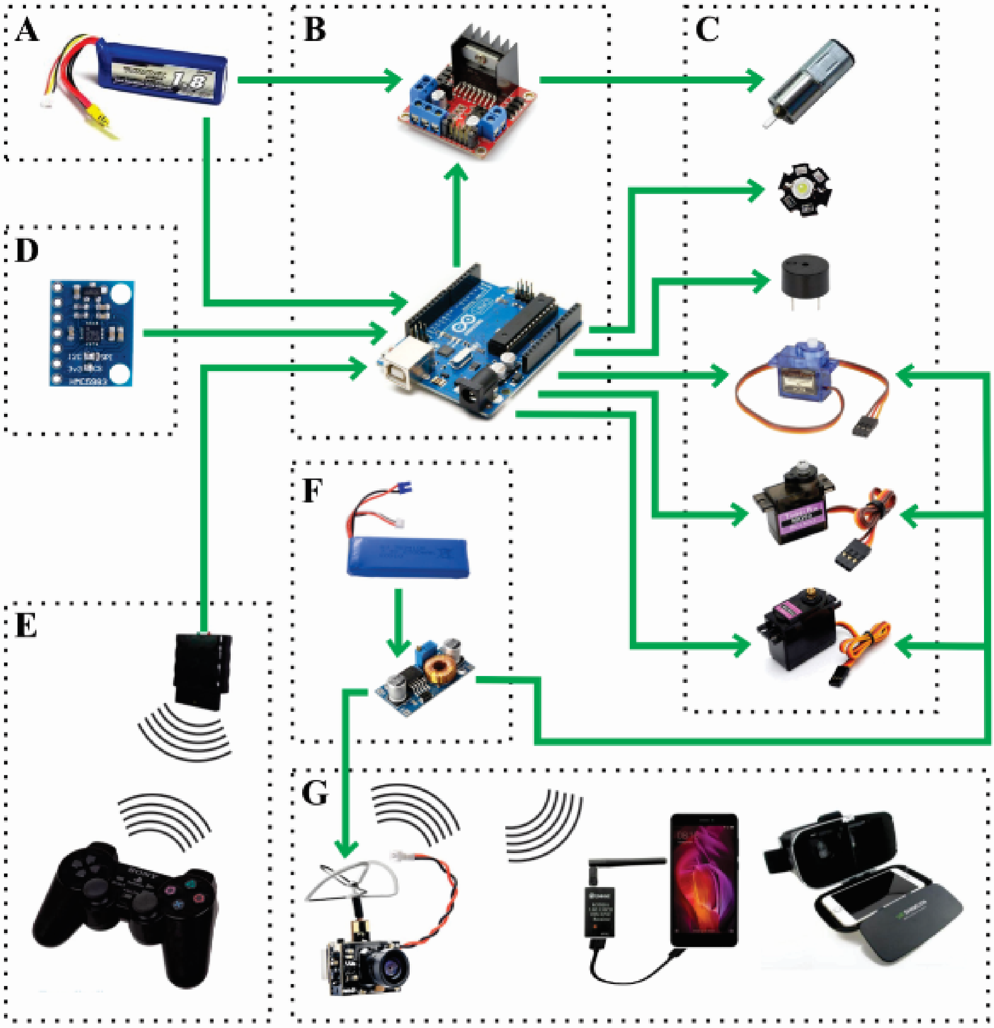
LAMPIRAN

Lampiran 1. Desain Robot



DESAIN ROBOT			KETERANGAN	
			A4	NO. 1
TEKNIK ELEKTRONIKA - D3 UNIVERSITAS NEGERI YOGYAKARTA	SKA. 1:1	DIP. FATCHUL A.		
	DIG. R. A. RAJIF	DIS. 24/6/2019	NIM. 16507134028	

Lampiran 2. Desain Sistem



A: Catu Daya 1 (Baterai Li-Po 3s 1800 mAh)
B: Sistem Kontrol (Arduino Uno R3, *Motor Driver* L298N)
C: *Output* (Motor DC, Servo Sg90, Servo Mg90, Servo Mg996, LED, *Buzzer*)
D: Sistem Navigasi (Sensor Kompas)
E: Kontroler (Stik PS2, *Receiver* Stik PS2)
F: Catu Daya 2 (Baterai Li-Po 2s 1000 mAh, DC-DC *Step Down*)
G: *Monitoring* (Kamera FPV, *Receiver* ROTG, *Smartphone*, VR *Box*)

DESAIN SISTEM			KETERANGAN	
			A4	NO. 2
TEKNIK ELEKTRONIKA - D3 UNIVERSITAS NEGERI YOGYAKARTA	SKA. 1:1	DIP. FATCHUL A.		
	DIG. R. A. RAJIF	DIS. 24/6/2019	NIM. 16507134028	

Lampiran 3. Foto Robot

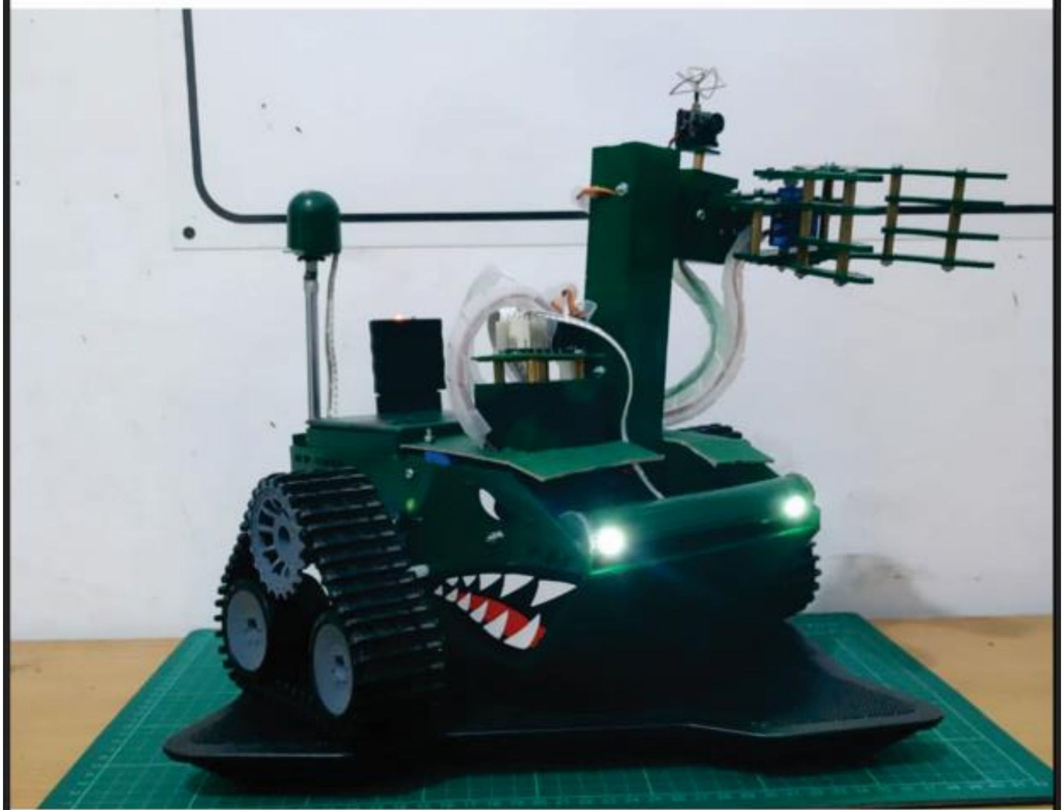
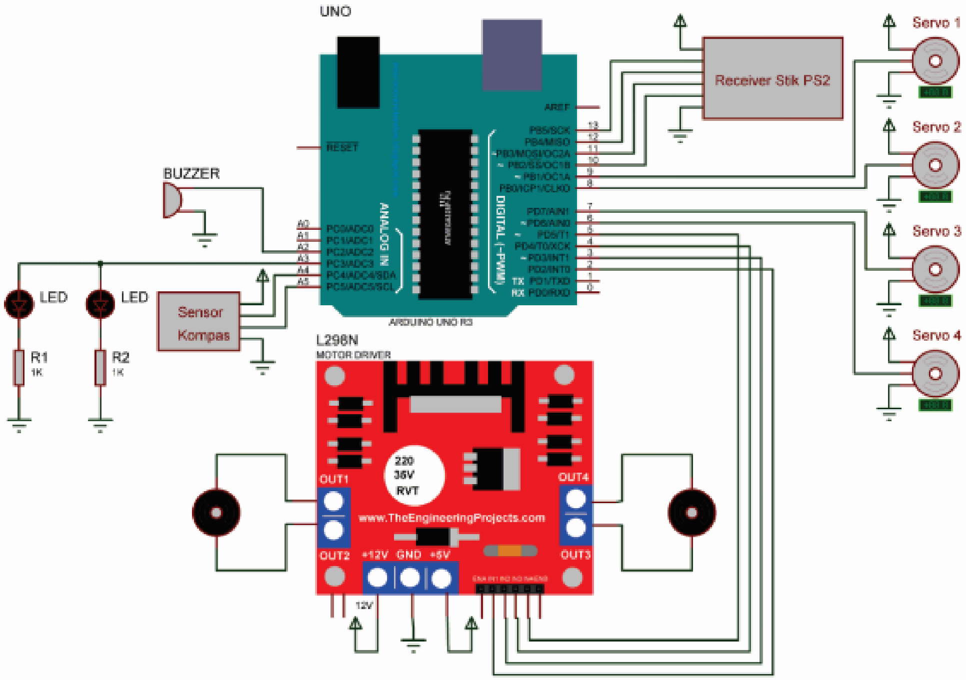


FOTO ROBOT			KETERANGAN	
			A4	NO. 3
TEKNIK ELEKTRONIKA - D3 UNIVERSITAS NEGERI YOGYAKARTA	SKA. 1:1	DIP. FATCHUL A.		
	DIG. R. A. RAJIF	DIS. 24/6/2019	NIM. 16507134028	

Lampiran 4. Rangkaian Elektronik



RANGKAIAN ELEKTRONIK			KETERANGAN	
TEKNIK ELEKTRONIKA - D3 UNIVERSITAS NEGERI YOGYAKARTA	SKA. 1:1	DIP. FATCHUL A.	A4	NO. 4
	DIG. R. A. RAJIF	DIS. 24/6/2019	NIM. 16507134028	

Lampiran 5. *Listing Program*

```
#include <PS2X_lib.h>
#include <Arduino.h>
#include <HMC5883L_Simple.h>
#include <Wire.h>
#include <Servo.h>

#define MR1    3
#define MR2    5
#define ML1    6
#define ML2    9
#define buzzer  A2
#define LED    A3

PS2X ps2x;
Servo shaftServo;
Servo rotateYServo;
Servo rotateXServo;
Servo gripServo;
HMC5883L_Simple Compass;

long periodMove = 50;
long periodNav = 50;
long periodLED = 50;
long periodArm = 50;
long time_Move = 0;
long time_Nav = 0;
long time_LED = 0;
long time_Arm = 0;
float sudutNav;

void indikatorSiap(){
 digitalWrite(buzzer, HIGH);
 delay(50);
 digitalWrite(buzzer, LOW);
 delay(50);
 digitalWrite(buzzer, HIGH);
 delay(50);
 digitalWrite(buzzer, LOW);
 delay(50);
 digitalWrite(buzzer, HIGH);
 delay(50);
 digitalWrite(buzzer, LOW);
}
void maju(){
 analogWrite(MR1, 255);
 analogWrite(ML1, 228);
}
void mundur(){
 analogWrite(MR2, 250);
 analogWrite(ML2, 255);
}
void belokKanan(){
 analogWrite(MR1, 255);
 analogWrite(ML2, 255);
}
void belokKiri(){
 analogWrite(MR2, 238);
 analogWrite(ML1, 255);
}
void robotStop(){
 analogWrite(MR1, 0);
 analogWrite(MR2, 0);
 analogWrite(ML1, 0);
 analogWrite(ML2, 0);
}
void robotNavigasi(){
 analogWrite(MR2, 192);
 analogWrite(ML1, 200);
}
void moveRobot(){
 if((ps2x.Analog(PSS_LY)<64)&&((ps2x.Analog(PSS_LX)>=64)
    &&(ps2x.Analog(PSS_LX)<=192))){
  maju();
 }
 else if((ps2x.Analog(PSS_LX)>192)&&((ps2x.Analog(PSS_LY)
      >=64)&&(ps2x.Analog(PSS_LY)<=192))){
  belokKanan();
 }
 else if((ps2x.Analog(PSS_LY)>192)&&((ps2x.Analog(PSS_LX)
      >=64)&&(ps2x.Analog(PSS_LX)<=192))){
  mundur();
 }
 else if((ps2x.Analog(PSS_LX)<64)&&((ps2x.Analog(PSS_LY)
      >=64)&&(ps2x.Analog(PSS_LY)<=192))){
  belokKiri();
 }
 else{
  robotStop();
 }
}
void sistemNavigasi(){
 if(ps2x.Button(PSB_PAD_UP)) {
  if(sudutNav > 335 && sudutNav < 345){
   robotStop();
  }
  else{
   robotNavigasi();
  }
 }
 else if(ps2x.Button(PSB_PAD_RIGHT)){
  if(sudutNav > 82 && sudutNav < 92){
   robotStop();
  }
  else{
   robotNavigasi();
  }
 }
```

LISTING PROGRAM			KETERANGAN	
TEKNIK ELEKTRONIKA - D3 UNIVERSITAS NEGERI YOGYAKARTA	SKA. 1:1	DIP. FATCHUL A.	A4	NO. 5
	DIG. R. A. RAJIF	DIS. 24/6/2019	NIM. 16507134028	

```
  else if(ps2x.Button(PSB_PAD_DOWN)){
    if(sudutNav > 205 && sudutNav < 215){
      robotStop();
    }
    else{
      robotNavigasi();
    }
  }
  else if(ps2x.Button(PSB_PAD_LEFT)){
    if(sudutNav > 270 && sudutNav < 280){
      robotStop();
    }
    else{
      robotNavigasi();
    }
  }
}
void lampu(){
  if(ps2x.Button(PSB_R1)){
    digitalWrite(LED, HIGH);
  }
  else if(ps2x.Button(PSB_L1)){
    digitalWrite(LED, LOW);
  }
}
void servoSpeed(Servo servo, int sudut,
                uint8_t Speed){
  if(Speed == 0){
    return;
  }
  int posisiSekarang = servo.read();
  for (int i = posisiSekarang; i != sudut;
      (posisiSekarang > sudut) ? i-- : i++){
    servo.write(i);
    if (Speed > 174){
      delayMicroseconds((256 - Speed) * 200);
    }
    else
    {
      delay((uint16_t)(256 - Speed) * 0.2f);
    }
  }
  servo.write(sudut);
}
void arm(){
  if((ps2x.Analog(PSS_RY) <= 0)){
    servoSpeed(shaftServo, 163, 235);
  }
  else if((ps2x.Analog(PSS_RY) <= 30)){
    servoSpeed(shaftServo, 150, 235);
  }
  else if((ps2x.Analog(PSS_RY) <= 60)){
    servoSpeed(shaftServo, 130, 235);
  }
  else if((ps2x.Analog(PSS_RY) <= 90)){
    servoSpeed(shaftServo, 110, 235);
  }
  else if((ps2x.Analog(PSS_RY) <= 127)){
    servoSpeed(shaftServo, 95, 235);
  }

  if(ps2x.NewButtonState(PSB_BLUE)){
    rotateXServo.write(90);
  }
  else if(ps2x.ButtonPressed(PSB_RED)){
    rotateXServo.write(180);
  }
  else if(ps2x.ButtonReleased(PSB_PINK)){
    rotateXServo.write(0);
  }

  if(ps2x.Button(PSB_GREEN)){
    rotateYServo.write(10);
  }
  else{
    rotateYServo.write(88);
  }

  if(ps2x.Button(PSB_R2)){
    gripServo.write(45);
  }
  else if(ps2x.Button(PSB_L2)){
    gripServo.write(90);
  }
}

void setup() {
  Serial.begin(115200);
  Wire.begin();
  ps2x.config_gamepad(10,12,11,13, false, false);
  pinMode(MR1, OUTPUT);
  pinMode(MR2, OUTPUT);
  pinMode(ML1, OUTPUT);
  pinMode(ML2, OUTPUT);
  pinMode(buzzer, OUTPUT);
  pinMode(LED, OUTPUT);

  Compass.SetDeclination(0, 49, 'E');
  Compass.SetSamplingMode(COMPASS_SINGLE);
  Compass.SetScale(COMPASS_SCALE_130);
  Compass.SetOrientation(COMPASS_HORIZONTAL_X_NORTH);

  shaftServo.attach(2);
  shaftServo.write(95);
  rotateYServo.attach(4);
  rotateYServo.write(88);
  rotateXServo.attach(7);
  rotateXServo.write(90);
  gripServo.attach(8);
  gripServo.write(90);
  delay(3000);
  robotStop();
  indikatorSiap();
}
```

LISTING PROGRAM			KETERANGAN	
			A4	NO. 5
TEKNIK ELEKTRONIKA - D3 UNIVERSITAS NEGERI YOGYAKARTA	SKA. 1:1	DIP. FATCHUL A.		
	DIG. R. A. RAJIF	DIS. 24/6/2019	NIM. 16507134028	

```
void loop() {
 ps2x.read_gamepad();
 sudutNav = Compass.GetHeadingDegrees();
 if(millis() - time_Move > periodMove){
  moveRobot();
  time_Move = millis();
 }
 if(millis() - time_Nav > periodNav){
  time_Nav = millis();
  sistemNavigasi();
 }
 if(millis() - time_LED > periodLED){
  time_LED = millis();
  lampu();
 }
 if(millis() - time_Arm > periodArm){
  time_Arm = millis();
  arm();
 }
delay(15);
}
```

LISTING PROGRAM			KETERANGAN	
			A4	NO. 5
TEKNIK ELEKTRONIKA - D3 UNIVERSITAS NEGERI YOGYAKARTA	SKA. 1:1	DIP. FATCHUL A.		
	DIG. R. A. RAJIF	DIS. 24/6/2019	NIM. 16507134028	

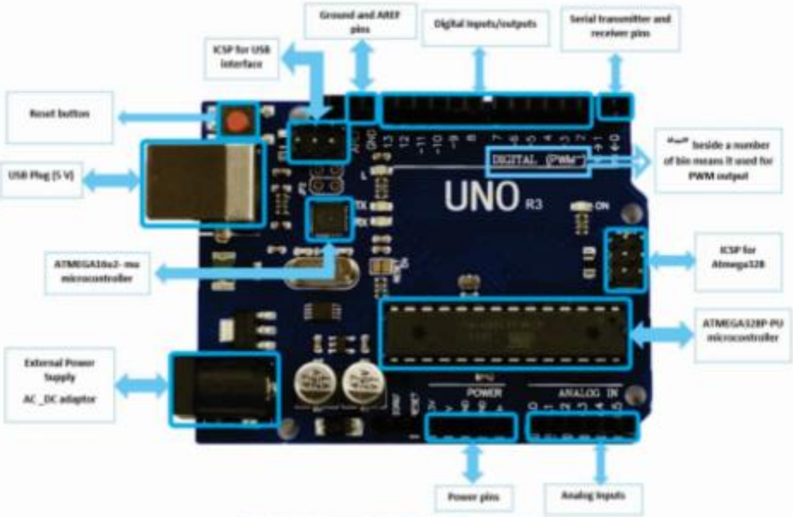
Lampiran 6. Daftar Komponen

1. Baterai Li-Po 3 Cell 1800 mAh
2. Baterai Li-Po 2 Cell 1000 mAh
3. Modul Regulator XL4005
4. Stik PS2 *Wireless*
5. Sensor Kompas
6. Kamera FPV
7. Arduino Uno R3
8. *Motor Driver* L298N
9. Motor DC
10. Motor Servo
11. LED
12. *Buzzer*
13. *Receiver* Kamera FPV ROTG
14. VR *Box*
15. *Box Panel*

DAFTAR KOMPONEN			KETERANGAN	
			A4	NO. 6
TEKNIK ELEKTRONIKA - D3 UNIVERSITAS NEGERI YOGYAKARTA	SKA. 1:1	DIP. FATCHUL A.		
	DIG. R. A. RAJIF	DIS. 24/6/2019	NIM. 16507134028	

Lampiran 7. *Datasheet* Arduino Uno R3

Arduino Uno R3



INTRODUCTION

Arduino is used for building different types of electronic circuits easily using of both a physical programmable circuit board usually microcontroller and piece of code running on computer with USB connection between the computer and Arduino.

Programming language used in Arduino is just a simplified version of C++ that can easily replace thousands of wires with words.

DATASHEET ARDUINO UNO R3			KETERANGAN	
TEKNIK ELEKTRONIKA - D3 UNIVERSITAS NEGERI YOGYAKARTA	SKA. 1:1	DIP. FATCHUL A.	A4	NO. 7
	DIG. R. A. RAJIF	DIS. 24/6/2019	NIM. 16507134028	

ARDUINO UNO-R3 PHYSICAL COMPONENTS

ATMEGA328P-PU microcontroller

The most important element in Arduino Uno R3 is ATMEGA328P-PU is an 8-bit Microcontroller with flash memory reach to 32k bytes. It's features as follow:

- **High Performance, Low Power AVR**
- **Advanced RISC Architecture**
  - 131 Powerful Instructions – Most Single Clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Up to 20 MIPS Throughput at 20 MHz
  - On-chip 2-cycle Multiplier
- **High Endurance Non-volatile Memory Segments**
  - 4/8/16/32K Bytes of In-System Self-Programmable Flash program memory
  - 256/512/512/1K Bytes EEPROM
  - 512/1K/1K/2K Bytes Internal SRAM
  - Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
  - Data retention: 20 years at 85°C/100 years at 25°C
  - Optional Boot Code Section with Independent Lock Bits
  - In-System Programming by On-chip Boot Program
  - True Read-While-Write Operation
  - Programming Lock for Software Security
- **Peripheral Features**
  - Two 8-bit Timer/Counters with Separate Prescaler and Compare Mode
  - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Six PWM Channels
  - 8-channel 10-bit ADC in TQFP and QFN/MLF package
  - Temperature Measurement
  - 6-channel 10-bit ADC in PDIP Package
  - Temperature Measurement
  - Programmable Serial USART

DATASHEET ARDUINO UNO R3			KETERANGAN	
			A4	NO. 7
TEKNIK ELEKTRONIKA - D3 UNIVERSITAS NEGERI YOGYAKARTA	SKA. 1:1	DIP. FATCHUL A.		
	DIG. R. A. RAJIF	DIS. 24/6/2019	NIM. 16507134028	

- Master/Slave SPI Serial Interface
- Byte-oriented 2-wire Serial Interface (Philips I2 C compatible)
- Programmable Watchdog Timer with Separate On-chip Oscillator
- On-chip Analog Comparator
- Interrupt and Wake-up on Pin Change

• **Special Microcontroller Features**

- Power-on Reset and Programmable Brown-out Detection
- Internal Calibrated Oscillator
- External and Internal Interrupt Sources
- Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby, and Extended Standby

• **I/O and Packages**

- 23 Programmable I/O Lines
- 28-pin PDIP, 32-lead TQFP, 28-pad QFN/MLF and 32-pad QFN/MLF

• **Operating Voltage:**

- 1.8 - 5.5V

• **Temperature Range:**

- -40°C to 85°C

• **Speed Grade:**

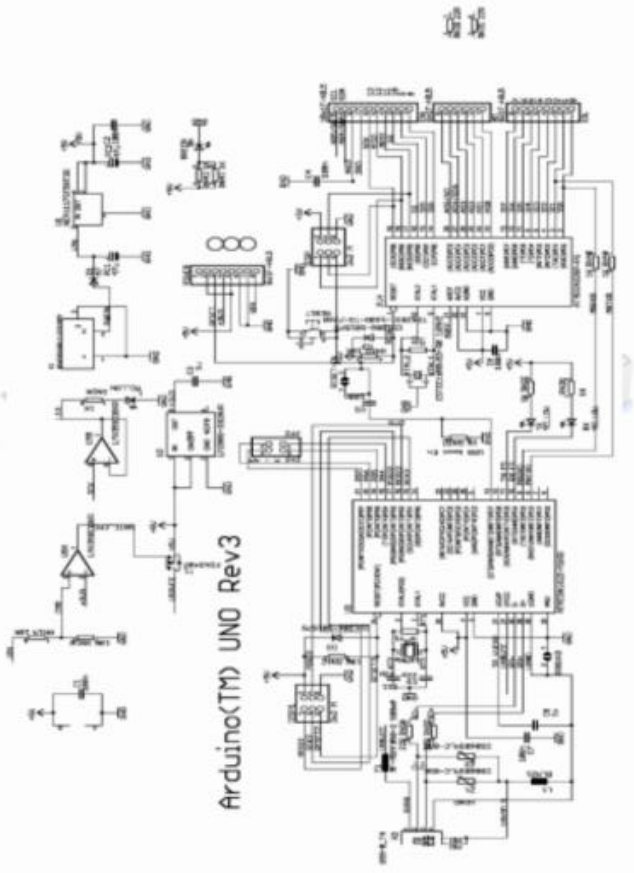
- 0 - 4 MHz@1.8 - 5.5V, 0 - 10 MHz@2.7 - 5.5V, 0 - 20 MHz @ 4.5 - 5.5V

• **Power Consumption at 1 MHz, 1.8V, 25°C**

- Active Mode: 0.2 mA
- Power-down Mode: 0.1 µA
- Power-save Mode: 0.75 µA (Including 32 kHz RTC)

			KETERANGAN	
DATASHEET ARDUINO UNO R3			A4	NO. 7
TEKNIK ELEKTRONIKA - D3 UNIVERSITAS NEGERI YOGYAKARTA	SKA. 1:1	DIP. FATCHUL A.		
	DIG. R. A. RAJIF	DIS. 24/6/2019	NIM. 16507134028	

ARDUINO UNO R3 SCHEMATIC DIAGRAM



DATASHEET ARDUINO UNO R3			KETERANGAN	
			A4	NO. 7
TEKNIK ELEKTRONIKA - D3 UNIVERSITAS NEGERI YOGYAKARTA	SKA. 1:1	DIP. FATCHUL A.		
	DIG. R. A. RAJIF	DIS. 24/6/2019	NIM. 16507134028	

Lampiran 8. *Datasheet Motor Driver* L298N

HT **Handson Technology**

User Guide

L298N Dual H-Bridge Motor Driver

This dual bidirectional motor driver, is based on the very popular L298 Dual H-Bridge Motor Driver Integrated Circuit. The circuit will allow you to easily and independently control two motors of up to 2A each in both directions.It is ideal for robotic applications and well suited for connection to a microcontroller requiring just a couple of control lines per motor. It can also be interfaced with simple manual switches, TTL logic gates, relays, etc. This board equipped with power LED indicators, on-board +5V regulator and protection diodes.

SKU: MDU-1049

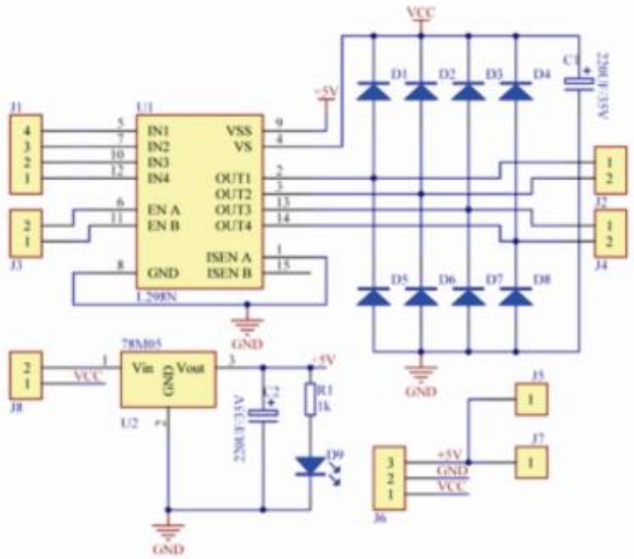
Brief Data:

- Input Voltage: 3.2V~40Vdc.
- Driver: L298N Dual H Bridge DC Motor Driver
- Power Supply: DC 5 V - 35 V
- Peak current: 2 Amp
- Operating current range: 0 ~ 36mA
- Control signal input voltage range :
- Low: -0.3V ≤ Vin ≤ 1.5V.
- High: 2.3V ≤ Vin ≤ Vss.
- Enable signal input voltage range :
    - Low: -0.3 ≤ Vin ≤ 1.5V (control signal is invalid).
    - High: 2.3V ≤ Vin ≤ Vss (control signal active).
- Maximum power consumption: 20W (when the temperature T = 75 ℃).
- Storage temperature: -25 ℃ ~ +130 ℃.
- On-board +5V regulated Output supply (supply to controller board i.e. Arduino).
- Size: 3.4cm x 4.3cm x 2.7cm

1 www.handsontec.com

DATASHEET MOTOR DRIVER L298N			KETERANGAN	
			A4	NO. 8
TEKNIK ELEKTRONIKA - D3 UNIVERSITAS NEGERI YOGYAKARTA	SKA. 1:1	DIP. FATCHUL A.		
	DIG. R. A. RAJIF	DIS. 24/6/2019	NIM. 16507134028	

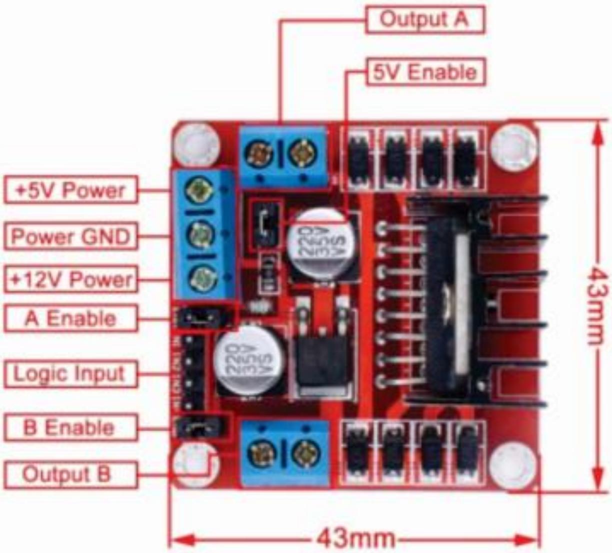
Schematic Diagram:



2 www.handsontec.com

DATASHEET MOTOR DRIVER L298N			KETERANGAN	
TEKNIK ELEKTRONIKA - D3 UNIVERSITAS NEGERI YOGYAKARTA	SKA. 1:1	DIP. FATCHUL A.	A4	NO. 8
	DIG. R. A. RAJIF	DIS. 24/6/2019	NIM. 16507134028	

Board Dimension & Pins Function:



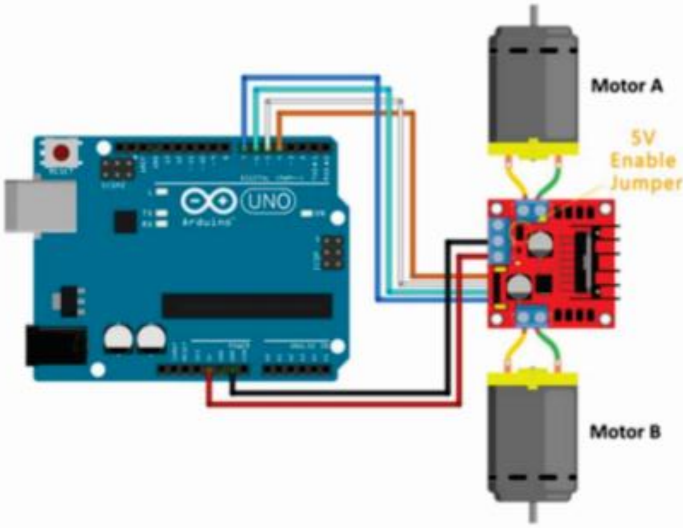
3 www.handsontec.com

DATASHEET MOTOR DRIVER L298N			KETERANGAN	
			A4	NO. 8
TEKNIK ELEKTRONIKA - D3 UNIVERSITAS NEGERI YOGYAKARTA	SKA. 1:1	DIP. FATCHUL A.		
	DIG. R. A. RAJIF	DIS. 24/6/2019	NIM. 16507134028	

Connection Examples:

Controlling 2-DC Motor with +5V Arduino onboard Power Supply:

Below is the circuit connection use the on-board +5V power supply from Arduino board, and should be done without the 5V Enable Jumper on (Active 5V). This connection can drive two 5V DC motors simultaneously.



Sketch Listing:

Copy and paste the sketch below to Arduino IDE and upload to Arduino Uno/Mega board.

```
/*=======================================================================
//  Author      : Handson Technology
//  Project     : Arduino Uno
//  Description : L298N Motor Driver
//  Source-Code : L298N_Motor.ino
//  Program: Control 2 DC motors using L298N H Bridge Driver
//=======================================================================
*/

// Definitions Arduino pins connected to input H Bridge
int IN1 = 4;
int IN2 = 5;
int IN3 = 6;
int IN4 = 7;

void setup()
{
 // Set the output pins
```


DATASHEET MOTOR DRIVER L298N			KETERANGAN	
TEKNIK ELEKTRONIKA - D3 UNIVERSITAS NEGERI YOGYAKARTA	SKA. 1:1	DIP. FATCHUL A.	A4	NO. 8
	DIG. R. A. RAJIF	DIS. 24/6/2019	NIM. 16507134028	

```
 pinMode(IN1, OUTPUT);
 pinMode(IN2, OUTPUT);
 pinMode(IN3, OUTPUT);
 pinMode(IN4, OUTPUT);
}

void loop()
{
 // Rotate the Motor A clockwise
 digitalWrite(IN1, HIGH);
 digitalWrite(IN2, LOW);
 delay(2000);
 // Motor A
 digitalWrite(IN1, HIGH);
 digitalWrite(IN2, HIGH);
 delay(500);

 // Rotate the Motor B clockwise
 digitalWrite(IN3, HIGH);
 digitalWrite(IN4, LOW);
 delay(2000);
 // Motor B
 digitalWrite(IN3, HIGH);
 digitalWrite(IN4, HIGH);
 delay(500);

 // Rotates the Motor A counter-clockwise
 digitalWrite(IN1, LOW);
 digitalWrite(IN2, HIGH);
 delay(2000);
 // Motor A
 digitalWrite(IN1, HIGH);
 digitalWrite(IN2, HIGH);
 delay(500);

 // Rotates the Motor B counter-clockwise
 digitalWrite(IN3, LOW);
 digitalWrite(IN4, HIGH);
 delay(2000);
 // Motor B
 digitalWrite(IN3, HIGH);
 digitalWrite(IN4, HIGH);
 delay(500);
}
```


DATASHEET MOTOR DRIVER L298N			KETERANGAN	
			A4	NO. 8
TEKNIK ELEKTRONIKA - D3 UNIVERSITAS NEGERI YOGYAKARTA	SKA. 1:1	DIP. FATCHUL A.		
	DIG. R. A. RAJIF	DIS. 24/6/2019	NIM. 16507134028	

Lampiran 9. *Datasheet* 3-*Axis Digital Compass*

3-axis Digital Compass Module

Using a magnetometer can be a little tricky, especially if you're unsure about the formulas to use to get the correct bearing and when other magnetic objects are interfering with your signal.
We've created a library for our HMC5883L module, which will also be compatible with other HMC5883L breakout boards made by other manufacturers.

Join us whilst we cover the following:

- *Understand what is a magnetometer and how they work.*
- *Introduce the HMC5883L Arduino Library*
- *Explain how to extract data from the HMC5883L.*
- *Explain how to calculate a bearing from this data.*

So, assuming you have an HMC5883L Breakout Board and an Arduino. We will walk through using the HMC5883L breakout board to output our bearing.

Note: The HMC5883L is not the same as the HMC5883. This library only covers the HMC5883L sold on our website.

How do compasses work?

Firstly an introduction, a (standard handheld) compass works by aligning itself to the earth magnetic field. Because the compass' needle is a ferrous material, it aligns swings on its bearing in the center as the magnetic field of the earth pulls it into alignment. These magnetic fields expand throughout the surface of the earth (and beyond) so we can use them to help us tell us which direction we facing.

Our magnetometer uses these magnetic fields, however it doesn't pull on a little needle inside it! (It probably wouldn't fit anyway). Inside our magnetometer are three magneto-resistive sensors on three axis. These can be quite complicated to understand (not to mention explain!), it is sufficient to say that the effect of magnetic fields on these sensors adjust the current flow through the sensor. By applying a scale to this current, we can tell the magnetic force (measured in Gauss) on this sensor.

DATASHEET 3-*AXIS DIGITAL COMPASS*			KETERANGAN	
TEKNIK ELEKTRONIKA - D3 UNIVERSITAS NEGERI YOGYAKARTA	SKA. 1:1	DIP. FATCHUL A.	A4	NO. 9
	DIG. R. A. RAJIF	DIS. 24/6/2019	NIM. 16507134028	

For a detailed explanation to magneto-resistive sensors use this application note: Magneto-Resistive Sensors.
By combining information about two or more of these axis we can start to use the difference in the magnetic fields in the sensors to infer our bearing to magnetic north.

How do we use one?

Okay, so now we know how to use one, the first step is to get some data out of our compass. The HMC5883L is a device which communicates over I2C, a really easy communication protocol to use, and our favorite way to interface with our breakout boards here at Love Electronics. All you need to do is plug the breakout board into your breadboard and connect up the following pins to your Arduino:

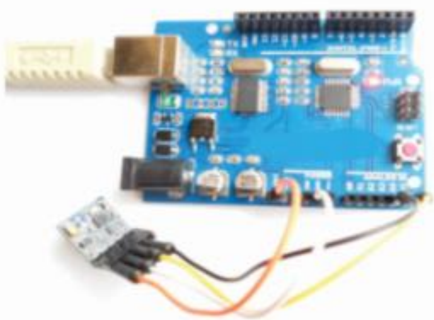
For Arduino UNO/Duemilanove:

- Arduino GND -> HMC5883L GND
- Arduino 3.3V -> HMC5883L VCC
- Arduino A4 (SDA) -> HMC5883L SDA
- Arduino A5 (SCL) -> HMC5883L SCL

For Arduino Mega:

- Arduino GND -> HMC5883L GND
- Arduino 3.3V -> HMC5883L VCC
- Arduino 20 (SDA) -> HMC5883L SDA
- Arduino 21 (SCL) -> HMC5883L SCL

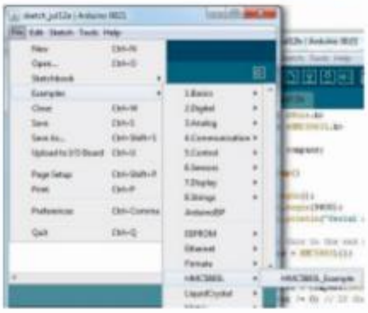
Note: You will also need to add two 'pull-up' resistors to enable I2C. For this, connect two 4.7k or 10k resistors between SDA and VCC, and SCL and VCC. We also test it without pull-up resistors. It works well. For more information about IIC pull-up resistor, please visit this page: http://arduino.cc/playground/Main/I2CBi-directionalLevelShifter



DATASHEET 3-AXIS DIGITAL COMPASS			KETERANGAN	
			A4	NO. 9
TEKNIK ELEKTRONIKA - D3 UNIVERSITAS NEGERI YOGYAKARTA	SKA. 1:1	DIP. FATCHUL A.		
	DIG. R. A. RAJIF	DIS. 24/6/2019	NIM. 16507134028	

*Now of course to talk to the HMC5883L we will need some code, helpfully we've written an Arduino library for the HMC5883L which makes this really easy. Simply download the Arduino Library for HMC5883L and extract it to your **library** folder in your Arduino installation. Mine is here: C:\Program Files (x86)\arduino-0021\libraries*

Once you have the library installed start your Arduino IDE and we can get coding. If you just want to go and get all the code up without coding along, simply open the HMC5883L_Example file from the Arduino Menu, otherwise we can code way together and you can fully understand everything we are going to write.



Using the HMC5883L Arduino Library

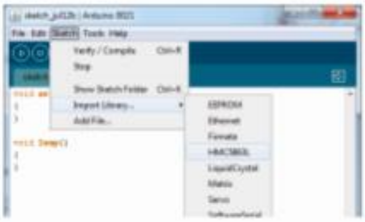
*What we want to accomplish is an Arduino sketch that will tell us the which direction we are pointing in degrees, so it should read 0° for when we are point at **magnetic north**, and 180° when we are pointed south.*

For a quick reality check, the bearing you get from your compass will be a little off, the compass senses magnetic fields, so any ferreous material anywhere near your compass can affect your output considerably. Also any kind of radio waves (I'm looking at you mobile phone!) and don't even let it see any stereo speakers!!

*So, open your Arduino IDE and begin a new sketch, and quickly start off by writing the initial **setup** and **loop** method placeholders.*

DATASHEET 3-AXIS DIGITAL COMPASS			KETERANGAN	
			A4	NO. 9
TEKNIK ELEKTRONIKA - D3 UNIVERSITAS NEGERI YOGYAKARTA	SKA. 1:1	DIP. FATCHUL A.		
	DIG. R. A. RAJIF	DIS. 24/6/2019	NIM. 16507134028	

Then we must import the HMC5883L Arduino Library for us to use. Because you installed the library into your libraries folder you should be able to add it by selecting the following menu option.



The HMC5883L communicates over I2C, so we must also import the Wire library (do this the same way) and add the following code to the setup() method to start the Wire library when we begin our Sketch:

```
Wire.begin();
```

We will now able to use devices on the I2C bus. We also want to communicate with our computer to report our findings, so initialize the serial port by adding the following two lines into **setup()**:

```
Serial.begin(9600);
Serial.println("Serial started.");
```

So, if you run this sketch we should get "Serial started." reported via the serial window. Great!
Now we need to declare an instance of the HMC5883L we can use throughout our sketch. So let's add an HMC5883L as a global variable by adding outside the **setup()** and **loop()** methods, and initialize it inside the **setup()** method.

```
// Add this outside the methods as a global variable.
HMC5883L compass;

// Add this to the end of setup() to create an instance of HMC5883L.
compass = HMC5883L();
```

Once we have an instance of the compass we need to set it up, the compass needs to know what kind of gain (response scale) to work out, and how to output its measurements. We can configure the compass by adding the following lines after creating the compass in **setup()**:

```
Serial.println("Setting scale to +/- 1.3 Ga");
int error = compass.SetScale(1.3); // Set the scale of the compass.
if(error != 0) // If there is an error, print it out.
  Serial.println(compass.GetErrorText(error));

Serial.println("Setting measurement mode to continuous.");
error = compass.SetMeasurementMode(Measurement_Continuous); // Set the measurement mode to Continuous
```

DATASHEET 3-AXIS DIGITAL COMPASS			KETERANGAN	
			A4	NO. 9
TEKNIK ELEKTRONIKA - D3 UNIVERSITAS NEGERI YOGYAKARTA	SKA. 1:1	DIP. FATCHUL A.		
	DIG. R. A. RAJIF	DIS. 24/6/2019	NIM. 16507134028	

```
if(error != 0) // If there is an error, print it out.
  Serial.println(compass.GetErrorText(error));
```

This will set the gain of the device to 1.3 Gauss (Ga), this means the device is able to report magnetic fields up to -/+ 1.3 Gauss. The lower you can make this number the more precice the compass will be, however if you have other objects interfering with the compass you may need to raise it to avoid overloading the sensor. Bear in mind that the compass can only accept certain gains, so check the datasheet to see what is available. If you choose an invalid gain the application will let you know by means of the error variable the Set method returns.

Your HMC5883L should now be configured and taking measurements, all the remains is to ask the sensor the data! To make sure we are on the same page, here is my current sketch:

So the HMC5883L Arduino Library can provide two different values for you to use. You can call either of the following:

```
// Retrive the raw values from the compass (not scaled).
MagnetometerRaw raw = compass.ReadRawAxis();
// Retrive the scaled values from the compass (scaled to the configured scale).
MagnetometerScaled scaled = compass.ReadScaledAxis();
```

DATASHEET 3-AXIS DIGITAL COMPASS			KETERANGAN	
TEKNIK ELEKTRONIKA - D3 UNIVERSITAS NEGERI YOGYAKARTA	SKA. 1:1	DIP. FATCHUL A.	A4	NO. 9
	DIG. R. A. RAJIF	DIS. 24/6/2019	NIM. 16507134028	

ReadRawAxis returns the values retrived straight from the magnetometer, if you are not interested in the actual magnetic strength of the field you can use this to return a **MagnetometerRaw** structure, which has three fields for you to access the values on each axis:

```
MagnetometerRaw raw = compass.ReadRawAxis();
int xAxis = raw.XAxis;
int yAxis = raw.YAxis;
int zAxis = raw.ZAxis;
```

The **ReadScaledAxis** provides the same data, in the same structure (named **MagnetometerScaled**), however these values are scaled to the gain we set the device to operate (-/+ 1.3 Ga) when we called **compass.SetScale(float gauss);**. You can use this data when you want to know the actual magnetic value each sensor is seeing.

Calculate your bearing

Now we know how to use the arduino library to communicate over I2C to the HMC5883L triple axis magnetometer chip we can put a little bit of math together to calculate the bearing.

What we are going to do is add the following code to the **loop()** method that is going to retrive the values from the device and do the actual bearing calculation:

```
void Loop()
{
  // Retrive the raw values from the compass (not scaled).
  MagnetometerRaw raw = compass.ReadRawAxis();
  // Retrived the scaled values from the compass (scaled to the configured scale).
  MagnetometerScaled scaled = compass.ReadScaledAxis();

  // Calculate heading when the magnetometer is level, then correct for signs of axis.
  float heading = atan2(raw.YAxis, raw.XAxis);

  // Correct for when signs are reversed.
  if(heading < 0)
    heading += 2*PI;

  // Convert radians to degrees for readability.
  float headingDegrees = heading * 180/M_PI;

  // Output the data via the serial port.
  Output(raw, scaled, heading, headingDegrees);
}
```

We also need to add the **Output** method we refer to at the end of this method:

```
// Output the data down the serial port.
void Output(MagnetometerRaw raw, MagnetometerScaled scaled, float heading, float headingDegrees)
{
```

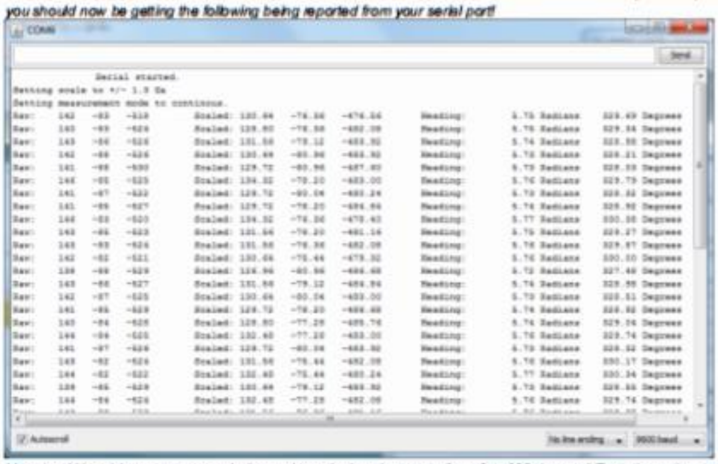
<table>
<tr><td colspan="3" rowspan="2">DATASHEET 3-AXIS DIGITAL COMPASS</td><td colspan="2">KETERANGAN</td></tr>
<tr><td rowspan="2">A4</td><td rowspan="2">NO. 9</td></tr>
<tr><td rowspan="2">TEKNIK ELEKTRONIKA - D3
UNIVERSITAS NEGERI YOGYAKARTA</td><td>SKA. 1:1</td><td>DIP. FATCHUL A.</td></tr>
<tr><td>DIG. R. A. RAJIF</td><td>DIS. 24/6/2019</td><td colspan="2">NIM. 16507134028</td></tr>
</table>

```
   Serial.print("Raw:\t");
   Serial.print(raw.XAxis);
   Serial.print("   ");
   Serial.print(raw.YAxis);
   Serial.print("   ");
   Serial.print(raw.ZAxis);
   Serial.print("   \tScaled:\t");

   Serial.print(scaled.XAxis);
   Serial.print("   ");
   Serial.print(scaled.YAxis);
   Serial.print("   ");
   Serial.print(scaled.ZAxis);

   Serial.print("   \tHeading:\t");
   Serial.print(heading);
   Serial.print(" Radians   \t");
   Serial.print(headingDegrees);
   Serial.println(" Degrees   \t");
}
```

You can now go ahead and run the sketch. Assuming you have got all the code in and connected your compass properly you should now be getting the following being reported from your serial port!



You should be able to rotate your device and see the bearing rotate from 0 to 360 degrees! Try to keep the compass away from anything magnetic to avoid interference, but try holding something like your mobile phone next to the compass to see how bad the effect can be on the results.

DATASHEET 3-*AXIS DIGITAL COMPASS*			KETERANGAN	
			A4	NO. 9
TEKNIK ELEKTRONIKA - D3 UNIVERSITAS NEGERI YOGYAKARTA	SKA. 1:1	DIP. FATCHUL A.		
	DIG. R. A. RAJIF	DIS. 24/6/2019	NIM. 16507134028	