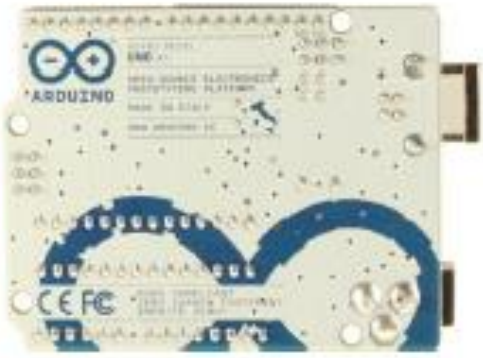
LAMPIRAN-LAMPIRAN

DATA SHEET ARDUINO UNO R3

Arduino Uno

Arduino Uno R3 Front



Arduino Uno R3 Back

Arduino Uno R2 Front

Arduino Uno SMD

Arduino Uno Front

Arduino Uno Back

Overview

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.
The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.
Revision 2 of the Uno board has a resistor pulling the 8U2 HWB line to ground, making it easier to put into DFU mode.
Revision 3 of the board has the following new features:

- 1.0 pinout: added SDA and SCL pins that are near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board. In future, shields will be compatible both with the board that use the AVR, which operate with 5V and with the Arduino Due that operate with 3.3V. The second one is a not connected pin, that is reserved for future purposes.
- Stronger RESET circuit.
- Atmega 16U2 replace the 8U2.

"Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduino, moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform; for a comparison with previous versions, see the index of Arduino boards.

Summary

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V

Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

Schematic & Reference Design

EAGLE files: arduino-uno-Rev3-reference-design.zip (NOTE: works with Eagle 6.0 and newer)
Schematic: arduino-uno-Rev3-schematic.pdf
Note: The Arduino reference design can use an Atmega8, 168, or 328, Current models use an ATmega328, but an Atmega8 is shown in the schematic for reference. The pin configuration is identical on all three processors.

Power

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically.
External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.
The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.
The power pins are as follows:

- **VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V.**This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it.
- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND.** Ground pins.

Memory

The ATmega328 has 32 KB (with 0.5 KB used for the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

Input and Output

Each of the 14 digital pins on the Uno can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
- **External Interrupts: 2 and 3.** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- **PWM: 3, 5, 6, 9, 10, and 11.** Provide 8-bit PWM output with the analogWrite() function.

- **SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK).** These pins support SPI communication using the SPI library.
- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function. Additionally, some pins have specialized functionality:

- **TWI: A4 or SDA pin and A5 or SCL pin.** Support TWI communication using the Wire library.

There are a couple of other pins on the board:

- **AREF.** Reference voltage for the analog inputs. Used with analogReference().
- **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

See also the mapping between Arduino pins and ATmega328 ports. The mapping for the Atmega8, 168, and 328 is identical.

Communication

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1).
A SoftwareSerial library allows for serial communication on any of the Uno's digital pins.
The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation for details. For SPI communication, use the SPI library.

Programming

The Arduino Uno can be programmed with the Arduino software (download). Select "Arduino Uno from the **Tools > Board** menu (according to the microcontroller on your board). For details, see the reference and tutorials.
The ATmega328 on the Arduino Uno comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).
You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.
The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available . The ATmega16U2/8U2 is loaded with a DFU bootloader, which can be activated by:

- On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2.
- On Rev2 or later boards: there is a resistor that pulling the 8U2/16U2 HWB line to ground, making it easier to put into DFU mode.

You can then use Atmel's FLIP software (Windows) or the DFU programmer (Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU bootloader). See this user-contributed tutorial for more information.

Automatic (Software) Reset

Rather than requiring a physical press of the reset button before an upload, the Arduino Uno is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload. This setup has other implications. When the Uno is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.
The Uno contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

USB Overcurrent Protection

The Arduino Uno has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

Physical Characteristics

The maximum length and width of the Uno PCB are 2.7 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Four screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.

DATA SHEET MODULE RELAY

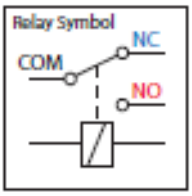
RELAY MODULE

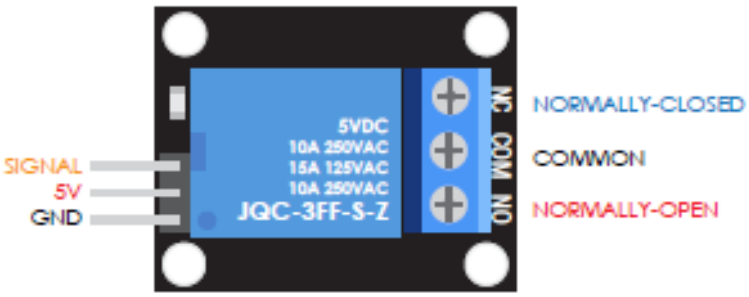
1-Channel Relay Module

Features

- 1 Channel of relay drive
- Transistor switch driven relay
- 5V Operated, 10mA drive current for each channel
- Built-in indicator LEDs for each channel
- Equipped with 250V 10A relay
- SPDT Configuration for all relay channels

PINOUT DIAGRAM





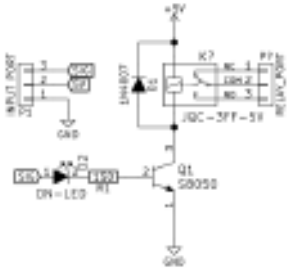
Safety Precaution
Do not connect VCC to GND !!
(short circuit)

1

RELAY MODULE

SCHEMATIC DIAGRAM

1-Channel Relay Module



DATA SHEET *WATERPUMP*

seeed

6V Mini Water Pump

SKU 108990019

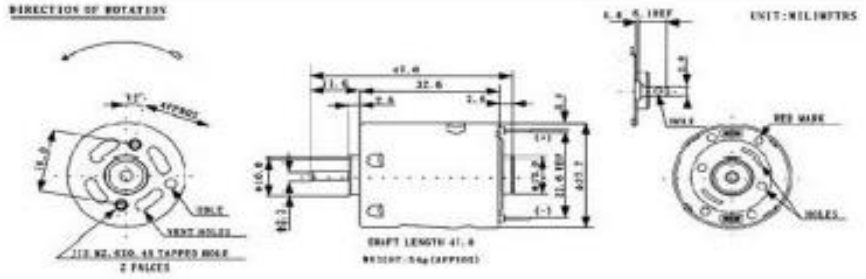
It is lightweight, small size, high efficiency, low consumption and low noise.
It has been used widely, in household include cooking, cleaning, etc.
Please do not test long time with no-load, inside is with plastic leaves, can't suction impurity.

Description

This is lightweight, small size, high efficiency, low consumption and low noise water pump.
It has been used widely, in household include cooking, cleaning, bathing, space heating and water flowers, etc.
Note: Please do not test long time with no-load, inside is with plastic leaves, can't suction impurit

Feature:

self-priming pump
Optimal For Voltage: 6V
Specification :
Working Voltage: 4v-12V
Working Current: 0.8A
Motor Diameter: 27mm
Water Pump lenght: 52mm
Drain Hole: 4mm
Weight:147g



Part List:

1 x water pump

DATA SHEET *INFRARED*

FC-51: IR Infrared Obstacle Detection Sensor Module 2 - 30cm

Giorgio De Nunzio – Giovanni Marsella

0 Premessa: i raggi infrarossi (IR)

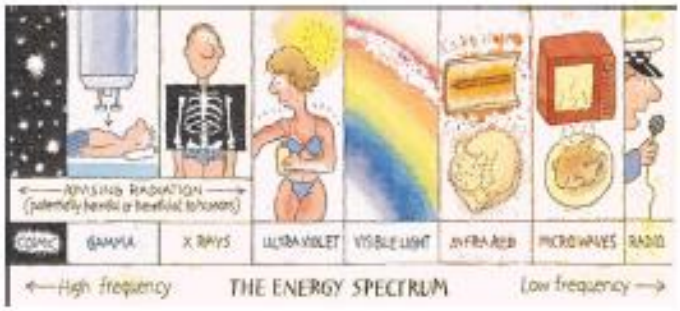


Fig. 0 – Lo spettro delle onde elettromagnetiche, in cui è presente la porzione dei raggi infrarossi (*infrared*) (da http://galileo.phys.virginia.edu/outreach/8thgradesol/RadiationProtectionFrm.htm, immagine proveniente dall'Uranium Information Centre, Melbourne, Australia)

1 Introduzione

La presenza di un oggetto (per esempio, un ostacolo davanti a un robot) può essere rilevata mediante un sistema a raggi infrarossi costituito da un trasmettitore e un ricevitore IR.

Più in dettaglio, un **trasmettitore IR**, o IR LED, invia un segnale a infrarossi ad una particolare lunghezza d'onda compatibile con un **ricevitore IR**, il quale ha il compito di rilevarlo.

Ci sono diversi tipi di sensori IR per diverse tipologie di applicazioni. La tecnologia IR è usata, ad esempio, nei sensori di prossimità per rilevare un oggetto nelle immediate vicinanze, nei sensori di contrasto per individuare un percorso tracciato sul pavimento, o nei sensori di conteggio per contare oggetti che passino davanti a sensore.

2 Principio di funzionamento

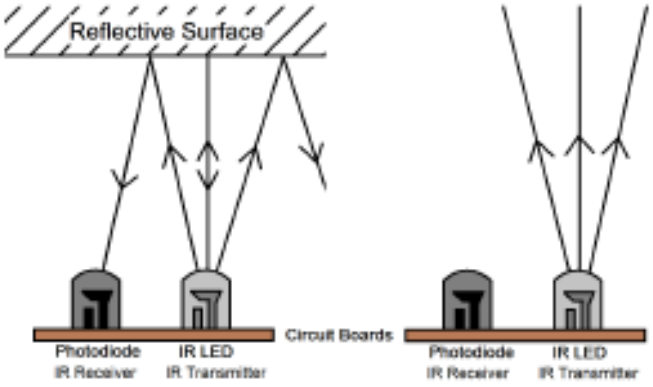


Fig. 1 – Principio di funzionamento di un sensore IR con e senza un oggetto in prossimità.

Il trasmettitore IR invia un segnale infrarosso che, in presenza di una superficie riflettente (soprattutto se di colore bianco), "rimbalza" in varie direzioni, compresa quella lungo la quale la radiazione colpisce il ricevitore IR, il quale cattura il segnale rilevando l'oggetto, e segnalandolo attraverso uno dei suoi pin.

Nel caso di una superficie assorbente (ad esempio di colore nero) il segnale IR è riflesso in minima parte, con la conseguenza che l'oggetto è difficilmente rilevato dal sensore.

2.1 Trasmettitore IR e ricevitore IR

Il trasmettitore IR è un particolare LED che emette radiazioni nel range di frequenza dell'infrarosso, invisibile a occhio nudo. Un LED infrarosso funziona esattamente come un LED nel visibile, tranne che per la lunghezza d'onda emessa.

La tensione di lavoro è di 3.3-5V DC e il consumo di corrente di circa 20mA. Il ricevitore IR, (un fotodiodo), è in grado di rilevare radiazioni infrarosse emesse da un trasmettitore IR. Esteticamente esso è simile ad un LED ma la capsula esterna può essere avvolta da un pellicola di colore scuro.

3 Sensore IR FC-51

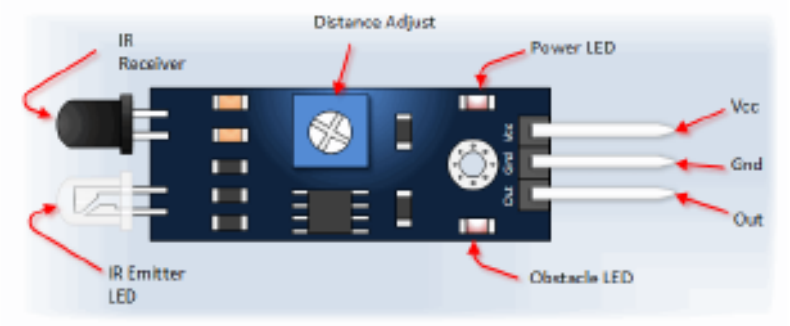


Fig.2 – Il Pin map del sensore FC-51.

Il sensore usato in queste demo è il modello **FC-51**. Si tratta di un sensore economico facilmente reperibile su internet a meno di 2€.

3.1 Configurazione dei Pin e schema elettronico

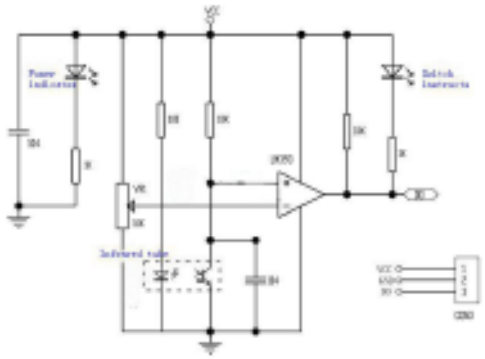


Fig.3 – Schema di un sensore IR FC-51. Il circuito integrato LM393 è un comparatore di tensione a collettore aperto (open-collector) il quale fornisce un'uscita se è presente una resistenza di pull-up tra l'uscita del IC (DO) e l'alimentazione Vcc (R=10KΩ): l'uscita DO è alta se l'oggetto non è rilevabile, bassa se l'oggetto è rilevabile.

Il componente ha tre pin di connessione:

1. Vcc per l'alimentazione 3.3-5V DC;

2. Gnd per il riferimento a massa;
3. Out per il segnale di uscita digitale del sensore.

Il Power LED sul dispositivo si illumina quando esso è alimentato. L'Obstacle LED si illumina quando si individua la presenza di un ostacolo.

L'uscita del sensore è alta (HIGH) se non è rilevato un ostacolo (il LED receiver non riceve segnali riflessi), è bassa (LOW) in caso di presenza di un ostacolo.

Questo sensore rileva **oggetti ad una distanza compresa tra 2 e 30cm**. Mediante il trimmer potenziometrico è possibile calibrare la sensibilità in base all'applicazione e alle condizioni ambientali (ad esempio la luminosità). **Ruotando il potenziometro in senso antiorario, la distanza alla quale il sensore rileva l'oggetto diminuisce, ruotandolo in senso orario tale distanza aumenta.**

E' opportuno notare che si tratta di un dispositivo piuttosto "rozzo", per cui può capitare che un ostacolo non sia rilevato se di colore scuro o se comunque assorbe le linghezze d'onda IR, e che la precisione e la portata effettiva dipendono dalla qualità del sensore (molto variabile) e dal materiale di cui è fatto l'oggetto da rilevare.

4 Alcune demo

4.1 Test IR sensor FC-51 mediante terminale seriale (Demo 01)

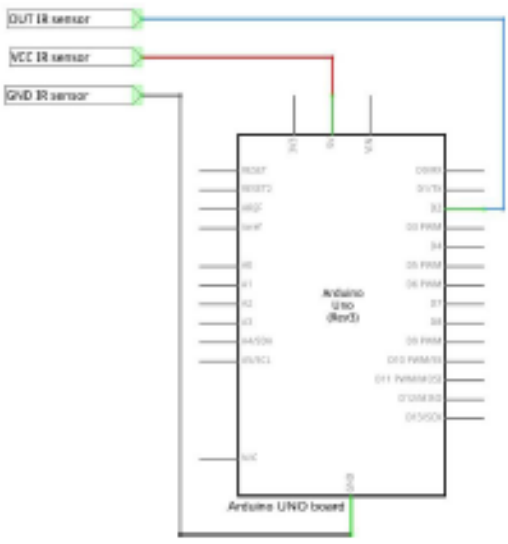


Fig.4 – Schema della demo 1.

Nella prima demo, mediante la connessione tra la porta seriale di Arduino e il PC, leggeremo l'informazione sul rilevamento dell'oggetto.

Diamo uno sguardo ai passaggi richiesti da questa demo:

1. Colleghiamo il pin d'uscita del sensore al pin digitale 2 di Arduino che chiameremo **IR**.
2. La funzione **setup()** è eseguita una sola volta prima della funzione loop. Inseriamo qui il codice di inizializzazione che abilita la porta seriale di Arduino e imposta il pin digitale 2 come un ingresso.
3. La funzione **loop()** è la funzione principale ed è ciclicamente ripetuta finché non si spegne la scheda Arduino. Convertiamo in linguaggio C il funzionamento del circuito elettronico analizzato in precedenza. Salviamo nella variabile **detection** il valore prelevato dal pin IR mediante la funzione **digitalRead**: se il valore è basso allora c'è un oggetto davanti al sensore, altrimenti non c'è (o non è rilevabile).

Ecco il codice:

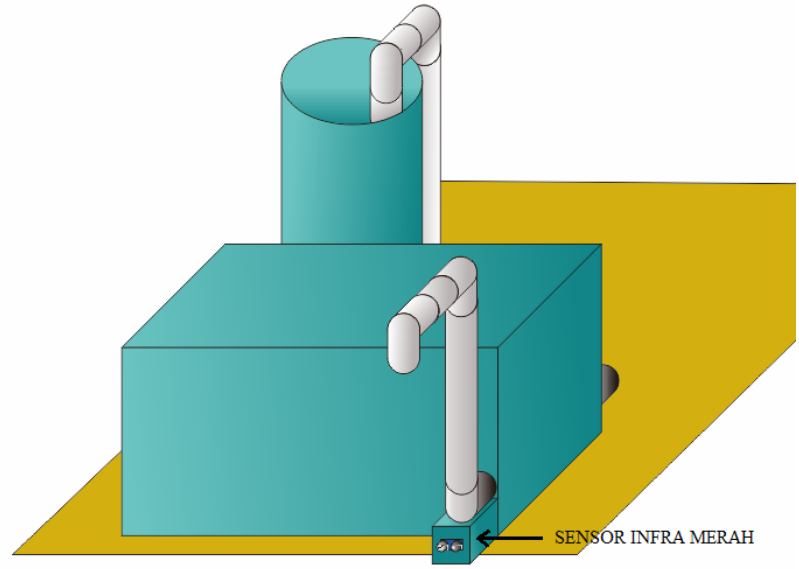
```
#define IR 2
int detection;
void setup() {
  Serial.begin(9600);
  pinMode(IR, INPUT);
}
void loop() {
  detection = digitalRead(IR);
  if(detection == LOW){
    Serial.print("There is an obstacle!\n");
  }
  else{
    Serial.print("No obstacle!\n");
  }
  delay(500);     // in ms
}
```

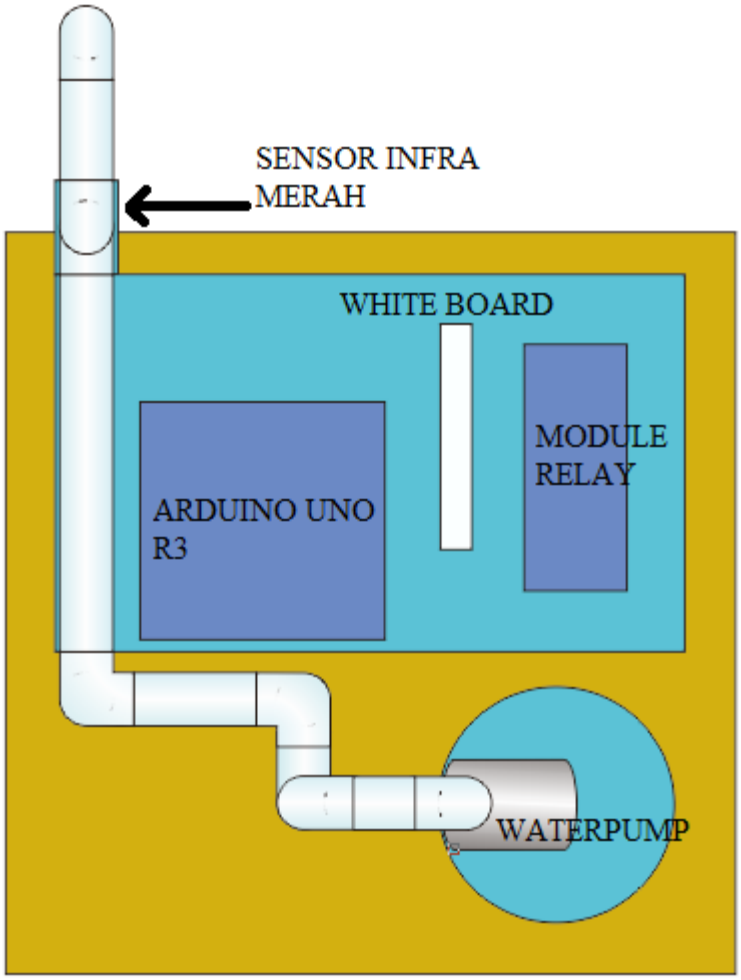
LISTING PROGRAM KESELURUHAN

```
const int pinSensor = 2;
const int pinPompa = 8;

int bacaSensor;
void setup() {
  // put your setup code here, to run once:
  // pin sensor sebagai input
  pinMode(pinSensor, INPUT);
  // pin pompa sebagai output
  pinMode(pinPompa, OUTPUT);

}

void loop() {
  // put your main code here, to run repeatedly:
  bacaSensor = digitalRead(pinSensor);
  if (bacaSensor == LOW)
  {
  // pompa air menyala dan air wastafel mengalir
  digitalWrite(pinPompa, LOW);
  // delay dalam satuan mikrodetik
  delay(1000);
  }
```

```
  else
  {
  // pompa air mati, air berhenti mengalir
  digitalWrite(pinPompa, HIGH);
  }
  delay(200);

}
```

SKEMA RANGKAIAN KESELURUHAN





DAFTAR KOMPONEN DAN HARGA KOMPONEN

No	Komponen	Harga
1	Arduino Uno R3	Rp 110.000,00
2	Sensor Infrared	Rp 25.000,00
3	Relay	Rp 10.000,00
4	Waterpump	Rp 40.000,00
5	Adapter -+9 Volt	Rp 45.000,00
6	Kabel Jumper	Rp 20.000,00
7	Kabel Data Arduino	Rp 10.000,00
8	Kabel Data Usb	Rp 10.000,00
9	Baut dan Mur Arduino	Rp 5.000,00

10	Akrilik alat	Rp 50.000,00
11	Lem Akrilik	Rp 11.000,00
12	Pipa 5/8	Rp 15.000,00
13	Pipa L 5/8	Rp 6.000,00
Total Harga		Rp 357.000,00

FOTO ALAT