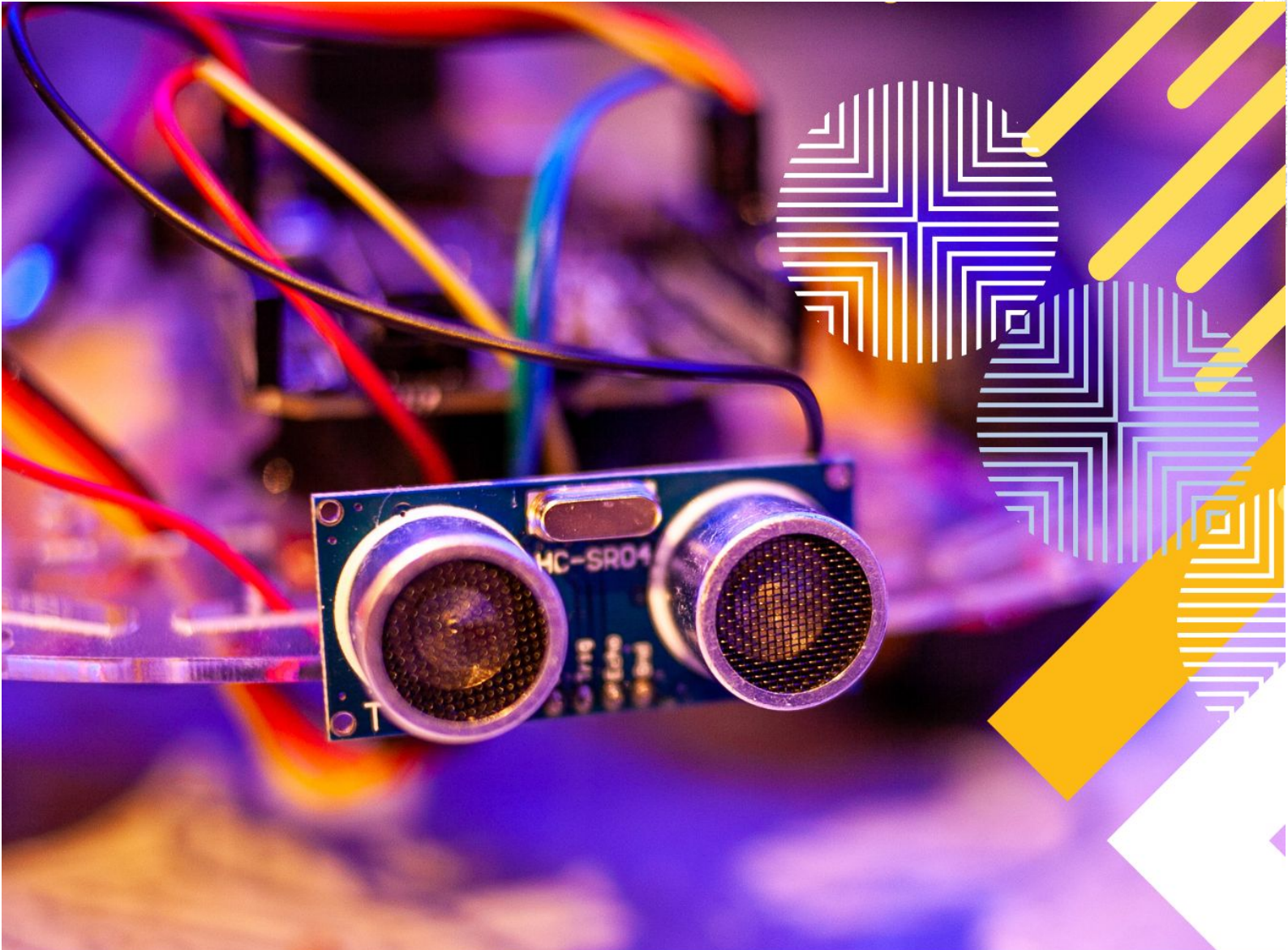




Pemrograman Arduino untuk Pemula: Proyek Sederhana dan Edukatif

Ir. Alfian Ma'arif, ST, M.Eng.,
Shandy Auliya Ma'arief,
Ir. Purwono, S.Kom., M.Kom



Penerbit Yayasan Peneliti Teknologi Teknik Indonesia

Pemrograman Arduino untuk Pemula: Proyek Sederhana dan Edukatif

Penulis:

Ir. Alfian Ma'arif, ST, M.Eng
Shandy Auliya Ma'arif
Ir. Purwono, S.Kom., M.Kom

Penerbit Yayasan Peneliti Teknologi Teknik Indonesia

Pemrograman Arduino untuk Pemula: Proyek Sederhana dan Edukatif

Oleh:

Ir. Alfian Ma'arif, ST, M.Eng
Shandy Auliya Ma'arif
Ir. Purwono, S.Kom., M.Kom

Hak Cipta © 2025 pada penulis,
Editor: Khoirun Nisa, S.Kom., M.Kom.

Hak Cipta dilindungi oleh undang-undang. Dilarang memperbanyak atau memindahkan Sebagian atau keseluruhan isi buku ini dalam bentuk apapun, secara elektronik maupun mekanis, termasuk mefotokopi, merekam, atau dengan Teknik perekaman lainnya, tanpa izin tertulis dari penerbit.

Diterbitkan oleh Penerbit Yayasan Peneliti Teknologi Teknik Indonesia
Jl. Empu Sedah No. 12, Pringwulung, Condongcatur, Kec. Depok,
Kabupaten Sleman, Daerah Istimewa Yogyakarta 55281, Indonesia

Pemrograman Arduino untuk Pemula: Proyek Sederhana dan Edukatif
-edisi Pertama – Purwokerto: Penerbit Yayasan Peneliti Teknologi Teknik
Indonesia, 2025

xvi + 165 hlm, 1 Jil: 23 cm

ISBN: In Process

KATA PENGANTAR

Bismillahirrahmaanirrahiim,

Assalaamu'alaikum warahmatullaahi wabarakaatuh

Segala puji bagi Allah SWT atas segala nikmat yang telah diberikan kepada penulis sehingga dapat menyelesaikan buku Arduino untuk pemula ini dengan baik. Selamat datang di dunia Arduino! Buku ini saya tulis dengan tujuan untuk memperkenalkan dan memudahkan para pemula dalam memahami serta mengaplikasikan platform Arduino dalam berbagai proyek kreatif.

Arduino merupakan platform open-source yang sangat populer di kalangan penggemar elektronik dan teknologi. Dengan Arduino, Anda dapat membuat berbagai proyek menarik, mulai dari lampu berkedip sederhana hingga robot yang kompleks. Buku ini dirancang untuk menjadi panduan praktis yang akan membantu Anda mengenal dasar-dasar Arduino dan memulai langkah pertama dalam perjalanan Anda sebagai seorang maker.

Saya memahami bahwa memulai sesuatu yang baru bisa menjadi tantangan tersendiri. Oleh karena itu, buku ini disusun dengan pendekatan yang mudah dipahami dan diikuti, lengkap dengan ilustrasi, contoh kode, dan penjelasan langkah demi langkah. Setiap bab dalam buku ini akan membimbing Anda melalui berbagai konsep dasar dan proyek-proyek sederhana yang akan memperkuat pemahaman Anda tentang cara kerja Arduino.

Saya mengucapkan terima kasih kepada semua pihak yang telah mendukung proses penulisan buku ini, terutama komunitas Arduino yang selalu berbagi ilmu dan pengalaman. Saya juga berterima kasih kepada Anda, pembaca, yang telah memilih buku ini sebagai panduan dalam petualangan Anda bersama Arduino.

Semoga buku ini dapat memberikan pengetahuan dan inspirasi yang Anda butuhkan untuk memulai dan berkembang di dunia Arduino. Selamat belajar dan selamat berkarya!

Salam hangat,
Yogyakarta, 18 Juni 2025

Penulis

DAFTAR ISI

COVER	II
KATA PENGANTAR.....	IV
DAFTAR ISI.....	V
DAFTAR GAMBAR	VII
DAFTAR TABEL	X
BAB 1 PENGENALAN ARDUINO.....	1
A. GAMBARAN UMUM ARDUINO	1
B. JENIS PAPAN.....	2
C. DESKRIPSI PAPAN ARDUINO.....	3
D. INSTALASI	6
E. STRUKTUR PROGRAM.....	10
F. TIPE DATA	12
G. VARIABEL DAN KONSTANTA	16
H. OPERATOR.....	17
I. KONTROL STATEMENT	20
J. LOOP.....	21
K. FUNGSI.....	22
L. STRING.....	24
M. OBJEK STRING	32
N. WAKTU	35
O. ARRAYS.....	35
BAB 2 FUNGSI LIBRARY ARDUINO	41
A. FUNGSI I/O.....	41
B. FUNGSI I/O TINGKAT LANJUT	45
C. FUNGSI CHARACTER	47
D. MATH LIBRARY	54
E. FUNGSI TRIGONOMETRI	61
BAB 3 ARDUINO TINGKAT LANJUT	63
A. ARDUINO DUE & ZERO	63
B. PULSE WIDTH MODULATION.....	66
C. RANDOM NUMBER	68
D. INTERRUPTS	70

E. KOMUNIKASI.....	73
F. SERIAL PERIPHERAL INTERFACE.....	81
BAB 4 PROYEK ARDUINO.....	85
A. LED BERKEDIP	85
B. LED REDUP	88
C. MEMBACA TEGANGAN ANALOG	90
D. GRAFIK BATANG LED.....	93
E. KEYBOARD LOGOUT.....	96
F. KEYBOARD MESSAGE	101
G. MOUSE BUTTON CONTROL	103
H. KEYBOARD SERIAL	105
BAB 5 SENSOR ARDUINO.....	108
A. SENSOR KELEMBABAN.....	108
B. SENSOR SUHU	112
C. DETEKTOR / SENSOR AIR	114
D. SENSOR PIR	116
E. SENSOR ULTRASONIK.....	120
F. SAKLAR PENGHUBUNG.....	123
BAB 6 MOTOR DC	127
A. MOTOR DC	127
B. MOTOR SERVO	133
C. MOTOR STEPPER	138
BAB 7 PROYEK LANJUTAN ARDUINO	143
A. PERPUSTAKAAN NADA	143
B. KOMUNIKASI NIRKABEL	148
C. KOMUNIKASI JARINGAN.....	152
PENUTUP.....	159

DAFTAR GAMBAR

Gambar 1. 1 Arduino uno r3 dan software arduino ide.....	2
Gambar 1. 2 Arduino Uno R3	4
Gambar 1. 3 USB standar A to B	6
Gambar 1. 4 USB A to mini-B	6
Gambar 1. 5 Unzip file arduino.....	7
Gambar 1. 6 Folder penyimpanan aplikasi Arduino IDE.....	7
Gambar 1. 7 Tampilan awal memulai proyek di Arduino IDE	8
Gambar 1. 8 Membuka contoh proyek.....	8
Gambar 1. 9 Memilih <i>board</i> yang digunakan	9
Gambar 1. 10 Memilih serial port.....	9
Gambar 1. 11 Fungsi simbol di Arduino IDE	10
Gambar 1. 12 Tampilan struktur fingsi di Arduino IDE	11
Gambar 1. 13 Tabel Ascii.....	13
Gambar 1. 14 Struktur pengambilan keputusan if...else	21
Gambar 1. 15 Struktur pengambilan keputusan pernyataan loop	22
Gambar 1. 16 Definisi suatu fungsi.....	23
Gambar 1. 17 Pemosisian array	36
Gambar 2. 1 Posisi pin analog <i>reference</i>	46
Gambar 3. 1 Papan Arduino Due	63
Gambar 3. 2 Papan Arduino Zero	65
Gambar 3. 3 Sinyal PWM	66
Gambar 3. 4 Posisi pin untuk PWM.....	67
Gambar 3. 5 Posisi signifikansi bit dalam bilangan biner.....	70
Gambar 3. 6 Gambaran intrupsi bekerja	71
Gambar 3. 7 Gambaran komunikasi paralel.....	74
Gambar 3. 8 Jalur data Bit.....	75
Gambar 3. 9 Cara kerja inter integrated circuit.....	78
Gambar 3. 10 Koneksi antar arduino	83
Gambar 4. 1 Rangkaian LED berkedip	85
Gambar 4. 2 Pin pada LED	86

Gambar 4. 3 Resistor yang ditebuk	86
Gambar 4. 4 Membuka <i>new sketch</i> di Arduino IDE	87
Gambar 4. 5 Rangkaian LED Redup	88
Gambar 4. 6 Rangkaian membaca tegangan analog dari potensiometer	91
Gambar 4. 7 Skema standar potensiometer dan potensiometer	91
Gambar 4. 8 Menunjukkan letak icon serial monitor	93
Gambar 4. 9 Rangkaian grafik batang LED	94
Gambar 4. 10 Grafik batang LED 10 segment	94
Gambar 4. 11 Rangkaian keyboard logout	97
Gambar 4. 12 Letak folder <i>library</i> untuk menambahkan ke Arduino IDE	97
Gambar 4. 13 Rangkaian <i>keyboard message</i>	101
Gambar 4. 14 Rangkaian mouse button control	103
Gambar 5. 1 Sensor DHT-22	108
Gambar 5. 2 Rangkaian proyek sensor kelembaban	109
Gambar 5. 3 Membuka <i>Sketch</i> baru di Arduino IDE	110
Gambar 5. 4 Sensor suhu LM35	112
Gambar 5. 5 Rangkaian proyek sensor suhu LM35	113
Gambar 5. 6 Sensor air	114
Gambar 5. 7 Rangkaian proyek sensor suhu	115
Gambar 5. 8 Sensor PIR	117
Gambar 5. 9 Bagian dalam sensor PIR	117
Gambar 5. 10 Pengaturan pada sensor PIR	118
Gambar 5. 11 Rangkaian proyek sensor PIR	118
Gambar 5. 12 Pengaturan waktu tunda dan sensitifitas sensor PIR	120
Gambar 5. 13 Sensor ultrasonik dan prinsip kerja	121
Gambar 5. 14 Rangkaian proyek sensor ultrasonik	122
Gambar 5. 15 Saklar tombol	123
Gambar 5. 16 Rangkaian digital resistor pull-down	124
Gambar 5. 17 Rangkaian proyek saklar penghubung	125
Gambar 6. 1 Motor DC 5v	127
Gambar 6. 2 Rangkaian proyek Motor DC	128
Gambar 6. 3 Rangkaian proyek untuk kecepatan motor DC	129

Gambar 6. 4 Tampilan kecepatan pada serial monitor	130
Gambar 6. 5 IC H-Bridge L298	131
Gambar 6. 6 Rangkaian antarmuka motor DC ke Arduino	131
Gambar 6. 7 Servo.....	134
Gambar 6. 8 Bagian dalam servo	134
Gambar 6. 9 Penggunaan sudut putaran pada servo	135
Gambar 6. 10 Rangkaian pengkabelan proyek servo	136
Gambar 6. 11 Membuat sketch baru di Arduino IDE	136
Gambar 6. 12 Rangkaian seri resistor	137
Gambar 6. 13 Pin pada potensimeter	138
Gambar 6. 14 Motor stepper	139
Gambar 6. 15 Bagian dalam pada motor stepper	140
Gambar 6. 16 Motor stepper bipolar kecil	140
Gambar 6. 17 Rangkaian proyek motor stepper.....	141
Gambar 7. 1 Rangkaian proyek nada di arduino.....	144
Gambar 7. 2 Membuat sketch baru di Arduino IDE	144
Gambar 7. 3 Menambahkan tab baru di Arduino IDE	145
Gambar 7. 4 Modul penerima dan pengirim	149
Gambar 7. 5 Rangkaian untuk komunikasi nirkabel.....	150
Gambar 7. 6 Lokasi folder <i>library</i> pada Arduino.....	151
Gambar 7. 7 Modul adafruit cc3000	152
Gambar 7. 8 Rangkaian proyek komunikasi jaringan.....	153
Gambar 7. 9 Tampilan hasil dari program inteface	158

DAFTAR TABEL

Tabel 1. 1 Papan Arduino berbasis mikrokontroler ATMEGA328	2
Tabel 1. 2 Papan Arduino berbasis mikrokontroler ATMEGA32u4	3
Tabel 1. 3 Papan Arduino berbasis mikrokontroler AT91SAM3X8E	3
Tabel 1. 4 Tipe data pada pemrograman arduino	12
Tabel 1. 5 Operator aritmatika	17
Tabel 1. 6 Operator perbandingan	17
Tabel 1. 7 Operator boolean	18
Tabel 1. 8 Operator bitwise	19
Tabel 1. 9 Operator gabungan	19
Tabel 1. 10 Pernyataan dan struktur kontrol if...else	21
Tabel 1. 11 Beberapa fungsi dan deskripsi loop.....	22
Tabel 1. 12 Daftar function to manipulation string arrays	27
Tabel 1. 13 Fungsi dan deskripsi waktu di Arduini IDE	35
Tabel 1. 14 Konsep dan dekskripsi array	40
Tabel 2. 1 Library fungsi character	48
Tabel 2. 2 Daftar librari makro untuk math.h.....	54
Tabel 2. 3 deskripsi dan sungsi dari library math.h.....	56
Tabel 6. 1 Arah putaran motor berdasarkan niali IN1 dan IN2	132

BAB 1

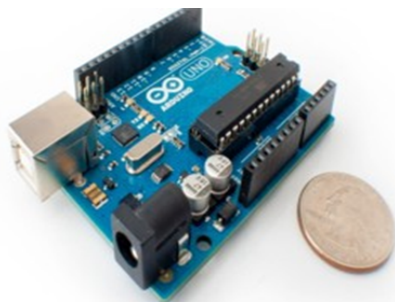
PENGENALAN ARDUINO

A. Gambaran Umum Arduino

Arduino adalah platform prototipe (open-source) berdasarkan perangkat keras dan perangkat lunak yang mudah digunakan. Ini terdiri dari papan sirkuit, yang dapat diprogram (disebut sebagai mikrokontroler) dan perangkat lunak siap pakai yang disebut Arduino IDE (Integrated Development Environment), yang digunakan untuk menulis dan mengunggah kode komputer ke papan fisik.

Fitur utamanya adalah sebagai berikut

- Papan Arduino dapat membaca sinyal input analog atau digital dari sensor yang berbeda dan mengubahnya menjadi output seperti mengaktifkan motor, menyalakan / mematikan LED, terhubung ke cloud dan banyak tindakan lainnya.
- Anda dapat mengontrol fungsi papan Anda dengan mengirimkan satu set instruksi ke mikrokontroler di papan melalui Arduino IDE (disebut sebagai perangkat lunak pengunggahan).
- Tidak seperti kebanyakan papan sirkuit yang dapat diprogram sebelumnya, Arduino tidak memerlukan perangkat keras tambahan (disebut programmer) untuk memuat kode baru ke papan. Anda cukup menggunakan kabel USB.
- Selain itu, Arduino IDE menggunakan versi C ++ yang disederhanakan, sehingga lebih mudah untuk belajar memprogram.
- Akhirnya, Arduino menyediakan faktor bentuk standar yang memecah fungsi mikrokontroler menjadi paket yang lebih mudah diakses.



Gambar 1. 1 Arduino uno r3 dan software arduino ide

B. Jenis papan

Berbagai macam papan Arduino tersedia tergantung pada mikrokontroler yang berbeda yang digunakan. Namun, semua papan Arduino memiliki satu kesamaan: mereka diprogram melalui Arduino IDE.

Perbedaannya didasarkan pada jumlah input dan output (jumlah sensor, LED, dan tombol yang dapat Anda gunakan pada satu papan), kecepatan, tegangan operasi, faktor bentuk, dll. Beberapa papan dirancang untuk disematkan dan tidak memiliki antarmuka pemrograman (perangkat keras), yang perlu Anda beli secara terpisah. Beberapa dapat berjalan langsung dari baterai 3.7V, yang lain membutuhkan setidaknya 5V.

Berikut adalah daftar papan Arduino berbeda yang tersedia.

Tabel 1. 1 Papan Arduino berbasis mikrokontroler ATMEGA328

Nama Papan	Volt Operasi	Kecepatan Jam	I/O digital	Input Analog	PWM	UART	Antarmuka Pemrograman
Arduino Uno R3	5V	16MHz	14	6	6	1	USB melalui ATmega16U2
Arduino Uno R3 SMD	5V	16MHz	14	6	6	1	USB melalui ATmega16U2
Papan Merah	5V	16MHz	14	6	6	1	USB melalui FTDI
Arduino Pro 3,3v/8 MHz	3,3V	8MHz	14	6	6	1	Header yang Kompatibel dengan FTDI
Arduino Pro 5V/16MHz	5V	16MHz	14	6	6	1	Header yang Kompatibel dengan FTDI
Arduino mini 05	5V	16MHz	14	8	6	1	Header
Arduino Pro mini 3.3v / 8mhz	3,3V	8MHz	14	8	6	1	Header yang Kompatibel dengan FTDI
Arduino Pro mini 5v / 16mhz	5V	16MHz	14	8	6	1	Header yang Kompatibel dengan FTDI
Arduino Ethernet	5V	16MHz	14	6	6	1	Header yang Kompatibel dengan FTDI
Arduino Fio	3,3V	8MHz	14	8	6	1	Header yang Kompatibel dengan FTDI

Nama Papan	Volt Operasi	Kecepatan Jam	I/O digital	Input Analog	PWM	UART	Antarmuka Pemrograman
Papan utama LilyPad Arduino 328	3,3V	8MHz	14	6	6	1	Header yang Kompatibel dengan FTDI
Papan sederhana LilyPad Arduino	3,3V	8MHz	9	4	5	0	Header yang Kompatibel dengan FTDI

Tabel 1. 2 Papan Arduino berbasis mikrokontroler ATMEGA32u4

Nama Papan	Volt Operasi	Kecepatan Jam	I/O digital	Input Analog	PWM	UART	Antarmuka Pemrograman
Arduino Leonardo	5V	16MHz	20	12	7	1	USB asli
Untuk mikro 5V/16MHz	5V	16MHz	14	6	6	1	USB asli
Untuk mikro 3.3V/8MHz	5V	16MHz	14	6	6	1	USB asli
LilyPad Arduino USB	3,3V	8MHz	14	6	6	1	USB asli

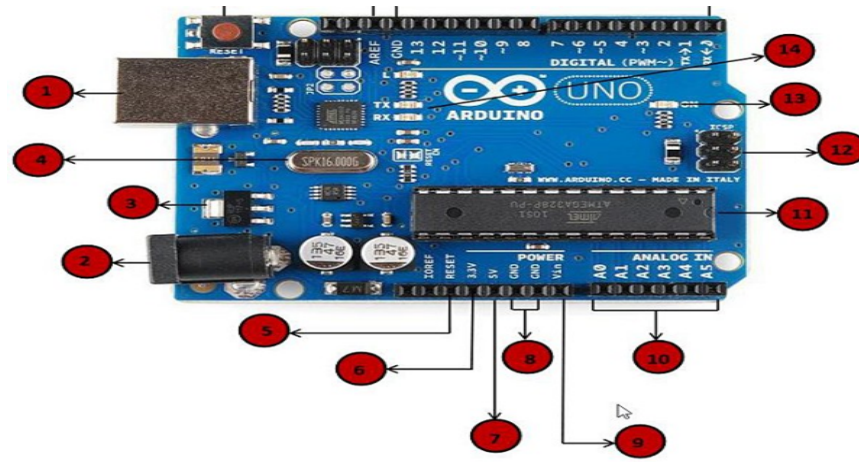
Tabel 1. 3 Papan Arduino berbasis mikrokontroler AT91SAM3X8E

Nama Papan	Volt Operasi	Kecepatan Jam	I/O digital	Input Analog	PWM	UART	Antarmuka Pemrograman
Arduino Mega 2560 R3	3,3V	84MHz	54	12	12	4	USB asli

C. Deskripsi papan Arduino

Dalam bab ini, kita akan belajar tentang berbagai komponen pada papan Arduino. Kami akan mempelajari papan Arduino UNO karena ini adalah papan paling populer di keluarga papan Arduino. Selain itu, ini adalah papan terbaik untuk memulai dengan elektronik dan pengkodean. Beberapa papan terlihat

sedikit berbeda dari yang diberikan di bawah ini, tetapi sebagian besar Arduino memiliki mayoritas komponen ini yang sama.



Gambar 1. 2 Arduino Uno R3

USB Daya

- 1 Papan Arduino dapat didukung dengan menggunakan kabel USB dari komputer Anda. Yang perlu Anda lakukan adalah menghubungkan kabel USB ke koneksi USB (1).

Daya (Jack Barel)

- 2 Papan Arduino dapat ditenagai langsung dari catu daya listrik AC dengan menghubungkannya ke Barrel Jack (2).

Regulator Tegangan

- 3 Fungsi regulator tegangan adalah untuk mengontrol tegangan yang diberikan ke papan Arduino dan menstabilkan tegangan DC yang digunakan oleh prosesor dan elemen lainnya.

Osilator Kristal

- 4 Osilator kristal membantu Arduino dalam menangani masalah waktu. Bagaimana Arduino menghitung waktu? Jawabannya adalah, dengan menggunakan osilator kristal. Angka yang tercetak di atas kristal Arduino adalah 16.000H9H. Ini memberitahu kita bahwa frekuensinya adalah 16.000.000 Hertz atau 16 MHz.

Atur Ulang Arduino

- 5, 17 Anda dapat mereset papan Arduino Anda, yaitu memulai program Anda dari awal. Anda dapat mereset papan UNO dengan dua cara. Pertama dengan menggunakan tombol reset (17) pada board. Kedua, Anda dapat menghubungkan tombol reset eksternal ke pin Arduino berlabel RESET (5).

6, 7, 8, 9	Pin (3.3, 5, GND, Vin)
	• 3.3V (6) - Suplai 3.3 volt output 5V • (7) - Suplai 5 volt output • Sebagian besar komponen yang digunakan dengan papan Arduino berfungsi dengan baik dengan 3,3 volt dan 5 volt. • GND (8) (Ground) - Ada beberapa pin GND pada Arduino, yang dapat digunakan untuk membumikan sirkuit Anda. • Vin (9) - Pin ini juga dapat digunakan untuk menyalakan papan Arduino dari sumber daya eksternal, seperti catu daya listrik AC.
	Pin analog
	10 Papan Arduino UNO memiliki enam pin input analog A0 hingga A5. Pin ini dapat membaca sinyal dari sensor analog seperti sensor kelembaban atau sensor suhu dan mengubahnya menjadi nilai digital yang dapat dibaca oleh mikroprosesor.
	Mikrokontroler utama Setiap papan Arduino memiliki mikrokontroler sendiri (11). Anda dapat menganggapnya sebagai otak papan Anda. IC utama (integrated circuit) pada Arduino sedikit berbeda dari board to board. Mikrokontroler biasanya dari Perusahaan ATMEL. Anda harus tahu IC apa yang dimiliki papan Anda sebelum memuat program baru dari Arduino IDE. Informasi ini tersedia di bagian atas IC. Untuk detail lebih lanjut tentang konstruksi dan fungsi IC, Anda dapat merujuk ke lembar data.
12	Pin ICSP
	Sebagian besar, ICSP (12) adalah AVR, header pemrograman kecil untuk Arduino yang terdiri dari MOSI, MISO, SCK, RESET, VCC, dan GND. Hal ini sering disebut sebagai SPI (Serial Peripheral Interface), yang dapat dianggap sebagai "ekspansi" dari output. Sebenarnya, Anda memperbudak perangkat output ke master bus SPI.
13	Indikator LED daya
14	LED TX dan RX
	Di papan Anda, Anda akan menemukan dua label: TX (mentransmisikan) dan RX (menerima). Mereka muncul di dua tempat di papan Arduino UNO. Pertama, pada pin digital 0 dan 1, untuk menunjukkan pin yang bertanggung jawab untuk komunikasi serial. Kedua, led TX dan RX (13). Led TX berkedip dengan kecepatan berbeda saat mengirim data serial. Kecepatan flashing tergantung pada baud rate yang digunakan oleh board. RX berkedip selama proses penerimaan.
15	I/O digital
	Papan Arduino UNO memiliki 14 pin I/O digital (15) (6 di antaranya memberikan keluaran PWM (Pulse Wide Modulation). Pin ini dapat dikonfigurasi untuk berfungsi sebagai pin digital masukan untuk membaca nilai logika (0 atau 1) atau sebagai digital pin keluaran untuk menggerakkan modul yang berbeda seperti LED, relay, dll. Pin berlabel "~" dapat digunakan untuk menghasilkan PWM.
16	AREF
	AREF adalah singkatan Referensi Analog. Kadang-kadang digunakan untuk mengatur tegangan referensi eksternal (antara 0 dan 5 Volt) sebagai batas atas untuk pin input analog.

D. Instalasi

Setelah mempelajari tentang bagian-bagian utama papan Arduino UNO, kita siap untuk mempelajari cara mengatur Arduino IDE. Setelah kami mempelajari ini, kami akan siap untuk mengunggah program kami di papan Arduino.

Pada bagian ini, kita akan belajar dalam langkah-langkah mudah, cara mengatur Arduino IDE di komputer kita dan menyiapkan papan untuk menerima program melalui kabel USB.

Langkah 1 - Pertama Anda harus memiliki papan Arduino Anda (Anda dapat memilih papan favorit Anda) dan kabel USB. Jika Anda menggunakan Arduino UNO, Arduino Duemilanove, Nano, Arduino Mega 2560, atau Diecimila, Anda memerlukan kabel USB standar (steker A ke steker B), jenis yang akan Anda sambungkan ke printer USB seperti yang ditunjukkan pada gambar berikut.



Gambar 1. 3 USB standar A to B

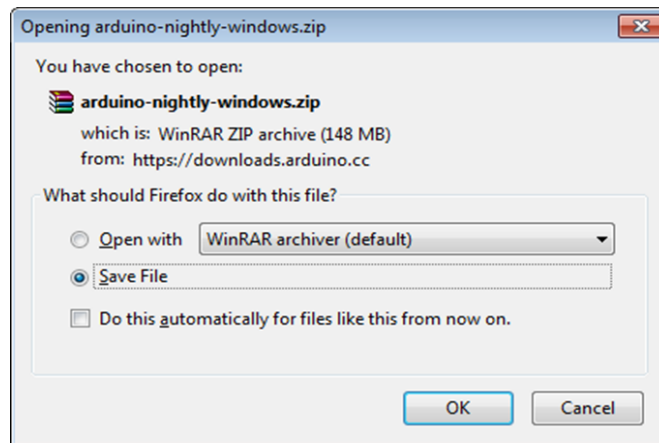
Jika Anda menggunakan Arduino Nano, Anda memerlukan kabel A ke Mini-B seperti yang ditunjukkan pada gambar berikut.

Gambar 1. 4 USB A to mini-B



Langkah 2 - Unduh Perangkat Lunak Arduino IDE.

Anda bisa mendapatkan berbagai versi Arduino IDE dari [halaman Unduh](#) di situs web resmi Arduino. Anda harus memilih perangkat lunak Anda, yang kompatibel dengan sistem operasi Anda (Windows, iOS, atau Linux). Setelah unduhan file selesai, unzip file tersebut.



Gambar 1. 5 Unzip file arduino

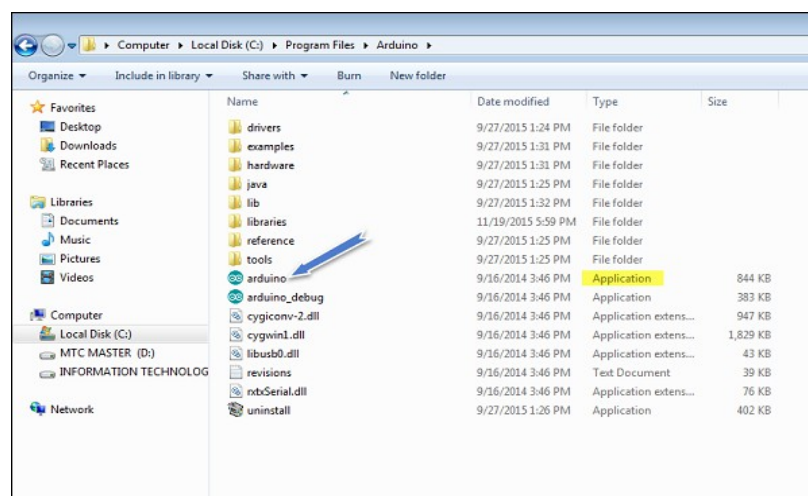
Langkah 3 - Nyalakan papan Arduino anda.

Arduino Uno, Mega, Duemilanove, dan Arduino Nano secara otomatis mengambil daya dari koneksi USB ke komputer atau catu daya eksternal. Jika Anda menggunakan Arduino Diecimila, Anda harus memastikan bahwa papan dikonfigurasi untuk mengambil daya dari koneksi USB. Sumber listrik dipilih dengan jumper, sepotong kecil plastik yang dipasang pada dua dari tiga pin antara USB dan colokan listrik. Periksa apakah ada pada dua pin yang paling dekat dengan port USB.

Hubungkan papan Arduino ke komputer Anda menggunakan kabel USB. LED daya hijau (berlabel PWR) akan menyala.

Langkah 4 - Luncurkan Arduino IDE.

Setelah perangkat lunak Arduino IDE Anda diunduh, Anda perlu mengekstrak foldernya. Di dalam folder tersebut, Anda dapat menemukan ikon aplikasi dengan label tak terhingga (application.exe). Klik dua kali ikon tersebut untuk memulai IDE.



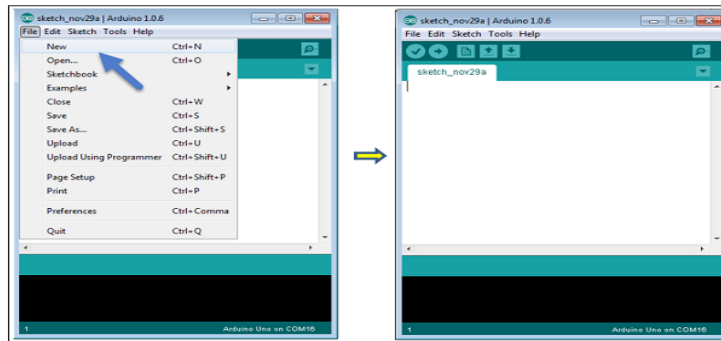
Gambar 1. 6 Folder penyimpanan aplikasi Arduino IDE

Langkah 5 - Buka proyek pertama Anda.

Setelah perangkat lunak dimulai, Anda memiliki dua opsi yaitu :

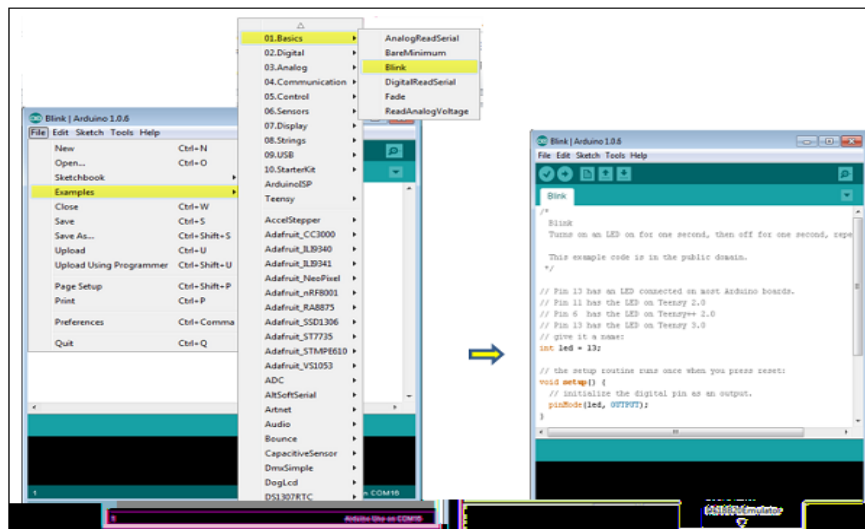
- Buat proyek baru.
- Buka contoh proyek yang sudah ada.

Untuk membuat proyek baru, pilih File → **Baru** .



Gambar 1. 7 Tampilan awal memulai proyek di Arduino IDE

Untuk membuka contoh proyek yang sudah ada, pilih File → **Contoh** → **Dasar** → **Blink**.



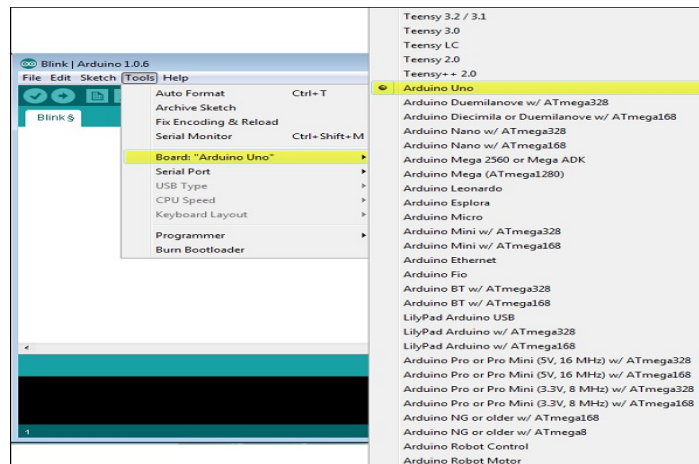
Gambar 1. 8 Membuka contoh proyek

Di sini, kami memilih salah satu contoh saja dengan nama **Blink** . Ini menyalakan dan mematikan LED dengan beberapa waktu tunda. Anda dapat memilih contoh lain dari daftar.

Langkah 6 - Pilih papan Arduino Anda.

Untuk menghindari kesalahan saat mengunggah program Anda ke papan, Anda harus memilih nama papan Arduino yang benar, yang cocok dengan papan yang terhubung ke komputer Anda.

Buka **tools** → **board** dan pilih papan Anda.



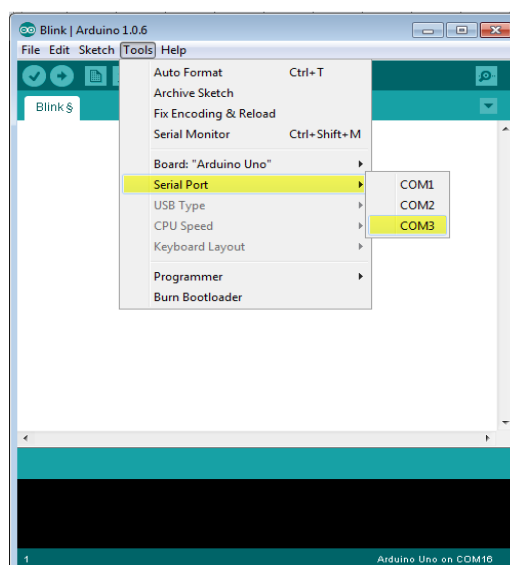
Gambar 1. 9 Memilih board yang digunakan

Di sini, kami telah memilih papan Arduino Uno sesuai dengan tutorial kami, tetapi Anda harus memilih nama yang cocok dengan papan yang Anda gunakan.

Langkah 7 - Pilih port serial Anda.

Pilih perangkat serial papan Arduino. Buka menu **Tools** → **Serial Port**. Ini mungkin COM3 atau lebih tinggi (COM1 dan COM2 biasanya disediakan untuk port serial perangkat keras). Untuk mengetahuinya, Anda dapat melepaskan papan Arduino Anda dan membuka kembali menu, entri yang hilang harus dari papan Arduino. Sambungkan kembali papan dan pilih port serial itu.

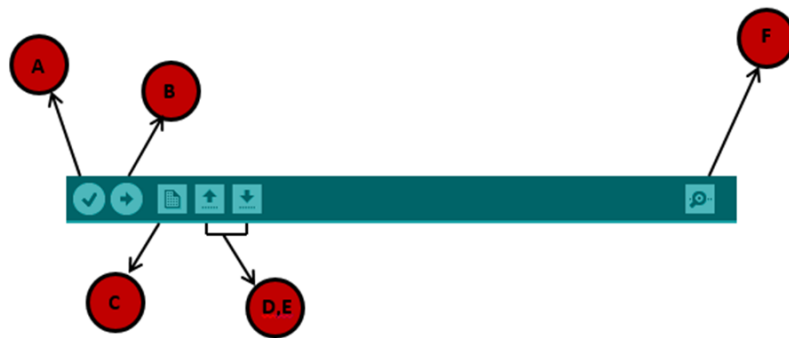
Gambar 1. 10 Memilih serial port



Langkah 8 - Unggah program ke papan Anda.

Sebelum menjelaskan bagaimana kita dapat mengunggah program kita ke papan, kita harus mendemonstrasikan fungsi setiap simbol yang muncul di toolbar Arduino IDE.

- A - Digunakan untuk memeriksa apakah ada kesalahan kompilasi.
- B - Digunakan untuk mengunggah program ke papan Arduino.
- C - Shortcut yang digunakan untuk membuat sketsa baru.
- D - Digunakan untuk langsung membuka salah satu contoh sketsa.
- E - Digunakan untuk menyimpan sketsa Anda.
- F - Monitor serial digunakan untuk menerima data serial dari papan dan mengirimkan data serial ke papan.



Gambar 1. 11 Fungsi simbol di Arduino IDE

Sekarang, cukup klik tombol "Unggah" di lingkungan. Tunggu beberapa detik; Anda akan melihat LED RX dan TX di papan berkedip. Jika upload berhasil maka akan muncul pesan "Done upload" pada status bar.

Note - Jika Anda memiliki Arduino Mini, NG, atau board lainnya, Anda perlu menekan tombol reset secara fisik di board tersebut, segera sebelum mengklik tombol upload pada Software Arduino.

E. Struktur Program

Dalam bab ini, kita akan mempelajari secara mendalam, struktur program Arduino dan mempelajari lebih banyak terminologi baru yang digunakan di dunia Arduino. Perangkat lunak Arduino adalah *open-source*. Source Kode untuk lingkungan Java dirilis di bawah GPL dan pustaka mikrokontroler C/C++ berada di bawah LGPL.

Sketch - Terminologi baru pertama adalah program Arduino yang disebut "sketsa".

Struktur

Program Arduino dapat dibagi menjadi tiga bagian utama: **Struktur**, **Nilai** (variabel dan konstanta), dan **Fungsi**. Dalam tutorial ini, kita akan belajar

tentang program perangkat lunak Arduino, langkah demi langkah, dan bagaimana kita dapat menulis program tanpa kesalahan sintaks atau kompilasi apa pun.

Mari kita mulai dengan **Strukturnya** . Struktur perangkat lunak terdiri dari dua fungsi utama :

- Setup() function
- Loop() function



Gambar 1. 12 Tampilan struktur fungsi di Arduino IDE

```
void setup( ) {  
}
```

- **PURPOSE** - Fungsi **setup()** dipanggil saat sketsa dimulai. Gunakan untuk menginisialisasi variabel, mode pin, mulai menggunakan perpustakaan, dll. Fungsi pengaturan hanya akan berjalan satu kali, setelah setiap penyalaan atau reset papan Arduino.
- **INPUT** – -
- **OUTPUT** – -
- **RETURN** – -

```
void Loop ( ) {  
}
```

- **PURPOSE** - Setelah membuat fungsi **setup()** , yang menginisialisasi dan menetapkan nilai awal, fungsi **loop()** melakukan persis seperti namanya, dan melakukan loop secara berurutan, memungkinkan program Anda berubah dan merespons. Gunakan untuk secara aktif mengontrol papan Arduino.
- **INPUT** – -

- **OUTPUT** – -
- **RETURN** – -

F. Tipe Data

Tipe data dalam C mengacu pada sistem ekstensif yang digunakan untuk mendeklarasikan variabel atau fungsi dari tipe yang berbeda. Jenis variabel menentukan berapa banyak ruang yang ditempati dalam penyimpanan dan bagaimana pola bit yang disimpan diinterpretasikan.

Tabel berikut menyediakan semua tipe data yang akan Anda gunakan selama pemrograman Arduino.

Tabel 1. 4 Tipe data pada pemrograman arduino

void	Boolean	char	Unsigned char	byte	int	Unsigned int	word
long	Unsigned long	short	float	double	array	String-char array	String-object

Void

Kata kunci void hanya digunakan dalam deklarasi fungsi. Ini menunjukkan bahwa fungsi tersebut diharapkan tidak mengembalikan informasi ke fungsi dari mana ia dipanggil.

Contoh:

```
Void Loop ( ) {
    // rest of the code
}
```

Boolean

Boolean mempunyai salah satu dari dua nilai, benar atau salah. Setiap variabel Boolean menempati satu byte memori.

Contoh:

```
boolean val = false ; // deklarasi variabel dengan tipe boolean
dan menginisialisasinya dengan false

boolean state = true ; // deklarasi variabel dengan tipe boolean
dan menginisialisasinya dengan true
```

Char

Tipe data yang menggunakan satu byte memori yang menyimpan nilai karakter. Literal karakter ditulis dalam tanda kutip tunggal seperti ini: 'A' dan untuk banyak karakter, string menggunakan tanda kutip ganda: "ABC".

Namun, karakter disimpan sebagai angka. Anda dapat melihat pengkodean spesifik di [bagan ASCII](#) . Artinya, dimungkinkan untuk melakukan operasi aritmatika pada karakter yang menggunakan nilai ASCII dari karakter tersebut. Misalnya 'A' + 1 bernilai 66, karena nilai ASCII huruf kapital A adalah 65.

Contoh:

```
Char chr_a = 'a'; //deklarasi variabel dengan tipe char dan
menginisialisasinya dengan karakter a Char

Char chr_c = 97; // deklarasi variabel dengan tipe char dan
menginisialisasinya dengan karakter 97
```

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2	SPC	!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL

Gambar 1. 13 Tabel Ascii

Unsigned char

Unsigned char adalah tipe data unsigned yang menempati satu byte memori. Tipe data unsigned char mengkodekan angka dari 0 hingga 255.

Contoh:

```
Unsigned Char chr_y = 121 ; // deklarasi variabel dengan tipe
Unsigned char dan inisialisasi dengan karakter y
```

Byte

Byte menyimpan nomor 8-bit yang tidak ditandatangani, dari 0 hingga 255.

Contoh:

```
byte m = 25; //deklarasi variabel dengan tipe byte dan menginisialisasinya dengan 25
```

Int

Bilangan bulat adalah tipe data utama untuk penyimpanan angka. `int` menyimpan nilai 16-bit (2-byte). Ini menghasilkan kisaran -32.768 hingga 32.767 (nilai minimum -2^{15} dan nilai maksimum $(2^{15}) - 1$).

Ukuran **int** bervariasi dari papan ke papan. Pada Arduino Due, misalnya, `int` menyimpan nilai 32-bit (4-byte). Ini menghasilkan kisaran -2.147.483.648 hingga 2.147.483.647 (nilai minimum -2^{31} dan nilai maksimum $(2^{31}) - 1$).

Contoh:

```
int counter = 32 ;// deklarasi variabel dengan ketik int dan inisialisasi dengan 32
```

Unsigned int

Unsigned ints (unsigned integer) sama dengan `int` dalam cara mereka menyimpan nilai 2 byte. Alih-alih menyimpan angka negatif, bagaimanapun, mereka hanya menyimpan nilai positif, menghasilkan kisaran yang berguna 0 hingga 65.535 ($2^{16} - 1$). The Due menyimpan nilai 4 byte (32-bit), mulai dari 0 hingga 4.294.967.295 ($2^{32} - 1$).

Contoh:

```
Unsigned int counter = 60 ; //deklarasi variabel dengan tipe unsigned int dan menginisialisasinya dengan 60
```

Word

Pada Uno dan papan berbasis ATMEGA lainnya, sebuah `word` menyimpan 16-bit unsigned number. Pada Due dan Zero, ia menyimpan 32-bit unsigned number.

Contoh:

```
word w = 1000 ;// deklarasi variabel dengan tipe word dan menginisialisasinya dengan 1000
```

Long

Long variabel adalah variabel ukuran yang diperluas untuk penyimpanan angka, dan menyimpan 32 bit (4 byte), dari -2.147.483.648 hingga 2.147.483.647.

Contoh:

```
long velocity = 102346; //deklarasi variabel dengan tipe Long dan menginisialisasinya dengan 102346
```

Unsigned long

Unsigned long variabel adalah variabel ukuran yang diperluas untuk penyimpanan nomor dan menyimpan 32 bit (4 byte). Tidak seperti long standar, unsigned long tidak akan menyimpan angka negatif, membuat kisarannya dari 0 hingga 4.294.967.295 ($2^{32} - 1$).

Contoh:

```
Unsigned Long velocity = 101006; //deklarasi variabel dengan tipe Unsigned Long dan inisialisasi dengan 101006
```

Short

Short adalah tipe data 16-bit. Pada semua Arduino (berbasis ATmega dan ARM), short menyimpan nilai 16-bit (2-byte). Ini menghasilkan kisaran -32.768 hingga 32.767 (nilai minimum -2^{15} dan nilai maksimum $(2^{15} - 1)$).

Contoh:

```
Short val = 13; // deklarasi variabel dengan tipe pendek dan menginisialisasinya dengan 13
```

Float

Tipe data untuk bilangan floating-point adalah bilangan yang memiliki koma desimal. Angka floating-point sering digunakan untuk memperkirakan nilai analog dan kontinu.

Contoh:

```
float num = 1.352; // deklarasi variabel dengan tipe float dan menginisialisasinya dengan 1,352
```

Double

Pada Uno dan papan berbasis ATMEGA lainnya, nomor floating-point presisi ganda menempati empat byte. Artinya, implementasi ganda persis sama dengan pelampung, tanpa peningkatan presisi. Pada Arduino Due, ganda memiliki presisi 8-byte (64 bit).

Contoh:

```
double num = 45.352; //deklarasi variabel dengan tipe ganda dan menginisialisasinya dengan 45.
```

G. Variabel dan konstanta

Sebelum kita mulai menjelaskan jenis-jenis variabel, subjek yang sangat penting yang perlu kita pastikan, Anda pahami sepenuhnya, disebut cakupan variabel .

Apa itu Cakupan Variabel?

Variabel dalam bahasa pemrograman C yang digunakan Arduino memiliki properti yang disebut scope. Cakupan adalah wilayah program dan terdapat tiga tempat di mana variabel dapat dideklarasikan yaitu adalah:

- Di dalam suatu fungsi atau blok, yang disebut **variabel lokal** .
- Dalam definisi parameter fungsi disebut **parameter formal** .
- Di luar semua fungsi, yang disebut **variabel global** .

Variabel Lokal

Variabel yang dideklarasikan di dalam suatu fungsi atau blok adalah variabel lokal. Mereka hanya dapat digunakan oleh pernyataan yang ada di dalam fungsi atau blok kode tersebut. Variabel lokal tidak diketahui berfungsi di luar variabelnya sendiri. Berikut adalah contoh penggunaan variabel lokal

```
void setup () {  
  
}  
void loop () {  
    int x , y ;  
    int z ; //deklarasi variabel lokal  
    x = 0;  
    y = 0; //inisialisasi aktual  
    z = 10;  
}
```

Variabel Global

Variabel global didefinisikan di luar semua fungsi, biasanya di bagian atas program. Variabel global akan mempertahankan nilainya sepanjang masa program Anda. Variabel global dapat diakses oleh fungsi apa pun. Artinya, variabel global tersedia untuk digunakan di seluruh program Anda setelah deklarasinya.

Contoh berikut menggunakan variabel global dan lokal

```
Int T , S ;  
float c = 0 ; //deklarasi global variabel  
Void setup () {
```

```

}
Void loop () {
    int x , y ;
    int z ; //deklarasi lokal variabel
    x = 0;
    y = 0; // inisialisasi aktual
    z = 10;}

```

H. Operator

Operator adalah simbol yang memberitahu kompiler untuk melakukan fungsi matematika atau logika tertentu. Bahasa C kaya akan operator bawaan dan menyediakan jenis operator berikut :

- Operator Aritmatika

Asumsikan variabel A menampung 10 dan variabel B menampung 20, lalu

– [Tunjukkan contoh](#)

Tabel 1. 5 Operator aritmatika

Nama operator	Operator sederhana	Deskripsi	Contoh
Operator penugasan	=	Menyimpan nilai di sebelah kanan tanda sama dengan dalam variabel di sebelah kiri tanda sama dengan.	A = B
penambahan	+	Menambahkan dua operan	A + B akan menghasilkan 30
Pengurangan	-	Mengurangi operan kedua dari yang pertama	A - B akan memberikan -10
Perkalian	*	Kalikan kedua operan	A * B akan memberikan 200
divisi	/	Membagi pembilang dengan penyebut	B / A akan memberikan 2

- Operator Perbandingan

Asumsikan variabel A menampung 10 dan variabel B menampung 20, lalu

– [Tunjukkan contoh](#)

Tabel 1. 6 Operator perbandingan

Nama operator	Operator sederhana	Deskripsi	Contoh
sama dengan	==	Memeriksa apakah nilai dua operan sama atau tidak, jika ya maka kondisi menjadi benar.	(A == B) adalah tidak benar

Nama operator	Operator sederhana	Deskripsi	Contoh
tidak sama dengan	<code>!=</code>	Memeriksa apakah nilai dua operan sama atau tidak, jika nilai tidak sama maka kondisi menjadi benar.	$(A \neq B)$ adalah benar
Kurang dari	<code><</code>	Memeriksa apakah nilai operan kiri kurang dari nilai operan kanan, jika ya maka kondisi menjadi benar.	$(A < B)$ adalah benar
Lebih besar dari	<code>></code>	Memeriksa apakah nilai operan kiri lebih besar dari nilai operan kanan, jika ya maka kondisi menjadi benar.	$(A > B)$ adalah tidak benar
kurang dari sama dengan	<code><=</code>	Memeriksa apakah nilai operan kiri kurang dari atau sama dengan nilai operan kanan, jika ya maka kondisi menjadi benar.	$(A \leq B)$ adalah benar
lebih besar dari sama dengan	<code>>=</code>	Memeriksa apakah nilai operan kiri lebih besar dari atau sama dengan nilai operan kanan, jika ya maka kondisi menjadi benar.	$(A \geq B)$ adalah tidak benar

- Operator Boolean

Asumsikan variabel A menampung 10 dan variabel B menampung 20, lalu

– [Tunjukkan contoh](#)

Tabel 1. 7 Operator boolean

Nama operator	Operator sederhana	Deskripsi	Contoh
dan	<code>&&</code>	Disebut operator Logical AND. Jika kedua operan bukan nol maka kondisi menjadi benar.	$(A \&\& B)$ benar
atau	<code> </code>	Disebut operator OR logis. Jika salah satu dari dua operan bukan nol maka kondisi menjadi benar.	$(A B)$ benar
		Disebut operator NOT logis. Gunakan untuk membalikkan keadaan logis operan. Jika	$!(A \&\& B)$

tidak	!	suatu kondisi benar maka operator Logical salah NOT akan membuat false.	
-------	---	---	--

- Operator Bitwise

Asumsikan variabel A menampung 60 dan variabel B menampung 13, lalu

– [Tunjukkan contoh](#)

Tabel 1. 8 Operator bitwise

Nama operator	Operator sederhana	Deskripsi	Contoh
and	&	Biner DAN Operator menyalin sedikit ke hasil jika ada di kedua operan.	(A & B) akan memberikan 12 yaitu 0000 1100
or		Operator OR Biner menyalin sedikit jika ada di salah satu operan	(A B) akan memberikan 61 yaitu 0011 1101
XOR	^	Operator XOR Biner menyalin bit jika diatur dalam satu operan tetapi tidak keduanya.	(A ^ B) akan menghasilkan 49 yaitu 0011 0001
not	~	Binary Ones Complement Operator tidak ary dan memiliki efek 'membalik' bit.	(~A) akan memberikan -60 yaitu 1100 0011
Shift left	<<	Operator Pergeseran Kiri Biner. Nilai operan kiri dipindahkan ke kiri dengan jumlah bit yang ditentukan oleh operan kanan.	A << 2 akan menghasilkan 240 yaitu 1111 0000
Shift right	>>	Operator Pergeseran Kanan Biner. Nilai operan kiri dipindahkan ke kanan dengan jumlah bit yang ditentukan oleh operan kanan.	A >> 2 akan menghasilkan 15 yaitu 0000 1111

- Operator Gabungan

Asumsikan variabel A menampung 10 dan variabel B menampung 20, lalu

– [Tunjukkan contoh](#)

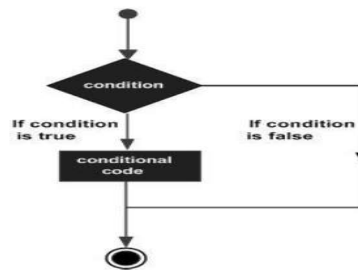
Tabel 1. 9 Operator gabungan

Nama operator	Operator sederhana	Deskripsi	Contoh
increment	++	Operator kenaikan, menambah nilai integer sebanyak satu	A++ akan memberikan 11
decrement	--	Operator pengurangan, mengurangi nilai integer sebanyak satu	A-- akan memberi 9
compound addition	+=	Tambahkan operator penugasan AND. Ia menambahkan operan kanan ke operan kiri dan menetapkan hasilnya ke operan kiri	B += A setara dengan B = B + A
compound subtraction	-=	Operator pengurangan DAN penugasan. Ini mengurangi operan kanan dari operan kiri dan menetapkan hasilnya ke operan kiri	B -= A setara dengan B = B - A
compound multiplication	*=	Perkalian DAN operator penugasan. Ini mengalikan operan kanan dengan operan kiri dan menetapkan hasilnya ke operan kiri	B *= A setara dengan B = B * A
compound division	/=	Bagilah DAN operator penugasan. Ini membagi operan kiri dengan operan kanan dan menetapkan hasilnya ke operan kiri	B /= A setara dengan B = B / A
compound modulo	%=	Modulus DAN operator penugasan. Dibutuhkan modulus menggunakan dua operan dan menetapkan hasilnya ke operan kiri	B %= A setara dengan B = B % A
compound bitwise or	=	bitwise inklusif OR dan operator penugasan	A = 2 sama dengan A = A 2
compound bitwise and	&=	Bitwise DAN operator penugasan	A &= 2 sama dengan A = A & 2

I. Kontrol statement

Struktur pengambilan keputusan mengharuskan pemrogram menentukan satu atau lebih kondisi untuk dievaluasi atau diuji oleh program. Itu harus bersama dengan pernyataan atau pernyataan yang akan dieksekusi jika kondisi ditentukan benar, dan secara opsional, pernyataan lain akan dieksekusi jika kondisi ditentukan salah.

Berikut ini adalah bentuk umum dari struktur pengambilan keputusan yang umum ditemukan di sebagian besar bahasa pemrograman



Gambar 1. 14 Struktur pengambilan keputusan if...else

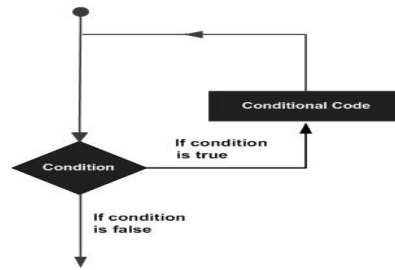
Pernyataan Kontrol adalah elemen dalam Kode Sumber yang mengontrol aliran eksekusi program. Yaitu adalah:

Tabel 1. 10 Pernyataan dan struktur kontrol if...else

S.NO.	Pernyataan & Deskripsi Kontrol
1	Pernyataan if Dibutuhkan ekspresi dalam tanda kurung dan pernyataan atau blok pernyataan. Jika ekspresi itu benar maka pernyataan atau blok pernyataan akan dieksekusi jika tidak pernyataan ini dilewati.
2	Pernyataan if....else Pernyataan if dapat diikuti oleh pernyataan else opsional, yang dijalankan ketika ekspresi false .
3	Pernyataan if....else if....else Pernyataan if dapat diikuti oleh pernyataan opsional jika... pernyataan lain , yang sangat berguna untuk menguji berbagai kondisi menggunakan single if... lain jika pernyataan.
4	Pernyataan Switch Case Mirip dengan pernyataan if , switch ... Case mengontrol aliran program dengan memungkinkan programmer untuk menentukan kode yang berbeda yang harus dieksekusi dalam berbagai kondisi.
5	Operator Bersyarat ? : Operator kondisional ? : adalah satu-satunya operator ternary di C.

J. Loop

Bahasa pemrograman menyediakan berbagai struktur kontrol yang memungkinkan jalur eksekusi lebih rumit. Pernyataan loop memungkinkan kita untuk mengeksekusi sebuah pernyataan atau sekelompok pernyataan beberapa kali dan berikut adalah bentuk umum dari pernyataan loop di sebagian besar bahasa pemrograman



Gambar 1. 15 Struktur pengambilan keputusan pernyataan loop

Bahasa pemrograman C menyediakan jenis loop berikut untuk menangani kebutuhan perulangan.

Tabel 1. 11 Beberapa fungsi dan deskripsi loop

S.NO.	Fungsi & Deskripsi
1	<u>while loop</u> Perulangan while akan berulang terus menerus, dan tanpa batas, hingga ekspresi di dalam tanda kurung, () menjadi salah. Sesuatu harus mengubah variabel yang diuji, atau perulangan while tidak akan pernah keluar.
2	<u>do...while loop</u> Perulangan do... while mirip dengan perulangan while. Pada perulangan while, kondisi kelanjutan perulangan diuji pada awal perulangan sebelum melakukan badan perulangan.
3	<u>for loop</u> Perulangan for mengeksekusi pernyataan beberapa kali yang telah ditentukan. Ekspresi kontrol untuk perulangan diinisialisasi, diuji, dan
4	<u>nested loop</u> Bahasa C memungkinkan Anda menggunakan satu loop di dalam loop lainnya. Contoh berikut mengilustrasikan konsep tersebut.
5	<u>infiniti loop</u> Ini adalah perulangan yang tidak memiliki kondisi terminasi, sehingga perulangan menjadi tak terbatas.

K. Fungsi

Fungsi memungkinkan penataan program dalam segmen kode untuk melakukan tugas individual. Kasus umum untuk membuat suatu fungsi adalah ketika seseorang perlu melakukan tindakan yang sama beberapa kali dalam suatu program.

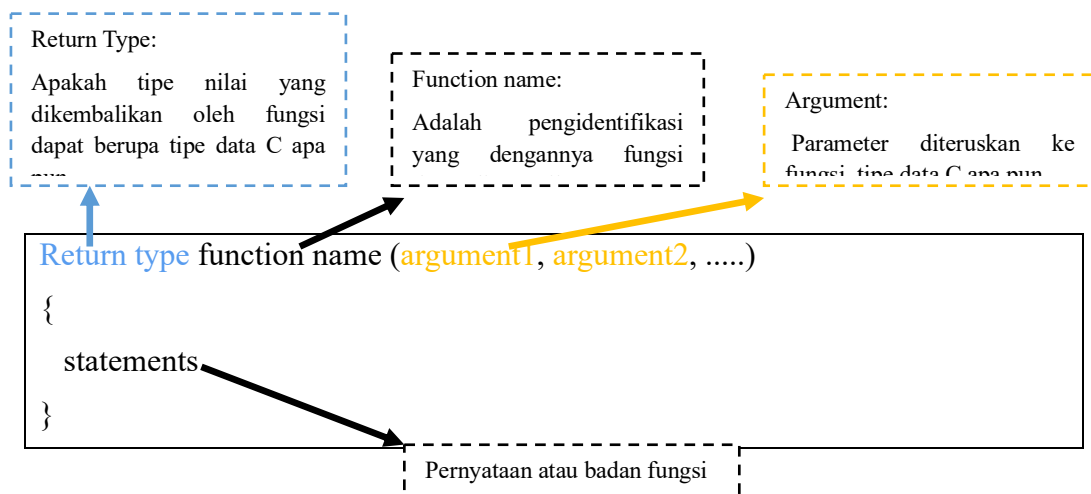
Menstandarisasi fragmen kode menjadi fungsi memiliki beberapa keuntungan :

- Fungsi membantu programmer tetap terorganisir. Seringkali hal ini membantu untuk membuat konsep program.
- Fungsi menyusun satu tindakan di satu tempat sehingga fungsi tersebut hanya perlu dipikirkan dan di-debug satu kali.

- Hal ini juga mengurangi kemungkinan kesalahan dalam modifikasi, jika kode perlu diubah.
- Fungsi membuat keseluruhan sketsa menjadi lebih kecil dan ringkas karena bagian kode digunakan berulang kali.
- Mereka mempermudah penggunaan kembali kode di program lain dengan menjadikannya modular, dan penggunaan fungsi sering kali membuat kode lebih mudah dibaca.

Ada dua fungsi yang dibutuhkan dalam sketsa atau program Arduino yaitu `setup()` dan `loop()`. Fungsi lain harus dibuat di luar tanda kurung kedua fungsi tersebut.

Sintaks paling umum untuk mendefinisikan suatu fungsi adalah:



Gambar 1. 16 Definisi suatu fungsi

Deklarasi Fungsi

Suatu fungsi dideklarasikan di luar fungsi lainnya, di atas atau di bawah fungsi `loop`. Kita dapat mendeklarasikan fungsi tersebut dengan dua cara berbeda. Cara pertama cukup menuliskan bagian fungsi yang disebut **prototipe fungsi** di atas fungsi `loop`, yang terdiri dari

- Tipe pengembalian fungsi
- Nama fungsi
- Tipe argumen fungsi, tidak perlu menulis nama argumen

Prototipe fungsi harus diikuti dengan titik koma (;).

Contoh berikut menunjukkan demonstrasi deklarasi fungsi menggunakan metode pertama.

```
int sum_func (int x, int y) // Deklarasi fungsi
{
    int z = 0;
    z = x+y ;
}
```

```

    return z; // Mengembalikan nilai
}
void setup () {
    Statements // Kelompok pernyataan
}
Void loop () {
    int result = 0 ;
    result = Sum_func (5,6) ; // panggilan fungsi
}

```

Bagian kedua, yang disebut definisi atau deklarasi fungsi, harus dideklarasikan di bawah fungsi loop, yang terdiri dari:

- Tipe pengembalian fungsi
- Nama fungsi
- Tipe argumen fungsi, di sini Anda harus menambahkan nama argumen
- Badan fungsi (pernyataan di dalam fungsi yang dijalankan ketika fungsi tersebut dipanggil)

Contoh berikut menunjukkan deklarasi fungsi menggunakan metode kedua.

```

int sum_func (int , int ) ; // Prototipe fungsi
void setup () {
    Statements // Kelompok pernyataan
}
Void loop () {
    int result = 0 ;
    result = Sum_func (5,6) ; // panggilan fungsi
}
int sum_func (int x, int y) // Deklarasi fungsi
{
    int z = 0;
    z = x+y ;
    return z; // Mengembalikan nilai
}

```

Metode kedua hanya mendeklarasikan fungsi di atas fungsi loop.

L. String

String digunakan untuk menyimpan teks. Mereka dapat digunakan untuk menampilkan teks pada LCD atau di jendela Arduino IDE Serial Monitor. String juga berguna untuk menyimpan input pengguna. Misalnya karakter yang diketik pengguna pada keypad yang terhubung ke Arduino.

Ada dua jenis string dalam pemrograman Arduino:

- Array karakter, yang sama dengan string yang digunakan dalam pemrograman C.
- Arduino String, yang memungkinkan kita menggunakan objek string dalam sketsa.

Pada bab ini, kita akan mempelajari String, objek, dan penggunaan string dalam sketsa Arduino. Di akhir bab ini, Anda akan mempelajari jenis string mana yang akan digunakan dalam sketsa.

- **String Character Arrays**

Tipe string pertama yang akan kita pelajari adalah string yang merupakan rangkaian karakter bertipe **char**. Pada bab sebelumnya, kita telah mempelajari apa itu array; serangkaian berturut-turut dari jenis variabel yang sama yang disimpan dalam memori. String adalah array variabel char.

String adalah array khusus yang memiliki satu elemen tambahan di akhir string, yang selalu bernilai 0 (nol). Ini dikenal sebagai "string yang diakhiri dengan nol".

Contoh string character arrays

Contoh ini akan menunjukkan cara membuat string dan mencetaknya ke jendela monitor serial.

```
void setup() {
    char my_str[6]; //array yang cukup besar untuk string 5 karakter
    Serial.begin(9600);
    my_str[0] = 'H'; //string terdiri dari 5 karakter
    my_str[1] = 'e';
    my_str[2] = 'l';
    my_str[3] = 'l';
    my_str[4] = 'o';
    my_str[5] = 0; //Elemen array ke-6 adalah terminator null
    Serial.println(my_str);
}

void loop() {
}
```

Contoh berikut menunjukkan terdiri dari apa sebuah string; array karakter dengan karakter yang dapat dicetak dan 0 sebagai elemen terakhir array untuk menunjukkan bahwa di sinilah string berakhir. String dapat dicetak ke jendela Arduino IDE Serial Monitor dengan menggunakan **Serial.println()** dan meneruskan nama string.

Contoh yang sama dapat ditulis dengan cara yang lebih mudah seperti yang ditunjukkan di bawah ini:

```
void setup() {
    char my_str[] = "Hello";
```

```

    Serial.begin(9600);
    Serial.println(my_str);
}
void loop() {
}

```

Dalam sketsa ini, kompiler menghitung ukuran array string dan juga secara otomatis null mengakhiri string dengan nol. Sebuah array yang panjangnya enam elemen dan terdiri dari lima karakter diikuti dengan angka nol dibuat dengan cara yang persis sama seperti pada sketsa sebelumnya.

- **Manipulating String Arrays**

Kita dapat mengubah array string dalam sketsa seperti yang ditunjukkan pada sketsa berikut.

```

void setup() {
    char like[] = "I like coffee and cake"; // Membuat string
    Serial.begin(9600);
    // (1) mencetak string
    Serial.println(like);
    // (2) Menghapus bagian dari string
    like[13] = 0;
    Serial.println(like);
    // (3) Mengganti kata ke dalam string
    like[13] = ' '; // Mengganti terminator null dengan spasi
    like[18] = 't'; // Menyisipkan kata baru
    like[19] = 'e';
    like[20] = 'a';
    like[21] = 0; // Mengakhiri string
    Serial.println(like);
}
void loop() {
}

```

Hasil:

```

I like coffee and cake
I like coffee
I like coffee and tea

```

Sketsanya bekerja dengan cara berikut.

- a. Membuat dan Mencetak String

Dalam sketsa yang diberikan di atas, string baru dibuat dan kemudian dicetak untuk ditampilkan di jendela Serial Monitor.

b. Memperpendek String

String dipersingkat dengan mengganti karakter ke-14 dalam string dengan nol yang mengakhiri nol (2). Ini adalah elemen nomor 13 dalam array string yang dihitung dari 0.

Ketika string dicetak, semua karakter dicetak hingga nol baru yang mengakhiri nol. Karakter lainnya tidak hilang; mereka masih ada di memori dan array stringnya masih berukuran sama. Satu-satunya perbedaan adalah fungsi apa pun yang bekerja dengan string hanya akan melihat string hingga terminator nol pertama.

c. Mengubah Kata dalam String

Terakhir, sketsa tersebut menggantikan kata "kue" dengan "teh" (3). Pertama-tama ia harus mengganti terminator nol di `like[13]` dengan spasi sehingga string dikembalikan ke format yang dibuat semula.

Karakter baru menempa "cak" dari kata "kue" dengan kata "teh". Hal ini dilakukan dengan menempa karakter individual. 'E' dari "kue" diganti dengan karakter pengakhiran nol yang baru. Hasilnya adalah string tersebut sebenarnya diakhiri dengan dua karakter null, karakter asli di akhir string dan karakter baru yang menggantikan 'e' di "kue". Hal ini tidak ada bedanya ketika string baru dicetak karena fungsi yang mencetak string berhenti mencetak karakter string ketika menemukan terminator nol pertama.

- **Functions to Manipulate String Arrays**

Sketsa sebelumnya memanipulasi string secara manual dengan mengakses karakter individual dalam string. Untuk mempermudah manipulasi array string, Anda dapat menulis fungsi Anda sendiri untuk melakukannya, atau menggunakan beberapa fungsi string dari pustaka bahasa C.

Diberikan di bawah ini adalah daftar untuk Functions to Manipulate String Arrays

Tabel 1. 12 Daftar function to manipulation string arrays

S.No.	Fungsi & Deskripsi
1	String() Kelas String, bagian dari inti pada versi 0019, memungkinkan Anda untuk menggunakan dan memanipulasi string teks dengan cara yang lebih kompleks daripada array karakter. Anda dapat menggabungkan String, menambahkannya, mencari dan mengganti substring, dan banyak lagi. Dibutuhkan lebih banyak memori daripada array karakter sederhana, tetapi juga lebih berguna. Sebagai referensi, array karakter disebut sebagai string dengan 's' kecil, dan instance kelas String disebut sebagai String dengan huruf kapital S. Perhatikan bahwa string konstanta, yang ditentukan dalam "tanda kutip ganda" diperlakukan sebagai array char, bukan instance dari kelas String
2	charAt() Akses karakter tertentu dari String.
3	compareTo() Membandingkan dua string, menguji apakah satu datang sebelum atau setelah yang lain, atau apakah mereka aqual. String dibandingkan karakter demi karakter, menggunakan nilai ASCII karakter. Itu berarti, misalnya, 'a' datang sebelum 'b' tetapi setelah 'A' nomor datang sebelum yang terakhir.
4	concat() Menambahkan parameter ke String.
5	c_str() Mengonversi konten string sebagai string gaya C, null-terminated. Perhatikan bahwa ini memberikan akses langsung ke buffer String internal dan harus digunakan dengan hati-hati. Secara khusus, Anda tidak boleh memodifikasi string melalui pointer yang dikembalikan. Ketika Anda mengubah objek string, atau ketika dihancurkan, pointer yang sebelumnya dikembalikan oleh c_str() menjadi tidak valid dan tidak boleh digunakan lagi.
6	endsWith() Menguji apakah String berakhir dengan karakter String lain atau tidak.
7	equals() Membandingkan dua string untuk kesetaraan. Perbandingannya peka huruf besar/kecil, artinya String "hello" tidak sama dengan String "HELLO".
8	equalsIgnoreCase() Membandingkan dua string untuk kesetaraan. Perbandingannya tidak peka huruf besar/kecil, artinya String ("halo") sama dengan String ("HELLO").
9	getBytes() Menyalin karakter string ke buffer yang disediakan.
10	indexOf() Menempatkan karakter atau String dalam String lain. Secara default, ia mencari dari awal String, tetapi juga dapat memulai dari indeks tertentu, memungkinkan untuk menemukan semua contoh karakter atau String.

S.No.	Fungsi & Deskripsi
11	lastIndexOf() Menempatkan karakter atau String dalam String lain. Secara default, ia mencari dari akhir String, tetapi juga dapat bekerja mundur dari indeks tertentu, memungkinkan untuk menemukan semua contoh karakter atau String.
12	length() Mengembalikan panjang String, dalam karakter. (Perhatikan bahwa ini tidak termasuk karakter null di belakang.)
13	remove() Ubah di tempat, string menghapus karakter dari indeks yang disediakan ke akhir string atau dari indeks yang disediakan ke indeks plus hitungan.
14	replace() Fungsi String replace() memungkinkan Anda mengganti semua contoh karakter tertentu dengan karakter lain. Anda juga dapat menggunakan replace untuk mengganti substring string dengan substring yang berbeda.
15	reserve() Fungsi String reserve() memungkinkan Anda mengalokasikan buffer dalam memori untuk memanipulasi string.
16	setCharAt() Mengatur karakter String. Tidak berpengaruh pada indeks di luar panjang String yang ada.
17	startsWith() Menguji apakah String dimulai dengan karakter String lain atau tidak.
18	toCharArray() Menyalin karakter string ke buffer yang disediakan.
19	substring() Mendapatkan substring dari String. Indeks awal bersifat inklusif (karakter yang sesuai disertakan dalam substring), tetapi indeks akhir opsional bersifat eksklusif (karakter yang sesuai tidak termasuk dalam substring). Jika indeks akhir dihilangkan, substring berlanjut ke akhir String.
20	toInt() Mengonversi String yang valid menjadi bilangan bulat. String input harus dimulai dengan bilangan bulat. Jika string berisi angka non-bilangan bulat, fungsi akan berhenti melakukan konversi.
21	toFloat() Mengonversi String yang valid menjadi float. String input harus dimulai dengan digit. Jika string berisi karakter non-digit, fungsi akan berhenti melakukan konversi. Misalnya, string "123,45", "123", dan "123fish" masing-masing dikonversi menjadi 123,45, 123,00, dan 123,00. Perhatikan bahwa "123.456" diperkirakan dengan 123.46. Perhatikan juga bahwa pelampung hanya memiliki presisi 6-7 digit desimal dan string yang lebih panjang mungkin terpotong.
22	toLowerCase()

S.No.	Fungsi & Deskripsi
	Dapatkan String versi huruf kecil. Pada 1.0, toLowerCase() memodifikasi string di tempat daripada mengembalikan yang baru.
23	toUpperCase() Dapatkan String versi huruf besar. Pada 1.0, toUpperCase() memodifikasi string di tempat daripada mengembalikan yang baru.
24	trim() Dapatkan versi String dengan spasi putih di depan dan di belakang dihapus. Pada 1.0, trim() memodifikasi string di tempat daripada mengembalikan yang baru.

Sketsa selanjutnya menggunakan beberapa fungsi string C.

```
void setup() {
    char str[] = "This is my string"; // Membuat string
    char out_str[40]; // output dari fungsi string ditempatkan di sini
    int num; // Bilangan bulat tujuan umum
    Serial.begin(9600);
    // (1) mencetak string
    Serial.println(str);

    // (2) Mendapatkan panjang string (tidak termasuk null terminator)
    num = strlen(str);
    Serial.print("String length is: ");
    Serial.println(num);

    // (3) Mendapatkan panjang array (termasuk null terminator)
    num = sizeof(str); // sizeof() bukan fungsi string C
    Serial.print("Size of the array: ");
    Serial.println(num);

    // (4) Menyalin string
    strcpy(out_str, str);
    Serial.println(out_str);

    // (5) Menambahkan string ke akhir string (menambahkan)
    strcat(out_str, " sketch.");
    Serial.println(out_str);
    num = strlen(out_str);
    Serial.print("String length is: ");
    Serial.println(num);
    num = sizeof(out_str);
    Serial.print("Size of the array out_str[]: ");
    Serial.println(num);
}
```

```
}  
void loop() {  
}
```

Hasil:

```
This is my string  
String length is: 17  
Size of the array: 18  
This is my string  
This is my string sketch.  
String length is: 25  
Size of the array out_str[]: 40
```

Sketsanya bekerja dengan cara berikut.

a. Cetak Stringnya

String yang baru dibuat dicetak ke jendela Serial Monitor seperti yang dilakukan pada sketsa sebelumnya.

b. Dapatkan Panjang Talinya

Fungsi `strlen()` digunakan untuk mengetahui panjang string. Panjang string hanya untuk karakter yang dapat dicetak dan tidak termasuk terminator nol. String berisi 17 karakter, jadi kita melihat 17 karakter tercetak di jendela Serial Monitor.

c. Dapatkan Panjang Array

Operator `sizeof()` digunakan untuk mendapatkan panjang array yang berisi string. Panjangnya termasuk terminator nol, jadi panjangnya satu lebih panjang dari panjang string. `sizeof()` tampak seperti fungsi, tetapi secara teknis merupakan operator. Ini bukan bagian dari perpustakaan string C, tetapi digunakan dalam sketsa untuk menunjukkan perbedaan antara ukuran array dan ukuran string (atau panjang string).

d. Salin sebuah String

Fungsi `strcpy()` digunakan untuk menyalin string `str[]` ke array `out_num[]`. Fungsi `strcpy()` menyalin string kedua yang diteruskan ke string pertama. Salinan string sekarang ada di array `out_num[]`, tetapi hanya menggunakan 18 elemen array, jadi kita masih memiliki 22 elemen char gratis di array. Elemen bebas ini ditemukan setelah string di memori. String tersebut disalin ke array sehingga kita memiliki ruang ekstra di array untuk digunakan di bagian sketsa selanjutnya, yaitu menambahkan string ke akhir string.

e. Menambahkan String ke String (Gabungan)

Sketsa tersebut menggabungkan satu string ke string lainnya, yang dikenal sebagai penggabungan. Ini dilakukan dengan menggunakan fungsi `strcat()`. Fungsi `strcat()` menempatkan string kedua yang diteruskan ke akhir string pertama yang diteruskan ke sana. Setelah penggabungan, panjang string dicetak untuk menunjukkan panjang string baru. Panjang array kemudian dicetak untuk menunjukkan bahwa kita memiliki string sepanjang 25 karakter dalam array sepanjang 40 elemen. Ingatlah bahwa string sepanjang 25 karakter sebenarnya memakan 26 karakter array karena nol yang mengakhiri nol.

- **Batas Array**

Saat bekerja dengan string dan array, sangat penting untuk bekerja dalam batasan string atau array. Dalam contoh sketsa, sebuah array dibuat sepanjang 40 karakter untuk mengalokasikan memori yang dapat digunakan untuk memanipulasi string. Jika array dibuat terlalu kecil dan kami mencoba menyalin string yang lebih besar dari array ke dalamnya, string tersebut akan disalin ke akhir array. Memori di luar akhir array dapat berisi data penting lainnya yang digunakan dalam sketsa, yang kemudian akan ditimpa oleh string kita. Jika memori di luar akhir string dibanjiri, sketsa dapat crash atau menyebabkan perilaku yang tidak diharapkan.

M. Objek string

Jenis string kedua yang digunakan dalam pemrograman Arduino adalah String Object.

Apa itu Objek?

Objek adalah konstruksi yang berisi data dan fungsi. Objek String dapat dibuat seperti variabel dan diberi nilai atau string. Objek String berisi fungsi (yang disebut "metode" dalam pemrograman berorientasi objek (OOP)) yang beroperasi pada data string yang terdapat dalam objek String.

Sketsa dan penjelasan berikut akan memperjelas apa itu objek dan bagaimana objek String digunakan.

```
void setup() {
    String my_str = "This is my string.";
    Serial.begin(9600);
    // (1) mencetak string
    Serial.println(my_str);
    // (2) Ubah string menjadi huruf besar
    my_str.toUpperCase();
    Serial.println(my_str);
    // (3) Menimpa string
    my_str = "My new string.";
    Serial.println(my_str);
    // (4) Mengganti kata dalam string
```

```

    my_str.replace("string", "Arduino sketch");
    Serial.println(my_str);
    // (5) mendapatkan panjang string
    Serial.print("String length is: ");
    Serial.println(my_str.length());
}
void loop() {
}

```

Hasil:

```

This is my string.
THIS IS MY STRING.
My new string.
My new Arduino sketch.
String length is: 22

```

Objek string dibuat dan diberi nilai (atau string) di bagian atas sketsa.

```
String my_str = "This is my string." ;
```

Ini menciptakan objek String dengan nama **my_str** dan memberinya nilai "Ini adalah string saya.".

Hal ini dapat dibandingkan dengan membuat variabel dan memberikan nilai padanya seperti bilangan bulat

```
int my_var = 102;
```

Sketsanya bekerja dengan cara berikut.

- a. Mencetak Tali

String dapat dicetak ke jendela Serial Monitor seperti string array karakter.

- b. Ubah String menjadi Huruf Besar

Objek string **my_str** yang telah dibuat, memiliki sejumlah fungsi atau metode yang dapat dioperasikan di dalamnya. Metode ini dipanggil dengan menggunakan nama objek diikuti dengan operator titik (.) dan kemudian nama fungsi yang akan digunakan.

```
my_str.toUpperCase();
```

Fungsi `toUpperCase()` beroperasi pada string yang terdapat dalam objek `my_str` yang bertipe `String` dan mengubah data string (atau teks) yang terdapat dalam objek menjadi karakter huruf besar. Daftar fungsi yang terdapat dalam kelas `String` dapat ditemukan di referensi `String Arduino`. Secara teknis, `String` disebut kelas dan digunakan untuk membuat objek `String`.

c. Timpa sebuah `String`

Operator penugasan digunakan untuk menetapkan string baru ke objek `my_str` yang menggantikan string lama

```
my_str = "My new string." ;
```

Operator penugasan tidak dapat digunakan pada string array karakter, tetapi hanya berfungsi pada objek `String`.

d. Mengganti Kata dalam `String`

Fungsi `replace()` digunakan untuk mengganti string pertama yang diteruskan dengan string kedua yang diteruskan ke sana. `replace()` adalah fungsi lain yang dibangun ke dalam kelas `String` sehingga tersedia untuk digunakan pada objek `String my_str`.

e. Mendapatkan Panjang `String`

Mendapatkan panjang string mudah dilakukan dengan menggunakan `length()`. Dalam contoh sketsa, hasil yang dikembalikan oleh `length()` diteruskan langsung ke `Serial.println()` tanpa menggunakan variabel perantara.

- **Kapan Menggunakan Objek `String`**

Objek `String` jauh lebih mudah digunakan daripada array karakter string. Objek tersebut memiliki fungsi bawaan yang dapat melakukan sejumlah operasi pada string. Kerugian utama menggunakan objek `String` adalah ia menggunakan banyak memori dan dapat dengan cepat menghabiskan memori RAM Arduino, yang dapat menyebabkan Arduino hang, crash, atau berperilaku tidak terduga. Jika sketsa di Arduino berukuran kecil dan membatasi penggunaan objek, maka seharusnya tidak ada masalah.

`String` array karakter lebih sulit digunakan dan Anda mungkin perlu menulis fungsi Anda sendiri untuk mengoperasikan jenis string ini. Keuntungannya adalah Anda memiliki kendali atas ukuran array string yang Anda buat, sehingga Anda dapat menjaga array tetap kecil untuk menghemat memori. Anda perlu memastikan bahwa Anda tidak menulis melebihi batas akhir array dengan array string. Objek `String` tidak memiliki masalah ini dan akan menangani batasan string untuk Anda, asalkan ada cukup memori untuk mengoperasikannya. Objek `String` dapat mencoba menulis ke memori yang tidak ada ketika kehabisan memori, tetapi tidak akan pernah menulis di akhir string yang dioperasikannya.

- **Dimana String Digunakan**

Dalam bab ini kita mempelajari tentang string, bagaimana perilakunya dalam memori, dan pengoperasiannya. Penggunaan praktis dari string akan dibahas pada bagian selanjutnya dari kursus ini ketika kita mempelajari cara mendapatkan input pengguna dari jendela Serial Monitor dan menyimpan input dalam sebuah string.

N. Waktu

Arduino menyediakan empat fungsi manipulasi waktu yang berbeda. Yaitu adalah:

Tabel 1. 13 Fungsi dan deskripsi waktu di Arduini IDE

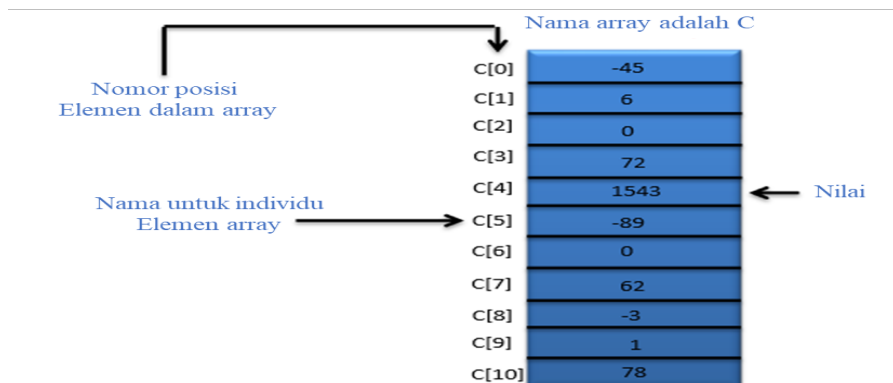
S.No.	Fungsi & Deskripsi
1	<u>delay () function</u> Cara kerja fungsi delay() cukup sederhana. Ini menerima argumen bilangan bulat tunggal (atau angka). Angka ini mewakili waktu (diukur dalam milidetik).
2	<u>delayMicroseconds () function</u> Fungsi delayMicroseconds() menerima argumen bilangan bulat (atau angka) tunggal. Ada seribu mikrodetik dalam milidetik, dan satu juta mikrodetik dalam satu detik.
3	<u>millis () function</u> Fungsi ini digunakan untuk mengembalikan jumlah milidetik pada saat itu, papan Arduino mulai menjalankan program saat ini
4	<u>micros () function</u> Fungsi micros() mengembalikan jumlah mikrodetik dari waktu, papan Arduino mulai menjalankan program saat ini. Jumlah ini meluap yaitu kembali ke nol setelah sekitar 70 menit.

O. Arrays

Array adalah sekelompok lokasi memori berurutan yang bertipe sama. Untuk merujuk ke lokasi atau elemen tertentu dalam array, kita menentukan nama array dan nomor posisi elemen tertentu dalam array.

Ilustrasi yang diberikan di bawah ini menunjukkan array integer bernama C yang berisi 11 elemen. Anda merujuk ke salah satu elemen ini dengan memberikan nama array diikuti dengan nomor posisi elemen tertentu dalam tanda kurung siku ([]). Nomor posisi lebih formal disebut subskrip atau indeks (angka ini menentukan jumlah elemen dari awal array). Elemen pertama memiliki subskrip 0 (nol) dan kadang-kadang disebut elemen nol.

Jadi, elemen array C adalah C[0] (diucapkan “C sub zero”), C[1], C[2] dan seterusnya. Subskrip tertinggi pada larik C adalah 10, yaitu 1 lebih sedikit dari jumlah elemen pada larik (11). Nama array mengikuti konvensi yang sama dengan nama variabel lainnya.



Gambar 1. 17 Pemosisian array

Subskrip harus berupa bilangan bulat atau ekspresi bilangan bulat (menggunakan tipe integral apa pun). Jika suatu program menggunakan ekspresi sebagai subskrip, maka program tersebut mengevaluasi ekspresi tersebut untuk menentukan subskripnya. Misalnya, jika kita berasumsi bahwa variabel a sama dengan 5 dan variabel b sama dengan 6, maka pernyataan tersebut menambahkan 2 ke elemen array C[11].

Nama array yang disubskrip adalah nilai, dapat digunakan di sisi kiri tugas, sama seperti nama variabel non-array.

Mari kita periksa array C pada gambar yang diberikan lebih dekat. Nama seluruh array adalah C. 11 elemennya disebut sebagai C[0] hingga C[10]. Nilai C[0] adalah -45, nilai C[1] adalah 6, nilai C[2] adalah 0, nilai C[7] adalah 62, dan nilai C[10] adalah 78.

Untuk mencetak jumlah nilai yang terkandung dalam tiga elemen pertama array C, kita akan menulis

```
Serial.print (C[ 0 ] + C[ 1 ] + C[ 2 ] );
```

Untuk membagi nilai C[6] dengan 2 dan menetapkan hasilnya ke variabel x, kita akan menulis

```
x = C[ 6 ] / 2;
```

▪ Mendeklarasikan Array

Array menempati ruang di memori. Untuk menentukan tipe elemen dan jumlah elemen yang diperlukan oleh array, gunakan deklarasi dalam bentuk

```
type arrayName [ arraySize ] ;
```

Kompiler mencadangkan jumlah memori yang sesuai. (Ingatlah bahwa deklarasi, yang menyimpan memori lebih dikenal sebagai definisi). `arraySize` harus berupa konstanta bilangan bulat yang lebih besar dari nol. Misalnya, untuk memberitahu kompiler agar mencadangkan 11 elemen untuk array bilangan bulat `C`, gunakan deklarasi

```
int C[ 12 ]; // C is an array of 12 integers
```

Array dapat dideklarasikan berisi nilai-nilai tipe data non-referensi apa pun. Misalnya, array bertipe string dapat digunakan untuk menyimpan string karakter.

▪ Contoh Menggunakan Array

Bagian ini memberikan banyak contoh yang menunjukkan cara mendeklarasikan, menginisialisasi, dan memanipulasi array.

Contoh 1: Mendeklarasikan Array dan menggunakan Loop untuk Inisialisasi Elemen Array

Program ini mendeklarasikan array bilangan bulat 10 elemen `n`. Baris a–b menggunakan pernyataan `For` untuk menginisialisasi elemen array ke nol. Seperti variabel otomatis lainnya, array otomatis tidak secara implisit diinisialisasi ke nol. Pernyataan keluaran pertama (baris c) menampilkan judul kolom untuk kolom yang dicetak pada pernyataan `for` berikutnya (baris d–e), yang mencetak larik dalam format tabel.

```
int n[ 10 ] ; // n adalah array dari 10 bilangan bulat
void setup () {

}
void loop () {
    for ( int i = 0; i < 10; ++i ) // Menginisialisasi elemen array
n ke 0
    {
        n[ i ] = 0; // Atur elemen di lokasi I ke 0
        Serial.print (i) ;
        Serial.print ( '\r' ) ;
    }
    for ( int j = 0; j < 10; ++j ) // menampilkan nilai setiap
elemen array
```

```
{
    Serial.print (n[j]) ;
    Serial.print ('\r') ;
}
}
```

Hasil:

Element	Value
0	0
1	0
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0

Contoh 2: Menginisialisasi Array dalam Deklarasi dengan Daftar Penginisialisasi

Elemen array juga dapat diinisialisasi dalam deklarasi array dengan mengikuti nama array dengan tanda sama dengan dan daftar inisialisasi yang dipisahkan tanda kurung kurawal dan dipisahkan koma. Program ini menggunakan daftar penginisialisasi untuk menginisialisasi array integer dengan 10 nilai (baris a) dan mencetak array dalam format tabel (baris b–c).

```
// n adalah array dari 10 bilangan bulat
int n[ 10 ] = { 32, 27, 64, 18, 95, 14, 90, 70, 60, 37 } ;
void setup () {
}
void loop () {
    for ( int i = 0; i < 10; ++i ) {
```

```

        Serial.print (i) ;
        Serial.print ('\r') ;
    }
    for ( int j = 0; j < 10; ++j ) // menampilkan nilai setiap
    elemen array
    {
        Serial.print (n[j]) ;
        Serial.print ('\r') ;
    }
}

```

Hasil:

Element	Value
0	32
1	27
2	64
3	18
4	95
5	14
6	90
7	70
8	60
9	37

Contoh 3: Menjumlahkan Elemen Array

Seringkali, elemen array mewakili serangkaian nilai yang akan digunakan dalam perhitungan. Misalnya, jika elemen array mewakili nilai ujian, seorang profesor mungkin ingin menjumlahkan elemen array dan menggunakan jumlah tersebut untuk menghitung rata-rata kelas untuk ujian tersebut. Program ini menjumlahkan nilai yang terkandung dalam array bilangan bulat 10 elemen a .

```

const int arraySize = 10; // variabel konstan yang menunjukkan
ukuran array
int a[ arraySize ] = { 87, 68, 94, 100, 83, 78, 85, 91, 76, 87 };
int total = 0;
void setup () {

```

```

}
void loop () {
    // Menjumlahkan konten array a
    for ( int i = 0; i < arraySize; ++i )
        total += a[ i ];
    Serial.print ( "Total of array elements : " ) ;
    Serial.print( total ) ;
}

```

Hasil:

Total of array elements: 849

Array penting bagi Arduino dan memerlukan lebih banyak perhatian. Konsep penting berikut terkait array harus jelas bagi Arduino

Tabel 1. 14 Konsep dan dekskripsi array

S.NO.	Konsep & Deskripsi
1	<u>Passing Arrays to Functions</u> Untuk meneruskan argumen array ke fungsi, tentukan nama array tanpa tanda kurung.
2	<u>Multi-Dimensional Arrays</u> Array dengan dua dimensi (yaitu, subskrip) sering mewakili tabel nilai yang terdiri dari informasi yang disusun dalam baris dan kolom.

BAB 2

FUNGSI LIBRARY ARDUINO

A. Fungsi I/O

Pin pada papan Arduino dapat dikonfigurasi sebagai input atau output. Kami akan menjelaskan fungsi pin dalam mode tersebut. Penting untuk dicatat bahwa sebagian besar pin analog Arduino dapat dikonfigurasi dan digunakan dengan cara yang persis sama seperti pin digital.

- **Pin Dikonfigurasi sebagai INPUT**

Pin Arduino secara default dikonfigurasi sebagai input, sehingga tidak perlu dinyatakan secara eksplisit sebagai input dengan `pinMode()` saat Anda menggunakannya sebagai input. Pin yang dikonfigurasi dengan cara ini dikatakan berada dalam keadaan impedansi tinggi. Pin masukan memberikan tuntutan yang sangat kecil pada rangkaian yang diambil sampelnya, setara dengan resistor seri 100 megaohm di depan pin.

Artinya, arus yang dibutuhkan sangat sedikit untuk mengalihkan pin input dari satu keadaan ke keadaan lainnya. Hal ini membuat pin berguna untuk tugas-tugas seperti penerapan sensor sentuh kapasitif atau membaca LED sebagai fotodiode.

Pin yang dikonfigurasi sebagai `pinMode(pin, INPUT)` tanpa ada koneksi apa pun, atau dengan kabel terhubung ke pin yang tidak terhubung ke sirkuit lain, melaporkan perubahan status pin yang tampaknya acak, menangkap gangguan listrik dari lingkungan, atau menggabungkan status secara kapasitif dari pin terdekat.

- **Resistor Pull-up**

Resistor pull-up sering kali berguna untuk mengarahkan pin masukan ke keadaan yang diketahui jika tidak ada masukan. Hal ini dapat dilakukan dengan menambahkan resistor pull-up (hingga +5V), atau resistor pull-down (resistor ke ground) pada input. Resistor 10K adalah nilai yang baik untuk resistor pull-up atau pull-down.

- **Menggunakan Resistor Pull-up Bawaan dengan Pin yang Dikonfigurasi sebagai Input**

Terdapat 20.000 resistor pull-up yang terpasang pada chip Atmega yang dapat diakses dari perangkat lunak. Resistor pull-up bawaan ini diakses dengan mengatur `pinMode()` sebagai `INPUT_PULLUP`. Hal ini secara efektif membalikkan perilaku

mode INPUT, dimana HIGH berarti sensor dalam keadaan OFF dan LOW berarti sensor dalam keadaan ON. Nilai pull-up ini tergantung pada mikrokontroler yang digunakan. Pada sebagian besar papan berbasis AVR, nilainya dijamin antara 20k Ω dan 50k Ω . Di Arduino Due, nilainya antara 50k Ω dan 150k Ω . Untuk nilai pastinya, lihat lembar data mikrokontroler di board Anda.

Saat menghubungkan sensor ke pin yang dikonfigurasi dengan INPUT_PULLUP, ujung lainnya harus dihubungkan ke ground. Dalam kasus saklar sederhana, hal ini menyebabkan pin terbaca HIGH ketika saklar terbuka dan LOW ketika saklar ditekan. Resistor pull-up menyediakan arus yang cukup untuk menyalakan LED yang terhubung secara redup ke pin yang dikonfigurasi sebagai input. Jika LED dalam suatu proyek tampak berfungsi, namun sangat redup, kemungkinan inilah yang terjadi.

Register yang sama (lokasi memori chip internal) yang mengontrol apakah pin TINGGI atau RENDAH mengontrol resistor pull-up. Akibatnya, pin yang dikonfigurasi untuk memiliki resistor pull-up dihidupkan ketika pin berada dalam mode INPUT, akan memiliki pin yang dikonfigurasi sebagai TINGGI jika pin tersebut kemudian dialihkan ke mode OUTPUT dengan pinMode(). Ini juga berfungsi dalam arah lain, dan pin keluaran yang dibiarkan dalam keadaan TINGGI akan mengatur resistor pull-up jika dialihkan ke masukan dengan pinMode().

Contoh:

```
pinMode(3,INPUT) ; //set Atur pin ke input tanpa menggunakan resistor
pull up bawaan
pin to input without using built in pull up resistor
pinMode(5,INPUT_PULLUP) ; //Atur Pin ke Input Menggunakan Resistor Pull
Up bawaan
```

- **Pin Dikonfigurasi sebagai OUTPUT**

Pin yang dikonfigurasi sebagai OUTPUT dengan pinMode() dikatakan berada dalam keadaan impedansi rendah. Ini berarti bahwa mereka dapat memberikan sejumlah besar arus ke sirkuit lain. Pin Atmega dapat sumber (menyediakan arus positif) atau menenggelamkan (menyediakan arus negatif) hingga 40 mA (miliamps) arus ke perangkat/rangkaian lain. Arus ini cukup untuk menyalakan LED dengan terang (jangan lupa resistor seri), atau menjalankan banyak sensor tetapi arusnya tidak cukup untuk menjalankan relay, solenoid, atau motor.

Mencoba menjalankan perangkat arus tinggi dari pin keluaran, dapat merusak atau menghancurkan transistor keluaran pada pin tersebut, atau merusak seluruh chip Atmega. Seringkali, hal ini mengakibatkan pin "mati" pada

mikrokontroler tetapi chip yang tersisa masih berfungsi dengan baik. Untuk alasan ini, merupakan ide bagus untuk menghubungkan pin OUTPUT ke perangkat lain melalui resistor 470Ω atau $1k$, kecuali arus maksimum yang diambil dari pin diperlukan untuk aplikasi tertentu.

▪ Fungsi pinMode()

Fungsi pinMode() digunakan untuk mengonfigurasi pin tertentu agar berperilaku sebagai input atau output. Dimungkinkan untuk mengaktifkan resistor pull-up internal dengan mode INPUT_PULLUP. Selain itu, mode INPUT secara eksplisit menonaktifkan pull-up internal.

Fungsi sintaks pinMode() :

```
Void setup () {  
    pinMode (pin , mode);  
}
```

- **pin** - nomor pin yang modenya ingin Anda atur
- **mode** - INPUT, OUTPUT, atau INPUT_PULLUP.

Contoh:

```
int button = 5 ; // Tombol terhubung ke pin 5  
int LED = 6; // LED terhubung ke pin 6  
  
void setup () {  
    pinMode(button , INPUT_PULLUP);  
    // Atur pin digital sebagai input dengan resistor pull-up  
    pinMode(button , OUTPUT); // Atur Pin Digital sebagai output  
}  
  
void setup () {  
    If (digitalRead(button ) == LOW) // jika tombol ditekan  
    {  
        digitalWrite(LED,HIGH); // Mengaktifkan LED  
        delay(500); // penundaan selama 500 ms  
        digitalWrite(LED,LOW); // Matikan LED  
        delay(500); // penundaan selama 500 ms  
    }  
}
```

▪ Fungsi digitalWrite()

Fungsi **digitalWrite()** digunakan untuk menulis nilai HIGH atau LOW ke pin digital. Jika pin telah dikonfigurasi sebagai OUTPUT dengan [pinMode\(\)](#), tegangannya akan diatur ke nilai yang sesuai: 5V (atau 3,3V pada papan 3,3V) untuk HIGH, 0V (ground) untuk LOW. Jika pin dikonfigurasi sebagai INPUT, digitalWrite() akan mengaktifkan (HIGH) atau menonaktifkan (LOW) pullup internal pada pin input. Dianjurkan untuk mengatur [pinMode\(\)](#) ke INPUT_PULLUP untuk mengaktifkan resistor pull-up internal.

Jika Anda tidak menyetel pinMode() ke OUTPUT, dan menghubungkan LED ke pin, saat memanggil digitalWrite(HIGH), LED mungkin tampak redup. Tanpa menyetel pinMode() secara eksplisit, digitalWrite() akan mengaktifkan resistor pull-up internal, yang bertindak seperti resistor pembatas arus yang besar.

Sintaks fungsi digitalWrite():

```
Void loop() {  
    digitalWrite (pin ,value);  
}
```

- **pin** - nomor pin yang modenyanya ingin Anda atur
- **value** - TINGGI, atau RENDAH.

Contoh:

```
int LED = 6; // LED terhubung ke pin 6  
  
void setup () {  
    pinMode(LED, OUTPUT); // Atur Pin Digital sebagai output  
}  
  
void loop () {  
    digitalWrite(LED,HIGH); // Mengaktifkan LED  
    delay(500); // penundaan selama 500 ms  
    digitalWrite(LED,LOW); // Matikan LED  
    delay(500); // penundaan selama 500 ms  
}
```

▪ Fungsi analogRead()

Arduino mampu mendeteksi apakah ada tegangan yang diberikan ke salah satu pinnya dan melaporkannya melalui fungsi `digitalRead()`. Ada perbedaan antara sensor on/off (yang mendeteksi keberadaan suatu benda) dengan sensor analog yang nilainya terus berubah. Untuk membaca sensor jenis ini, kita memerlukan jenis pin yang berbeda.

Di bagian kanan bawah papan Arduino, Anda akan melihat enam pin bertanda “Analog In”. Pin khusus ini tidak hanya memberi tahu apakah ada tegangan yang diberikan padanya, tetapi juga nilainya. Dengan menggunakan fungsi **`analogRead()`**, kita dapat membaca tegangan yang diberikan ke salah satu pin.

Fungsi ini mengembalikan angka antara 0 dan 1023, yang mewakili tegangan antara 0 dan 5 volt. Misalnya, jika ada tegangan 2,5 V diterapkan ke pin nomor 0, `analogRead(0)` mengembalikan 512.

Sintaks fungsi `analogRead()` :

```
analogRead(pin);
```

- **pin** - jumlah pin input analog yang akan dibaca (0 hingga 5 pada sebagian besar papan, 0 hingga 7 pada Mini dan Nano, 0 hingga 15 pada Mega)

Contoh:

```
int analogPin = 3; //Wiper potensiometer (terminal tengah)
// Terhubung ke Pin Analog 3
int val = 0; // variabel untuk menyimpan nilai yang dibaca

void setup() {
    Serial.begin(9600); // Serial penyiapan
}

void loop() {
    val = analogRead(analogPin); // Baca Pin Input
    Serial.println(val); // nilai debug
}
```

B. Fungsi I/O Tingkat Lanjut

Dalam bab ini, kita akan mempelajari beberapa Fungsi Input dan Output tingkat lanjut.

- Fungsi `analogReference()`

Mengonfigurasi tegangan referensi yang digunakan untuk input analog (yaitu nilai yang digunakan sebagai bagian atas rentang input). Pilihannya adalah –

- **DEFAULT** - Referensi analog default 5 volt (pada papan Arduino 5V) atau 3,3 volt (pada papan Arduino 3,3V)
- **INTERNAL** - Referensi bawaan, sama dengan 1,1 volt pada ATmega168 atau ATmega328 dan 2,56 volt pada ATmega8 (tidak tersedia di Arduino Mega)
- **INTERNAL1V1** - Referensi 1.1V bawaan (khusus Arduino Mega)
- **INTERNAL2V56** - Referensi 2.56V bawaan (hanya Arduino Mega)
- **EXTERNAL** - Tegangan yang diberikan ke pin AREF (hanya 0 hingga 5V) digunakan sebagai referensi

Sintaks fungsi `analogReference()`:

```
analogReference (type);
```

- **type** - dapat menggunakan semua jenis berikut (DEFAULT, INTERNAL, INTERNAL1V1, INTERNAL2V56, EXTERNAL)

Jangan gunakan tegangan kurang dari 0V atau lebih dari 5V untuk tegangan referensi eksternal pada pin AREF. Jika Anda menggunakan referensi eksternal pada pin AREF, Anda harus mengatur referensi analog ke EXTERNAL sebelum memanggil fungsi **`analogRead()`** . Jika tidak, Anda akan menyebabkan korsleting pada tegangan referensi aktif (dihasilkan secara internal) dan pin AREF, yang mungkin merusak mikrokontroler pada papan Arduino Anda.

Gambar 2. 1 Posisi pin analog *reference*

Alternatifnya, Anda dapat menghubungkan tegangan referensi eksternal ke pin AREF melalui resistor 5K, memungkinkan Anda untuk beralih antara tegangan referensi eksternal dan internal.

Perhatikan bahwa resistor akan mengubah tegangan yang digunakan sebagai referensi karena terdapat resistor internal 32K pada pin AREF. Keduanya bertindak sebagai pembagi tegangan. Misalnya, 2,5V yang diterapkan melalui resistor akan menghasilkan $2,5 * 32 / (32 + 5) = \sim 2,2V$ pada pin AREF.

Contoh:

```
int analogPin = 3; // Wiper potensiometer (terminal tengah)
terhubung ke pin analog 3
int val = 0; // variabel untuk menyimpan nilai baca

void setup() {
    Serial.begin(9600); // Serial penyiapan
    analogReference(EXTERNAL); // tegangan yang diterapkan ke pin
    AREF (hanya 0 hingga 5V)
    // digunakan sebagai referensi.
}

void loop() {
    val = analogRead(analogPin); // Baca Pin Input
    Serial.println(val); // nilai debug
}
```

C. Fungsi Character

Semua data dimasukkan ke dalam komputer sebagai karakter, yang meliputi huruf, angka, dan berbagai simbol khusus. Pada bagian ini, kita membahas kemampuan C++ untuk memeriksa dan memanipulasi karakter individu.

Pustaka penanganan karakter mencakup beberapa fungsi yang melakukan pengujian dan manipulasi data karakter yang berguna. Setiap fungsi menerima karakter, direpresentasikan sebagai int, atau EOF sebagai argumen. Karakter sering dimanipulasi sebagai bilangan bulat.

Ingatlah bahwa EOF biasanya memiliki nilai -1 dan beberapa arsitektur perangkat keras tidak mengizinkan nilai negatif disimpan dalam variabel char. Oleh karena itu, fungsi penanganan karakter memanipulasi karakter sebagai bilangan bulat.

Tabel berikut merangkum fungsi perpustakaan penanganan karakter. Saat menggunakan fungsi dari pustaka penanganan karakter, sertakan header `<cctype>`.

Tabel 2. 1 Library fungsi character

S.No.	Prototipe & Deskripsi
1	int isdigit(int c) Mengembalikan 1 jika c adalah digit dan 0 jika tidak.
2	int isalpha(int c) Mengembalikan 1 jika c adalah huruf dan 0 jika tidak.
3	int isalnum(int c) Mengembalikan 1 jika c adalah digit atau huruf dan 0 jika tidak.
4	int isxdigit(int c) Mengembalikan 1 jika c adalah karakter digit heksadesimal dan 0 jika tidak. (Lihat Lampiran D, Sistem Bilangan, untuk penjelasan rinci tentang bilangan biner, oktal, desimal dan heksadesimal.)
5	int lebih rendah (int c) Mengembalikan 1 jika c adalah huruf kecil dan 0 jika tidak.
6	int isupper(int c) Mengembalikan 1 jika c adalah huruf besar; 0 sebaliknya.
7	int isspace (int c) Mengembalikan 1 jika c adalah karakter spasi kosong—baris baru ('\n'), spasi (' '), umpan formulir ('\f'), carriage return ('\r'), tab horizontal ('\t'), atau tab vertikal ('\v')—dan 0 sebaliknya.
8	int iscntrl (int c) Mengembalikan 1 jika c adalah karakter kontrol, seperti baris baru ('\n'), umpan formulir ('\f'), pengangkutan kembali ('\r'), tab horizontal ('\t'), tab vertikal ('\v'), alert ('\a'), atau backspace ('\b')—dan 0 sebaliknya.
9	int ispunct(int c) Mengembalikan 1 jika c adalah karakter pencetakan selain spasi, digit, atau huruf dan 0 sebaliknya.
10	int isprint (int c) Mengembalikan 1 jika c adalah karakter pencetakan termasuk spasi (' ') dan 0 jika tidak.
11	int isgraf(int c) Mengembalikan 1 jika c adalah karakter pencetakan selain spasi (' ') dan 0 jika tidak.

Contoh 1 berikut menunjukkan penggunaan fungsi **isdigit**, **isalpha**, **isalnum** dan **isxdigit**. Fungsi **isdigit** menentukan apakah argumennya berupa digit (0–9). Fungsi **isalpha** menentukan apakah argumennya berupa huruf besar (A–Z) atau huruf kecil (a–z). Fungsi **isalnum** menentukan apakah argumennya berupa huruf besar, huruf kecil, atau angka. Fungsi **isxdigit** menentukan apakah argumennya berupa digit heksadesimal (A–F, a–f, 0–9).

Contoh 1:

```
void setup () {
  Serial.begin (9600);
  Serial.print ("According to isdigit:\r");
  Serial.print (isdigit( '8' ) ? "8 is a": "8 is not a");
  Serial.print (" digit\r" );
  Serial.print (isdigit( '#' ) ?"# is a": "# is not a" );
  Serial.print (" digit\r");
  Serial.print ("\rAccording to isalpha:\r" );
  Serial.print (isalpha('A' ) ?"A is a": "A is not a");
  Serial.print (" letter\r");
  Serial.print (isalpha('a' ) ?"b is a": "b is not a");
  Serial.print (" letter\r");
  Serial.print (isalpha('&') ?"& is a": "& is not a");
  Serial.print (" letter\r");
  Serial.print (isalpha( '4' ) ?"4 is a": "4 is not a");
  Serial.print (" letter\r");
  Serial.print ("\rAccording to isalnum:\r");
  Serial.print (isalnum( 'A' ) ?"A is a" : "A is not a" );

  Serial.print (" digit or a letter\r" );
  Serial.print (isalnum( '8' ) ?"8 is a" : "8 is not a" );
  Serial.print (" digit or a letter\r");
  Serial.print (isalnum( '#' ) ?"# is a" : "# is not a" );
  Serial.print (" digit or a letter\r");
  Serial.print ("\rAccording to isxdigit:\r");
  Serial.print (isxdigit( 'F' ) ?"F is a" : "F is not a" );
  Serial.print (" hexadecimal digit\r" );
  Serial.print (isxdigit( 'J' ) ?"J is a" : "J is not a" );
  Serial.print (" hexadecimal digit\r" );
}
```

```

    Serial.print (isdigit( '7' ) ? "7 is a" : "7 is not a" ) ;

    Serial.print ( " hexadecimal digit\r" );
    Serial.print (isdigit( '$' ) ? "$ is a" : "$ is not a" );
    Serial.print ( " hexadecimal digit\r" );
    Serial.print (isdigit( 'f' ) ? "f is a" : "f is not a");

}
void loop () {
}

```

Hasil 1:

```

According to isdigit:
8 is a digit
# is not a digit
According to isalpha:
A is a letter
b is a letter
& is not a letter
4 is not a letter
According to isalnum:
A is a digit or a letter
8 is a digit or a letter
# is not a digit or a letter
According to isxdigit:
F is a hexadecimal digit
J is not a hexadecimal digit
7 is a hexadecimal digit
$ is not a hexadecimal digit
f is a hexadecimal digit

```

Kami menggunakan operator kondisional (?:) dengan setiap fungsi untuk menentukan apakah string " adalah " atau string " bukan " harus dicetak dalam output untuk setiap karakter yang diuji. Misalnya, baris **a** menunjukkan bahwa jika '8' adalah sebuah digit—yaitu, jika **isdigit** mengembalikan nilai yang benar (bukan nol)—string "8 adalah a " akan dicetak. Jika '8' bukan sebuah digit (misalnya, jika **isdigit** menghasilkan 0), string " 8 bukan sebuah " akan dicetak.

Contoh 2 berikut menunjukkan penggunaan fungsi **islower** dan **isupper** . Fungsi **islower** menentukan apakah argumennya berupa huruf kecil (a–z). Fungsi **isupper** menentukan apakah argumennya berupa huruf besar (A–Z).

Contoh 2:

```
int thisChar = 0xA0;
void setup () {
    Serial.begin (9600);
    Serial.print ("According to islower:\r") ;
    Serial.print (islower( 'p' ) ? "p is a" : "p is not a" );
    Serial.print ( " lowercase letter\r" );
    Serial.print ( islower( 'P' ) ? "P is a" : "P is not a" ) ;
    Serial.print ("lowercase letter\r");
    Serial.print (islower( '5' ) ? "5 is a" : "5 is not a" );
    Serial.print ( " lowercase letter\r" );
    Serial.print ( islower( '!' )? "! is a" : "! is not a" ) ;
    Serial.print ("lowercase letter\r");

    Serial.print ("\rAccording to isupper:\r") ;
    Serial.print (isupper ( 'D' ) ? "D is a" : "D is not an" );
    Serial.print ( " uppercase letter\r" );
    Serial.print ( isupper ( 'd' )? "d is a" : "d is not an" ) ;
    Serial.print ( " uppercase letter\r" );
    Serial.print (isupper ( '8' ) ? "8 is a" : "8 is not an" );
    Serial.print ( " uppercase letter\r" );
    Serial.print ( islower( '$' )? "$ is a" : "$ is not an" ) ;
    Serial.print ("uppercase letter\r " );
}
void setup () {
}
```

Hasil 2:

```
According to islower:
p is a lowercase letter
P is not a lowercase letter
```

5 is not a lowercase letter

! is not a lowercase letter

According to isupper:

D is an uppercase letter

d is not an uppercase letter

8 is not an uppercase letter

\$ is not an uppercase letter

Contoh 3 berikut menunjukkan penggunaan fungsi **isspace**, **isctrl**, **ispunct**, **isprint** dan **isgraph** .

- Fungsi **isspace** menentukan apakah argumennya berupa karakter spasi, seperti spasi (' '), form feed ('\f'), baris baru ('\n'), carriage return ('\r'), tab horizontal ('\t') atau tab vertikal ('\v').
- Fungsi **isctrl** menentukan apakah argumennya merupakan karakter kontrol seperti tab horizontal ('\t'), tab vertikal ('\v'), form feed ('\f'), alert ('\a'), backspace ('\b'), pengangkutan kembali ('\r') atau baris baru ('\n').
- Fungsi **ispunct** menentukan apakah argumennya berupa karakter pencetakan selain spasi, angka, atau huruf, seperti \$, #, (,), [,], {, }, :, ; atau %.
- Fungsi **isprint** menentukan apakah argumennya berupa karakter yang dapat ditampilkan di layar (termasuk karakter spasi).
- Fungsi **isgraph** menguji karakter yang sama dengan isprint, namun karakter spasi tidak disertakan.

Contoh 3:

```
void setup () {  
  Serial.begin (9600);  
  Serial.print ( " According to isspace:\rNewline " ) ;  
  Serial.print (isspace( '\n' )? " is a" : " is not a" );  
  Serial.print ( " whitespace character\rHorizontal tab" ) ;  
  Serial.print (isspace( '\t' )? " is a" : " is not a" );  
  Serial.print ( " whitespace character\n" ) ;  
  Serial.print (isspace('%')? " % is a" : " % is not a" );  
  
  Serial.print ( " \rAccording to isctrl:\rNewline" ) ;  
  Serial.print ( isctrl( '\n' )?"is a" : " is not a" ) ;
```

```

Serial.print (" control character\r");
Serial.print (iscntrl( '$' ) ? " $ is a" : " $ is not a" );
Serial.print (" control character\r");
Serial.print ("\rAccording to ispunct:\r");
Serial.print (ispunct(';') ? "; is a" : "; is not a" );
Serial.print (" punctuation character\r");
Serial.print (ispunct('Y') ? "Y is a" : "Y is not a" );
Serial.print ("punctuation character\r");
Serial.print (ispunct('#') ? "# is a" : "# is not a" );
Serial.print ("punctuation character\r");

Serial.print ( "\r According to isprint:\r");
Serial.print (isprint('$') ? "$ is a" : "$ is not a" );
Serial.print (" printing character\rAlert ");
Serial.print (isprint('\a') ? " is a" : " is not a" );
Serial.print (" printing character\rSpace ");
Serial.print (isprint(' ') ? " is a" : " is not a" );
Serial.print (" printing character\r");

Serial.print ("\r According to isgraph:\r");
Serial.print (isgraph ('Q') ? "Q is a" : "Q is not a" );
Serial.print ("printing character other than a space\rSpace
");
Serial.print (isgraph (' ') ? " is a" : " is not a" );
Serial.print ("printing character other than a space ");
}
void loop () {
}

```

Hasil 3:

According to isspace:
 Newline is a whitespace character
 Horizontal tab is a whitespace character
 % is not a whitespace character
 According to iscntrl:
 Newline is a control character
 \$ is not a control character

According to ispunct:

; is a punctuation character

Y is not a punctuation character

is a punctuation character

According to isprint:

\$ is a printing character

Alert is not a printing character

Space is a printing character

According to isgraph:

Q is a printing character other than a space

Space is not a printing character other than a space

D. Math Library

Pustaka Arduino Math (math.h) mencakup sejumlah fungsi matematika yang berguna untuk memanipulasi bilangan floating-point.

- **Library Macros**

Berikut adalah makro yang ditentukan di header math.h

Tabel 2. 2 Daftar librari makro untuk math.h

Macro	Nilai	Deskripsi
M_E	2.7182818284590452354	Konstanta e.
M_LOG2E	1.4426950408889634074 /* log_2 e */	Logaritma e ke basis 2
M_1_PI	0.31830988618379067154 /* 1/pi */	Konstanta 1/pi
M_2_PI	0.63661977236758134308 /* 2/pi */	Konstanta 2/pi
M_2_SQRTPI	1.12837916709551257390 /* 2/sqrt(pi) */	Panduan Cepat Konstanta 2/sqrt(pi)
M_LN10	2.30258509299404568402 /* log_e 10 */	Logaritma natural dari 10
M_LN2	0.69314718055994530942 /* log_e 2 */	Logaritma natural dari 2
	0.43429448190325182765	

Macro	Nilai	Deskripsi
M_LOG10E	<code>/* log_10 e */</code>	Logaritma e ke basis 10
M_PI	3.14159265358979323846 <code>/*Pi*/</code>	Konstanta pi
M_PI_2	3.3V1.57079632679489661923 <code>/* pi/2 */</code>	Konstanta pi/2
M_PI_4	0.78539816339744830962 <code>/* pi/4 */</code>	Konstanta pi/4
M_SQRT1_2	0.70710678118654752440 <code>/* 1/sqrt(2) */</code>	Konstanta 1/sqrt(2)
M_SQRT2	1.41421356237309504880 <code>/* sqrt(2) */</code>	Akar kuadrat dari 2
acosf	-	Alias untuk fungsi acos()
asinf	-	Fungsi alias untuk asin()
atan2f	-	Alias untuk fungsi atan2()
cbrtf	-	Alias untuk fungsi cbrt()
ceilf	-	Fungsi alias untuk ceil()
copysignf	-	Fungsi alias untuk copysign()
coshf	-	Alias untuk fungsi cosh()
expf	-	Fungsi alias untuk exp()
fabsf	-	Fungsi alias untuk fabs()
fdimf	-	Fungsi alias untuk fdim()
floorf	-	Fungsi alias untuk floor()
fmaxf	-	Fungsi alias untuk fmax()
fminf	-	Fungsi alias untuk fmin()
fmodf	-	Fungsi alias untuk fmod()
frexpf	-	Fungsi alias untuk frexp()
hypotf	-	Alias untuk fungsi hypot()
INFINITY	-	Konstanta INFINITY
isfinitef	-	Alias untuk fungsi isfinite()
isinf	-	Fungsi alias untuk isinf()
isnanf	-	Fungsi alias untuk isnan()
ldexpf	-	Fungsi alias untuk ldexp()

Macro	Nilai	Deskripsi
log10f	-	Alias untuk fungsi log10()
logf	-	Fungsi alias untuk log()
lrintf	-	Alias untuk fungsi lrint()
lroundf	-	Alias untuk fungsi lround()

- **Library Functions**

Fungsi berikut didefinisikan di header **math.h**

Tabel 2. 3 deskripsi dan sungsi dari library math.h

S.No.	Fungsi & Deskripsi Perpustakaan
1	double acos (Double __x) Fungsi acos() menghitung nilai utama busur kosinus dari __x. Nilai yang dikembalikan berada dalam kisaran [0, pi] radian. Terjadi kesalahan domain
2	double asin (double __x) Fungsi asin() computa nilai utama sinus busur __x. Nilai yang dikembalikan berada dalam kisaran. Kesalahan domain terjadi untuk argumen yang tidak berada dalam rentang [-1, +1].
3	double atan (double __x) Fungsi atan() menghitung nilai utama tangen busur __x. Nilai yang dikembalikan berada dalam rentang radian [-pi/2, pi/2].
4	double atan2 (double __y, double __x) Fungsi atan2() menghitung nilai pokok tangen busur dari __y / __x, menggunakan tanda-tanda kedua argumen untuk menentukan kuadran nilai kembali. Nilai yang dikembalikan berada dalam rentang radian [-pi, +pi].
5	double cbrt (double __x) Fungsi cbrt() mengembalikan akar kubus dari __x.
6	double ceil (double __x) Fungsi ceil() mengembalikan nilai integral terkecil yang lebih besar dari atau sama dengan __x, dinyatakan sebagai bilangan floating-point.
7	static double copysign (double __x, double __y) Fungsi copysign() mengembalikan __x tetapi dengan tanda __y. Mereka bekerja bahkan jika __x atau __y adalah NaN atau nol.

S.No.	Fungsi & Deskripsi Perpustakaan
	double cos(double __x)
8	Fungsi cos() mengembalikan kosinus __x, diukur dalam radian.
	double cosh (double __x)
9	Fungsi cosh() mengembalikan kosinus hiperbolik __x.
	double exp (double __x)
10	Fungsi exp() mengembalikan nilai eksponensial __x.
	double fabs (double __x)
11	Fungsi fabs() menghitung nilai absolut dari bilangan floating-point __x.
	double fdim (double __x, double __y)
12	Fungsi fdim() mengembalikan max(__x - __y, 0). Jika __x atau __y atau keduanya adalah NaN, NaN dikembalikan.
	double floor (double __x)
13	Fungsi floor() mengembalikan nilai integral terbesar kurang dari atau sama dengan __x, dinyatakan sebagai bilangan floating-point.
	double fma (double __x, double __y, double __z)
14	Fungsi fma() melakukan floating-point multiply-add. Ini adalah operasi ((__x * __y) + __z), tetapi hasil antara tidak dibulatkan ke jenis tujuan. Ini terkadang dapat meningkatkan ketepatan perhitungan.
	double fmax (double __x, double __y)
15	Fungsi fmax() mengembalikan nilai yang lebih besar dari dua nilai __x dan __y. Jika argumen adalah NaN, argumen lainnya dikembalikan. Jika kedua argumen adalah NaN, NaN dikembalikan.
	double fmin (double __x, double __y)
16	Fungsi fmin() mengembalikan nilai yang lebih rendah dari dua nilai __x dan __y. Jika argumen adalah NaN, argumen lainnya dikembalikan. Jika kedua argumen adalah NaN, NaN dikembalikan.
	double fmod (double __x, double __y)
17	Fungsi fmod() mengembalikan sisa floating-point dari __x / __y.
	double frexp (double __x, int * __pexp)
	Fungsi frexp() memecah bilangan floating-point menjadi pecahan yang dinormalisasi dan pangkat integral 2. Ini menyimpan bilangan bulat dalam objek int yang ditunjukkan oleh __pexp. Jika __x adalah bilangan float point normal, fungsi frexp() mengembalikan nilai v, sehingga v memiliki besaran dalam interval [1/2, 1) atau nol, dan __x sama dengan v kali 2 dipangkatkan __pexp. Jika __x adalah nol, kedua bagian hasilnya adalah nol. Jika __x

S.No.	Fungsi & Deskripsi Perpustakaan
18	bukan bilangan terbatas, frexp() mengembalikan ___x apa adanya dan menyimpan 0 dengan ___pexp. Catatan - Implementasi ini memungkinkan pointer nol sebagai arahan untuk melewati penyimpanan eksponen.
19	double hypot (double ___x, double ___y) Fungsi hypot() mengembalikan $\sqrt{___x^2 + ___y^2}$. Ini adalah panjang sisi miring segitiga siku-siku dengan sisi panjang ___x dan ___y, atau jarak titik (___x, ___y) dari titik asal. Menggunakan fungsi ini alih-alih rumus langsung adalah bijaksana, karena kesalahannya jauh lebih kecil. Tidak ada aliran bawah dengan ___x kecil dan ___y. Tidak ada luapan jika hasilnya dalam jangkauan.
20	static int isfinite (double ___x) Fungsi isfinite() mengembalikan nilai bukan nol jika ___x terbatas: bukan plus atau minus tak terhingga, dan bukan NaN.
21	Int isinf (double ___x) Fungsi isinf() mengembalikan 1 jika arduen ___x adalah positif tak terhingga, -1 jika ___x negatif tak terhingga, dan 0 sebaliknya. Catatan - GCC 4.3 dapat mengganti fungsi ini dengan kode sebaris yang mengembalikan nilai 1 untuk kedua infinitas (bug gcc #35509).
22	int isnan (double ___x) Fungsi isnan() mengembalikan 1 jika argumen ___x mewakili objek "not-a-number" (NaN), jika tidak 0.
23	double ldexp (double ___x, int ___exp) Fungsi ldexp() mengalikan bilangan floating-point dengan pangkat integral 2. Ini mengembalikan nilai ___x kali 2 yang dipangkatkan ke ___exp daya.
24	double log (double ___x) Fungsi log() mengembalikan logaritma natural argumen ___x.
25	double log10(double ___x) Fungsi log10() mengembalikan logaritma argumen ___x ke basis 10.
26	long lrint (double ___x) Fungsi lrint() membulatkan ___x ke bilangan bulat terdekat, membulatkan setengah kasus ke arah bilangan bulat genap. (Artinya, nilai 1,5 dan 2,5 dibulatkan menjadi 2). Fungsi ini mirip dengan fungsi rint(), tetapi berbeda dalam jenis nilai pengembalian dan bahwa luapan dimungkinkan. Kembali Nilai bilangan bulat panjang yang dibulatkan. Jika ___x bukan bilangan terbatas atau luapan, realisasi ini mengembalikan nilai LONG_MIN (0x80000000).

S.No.	Fungsi & Deskripsi Perpustakaan
	long lround (double __x) Fungsi lround() membulatkan __x ke bilangan bulat terdekat, tetapi membulatkan setengah kasus menjauh dari nol (bukan ke bilangan bulat genap terdekat). Fungsi ini mirip dengan fungsi round(), tetapi berbeda dalam jenis nilai pengembalian dan bahwa luapan dimungkinkan. Kembali 27 Nilai bilangan bulat panjang yang dibulatkan. Jika __x bukan bilangan terbatas atau luapan dulu, realisasi ini mengembalikan nilai LONG_MIN (0x80000000).
	double modf (double __x, double * __iptr) 28 Fungsi modf() memecah argumen x menjadi bagian integral dan pecahan, yang masing-masing memiliki tanda yang sama dengan argumen. Ini menyimpan bagian integral sebagai ganda dalam objek yang ditunjuk oleh __iptr. Fungsi modf() mengembalikan bagian pecahan bertanda tangan dari __x.
	float modff (float __x, float * __iptr) 29 Alias untuk modf().
	double pow (double __x, double __y) 30 Fungsi pow() mengembalikan nilai __x ke __y eksponen.
	double round (double __x) Fungsi round() membulatkan __x ke bilangan bulat terdekat, tetapi membulatkan setengah kasus dari nol (bukan ke bilangan bulat genap terdekat). Overflow tidak mungkin. Kembali 31 Nilai yang dibulatkan. Jika __x adalah integral atau tak terbatas, __x itu sendiri dikembalikan. Kalau __x adalah NaN, maka NaN dikembalikan.
	int signbit (double __x) 32 Fungsi signbit() mengembalikan nilai bukan nol jika nilai __x memiliki bit tanda yang ditetapkan. Ini tidak sama dengan '$__x < 0,0$', karena floating point IEEE 754 memungkinkan nol untuk ditandatangani. Perbandingan '$-0,0 < 0,0$' salah, tetapi 'signbit (-0,0)' akan mengembalikan nilai bukan nol.
	double sin (double __x) 33 Fungsi sin() mengembalikan sinus __x, diukur dalam radian.
	double sinh (double __x) 34 Fungsi sinh() mengembalikan sinus hiperbolik __x.
	double sqrt (double __x) 35 Fungsi sqrt() mengembalikan akar kuadrat non-negatif dari __x.

S.No.	Fungsi & Deskripsi Perpustakaan
36	double square (double __x) Fungsi square() mengembalikan $__x * __x$. Catatan - Fungsi ini tidak termasuk dalam definisi standar C.
37	double tan (double __x) Fungsi tan() mengembalikan tangen $__x$, diukur dalam radian.
38	double tanh (double __x) Fungsi tanh() mengembalikan tangen hiperbolik $__x$.
39	double trunc (double __x) Fungsi trunc() membulatkan $__x$ ke bilangan bulat terdekat yang tidak lebih besar dalam nilai absolut.

Contoh berikut menunjukkan cara menggunakan fungsi perpustakaan math.h yang paling umum

Contoh:

```
double double__x = 45.45 ;
double double__y = 30.20 ;

void setup() {
  Serial.begin(9600);
  Serial.print("cos num = ");
  Serial.println (cos (double__x) ); // Mengembalikan kosinus dari x
  Serial.print("absolute value of num = ");
  Serial.println (fabs (double__x) ); // Nilai absolut float
  Serial.print("floating point modulo = ");
  Serial.println (fmod (double__x, double__y)); // Modulo Titik Apung
  Serial.print("sine of num = ");
  Serial.println (sin (double__x) ) ;// Mengembalikan sinus dari x
  Serial.print("square root of num : ");
  Serial.println ( sqrt (double__x) );// Mengembalikan akar kuadrat dari
x
  Serial.print("tangent of num : ");
  Serial.println ( tan (double__x) ); // Mengembalikan nilai tangen dari
x
  Serial.print("exponential value of num : ");
  Serial.println ( exp (double__x) ); // Fungsi mengembalikan nilai
eksponensial x.
  Serial.print("cos num : ");
```

```

Serial.println (atan (double__x) ); // busur tangen x
Serial.print("tangent of num : ");
Serial.println (atan2 (double__y, double__x) );// busur tangen y / x
Serial.print("arc tangent of num : ");
Serial.println (log (double__x) ) ; // Logaritma natural x
Serial.print("cos num : ");
Serial.println ( log10 (double__x)); // Logaritma x ke basis 10.
Serial.print("logarithm of num to base 10 : ");
Serial.println (pow (double__x, double__y) );// x pangkat y
Serial.print("power of num : ");
Serial.println (square (double__x)); // kuadrat x
}
void loop() {
}

```

Hasil:

```

cos num = 0.10
absolute value of num = 45.45
floating point modulo =15.25
sine of num = 0.99
square root of num : 6.74
tangent of num : 9.67
exponential value of num : ovf
cos num : 1.55
tangent of num : 0.59
arc tangent of num : 3.82
cos num : 1.66
logarithm of num to base 10 : inf
power of num : 2065.70

```

E. Fungsi trigonometri

Anda perlu menggunakan Trigonometri secara praktis seperti menghitung jarak benda bergerak atau kecepatan sudut. Arduino menyediakan fungsi trigonometri tradisional (sin, cos, tan, asin, acos, atan) yang dapat diringkas dengan menulis prototipenya. Math.h berisi prototipe fungsi trigonometri.

Sintaks Eksak Trigonometri:

```
double sin(double x); //Mengembalikan sinus dari x radian
double cos(double y); //Mengembalikan kosinus dari y radian
double tan(double x); //Mengembalikan nilai tangen dari x radian
double acos(double x); //mengembalikan A, sudut yang sesuai dengan cos
(A) = x
double asin(double x); //mengembalikan A, sudut yang sesuai dengan sin
(A) = x
double atan(double x); //mengembalikan A, sudut yang sesuai dengan tan
(A) = x
```

Contoh:

```
double sine = sin(2); // kira-kira 0.90929737091
double cosine = cos(2); // kira-kira -0.41614685058
double tangent = tan(2); // kira-kira -2.18503975868
```

BAB 3

ARDUINO TINGKAT LANJUT

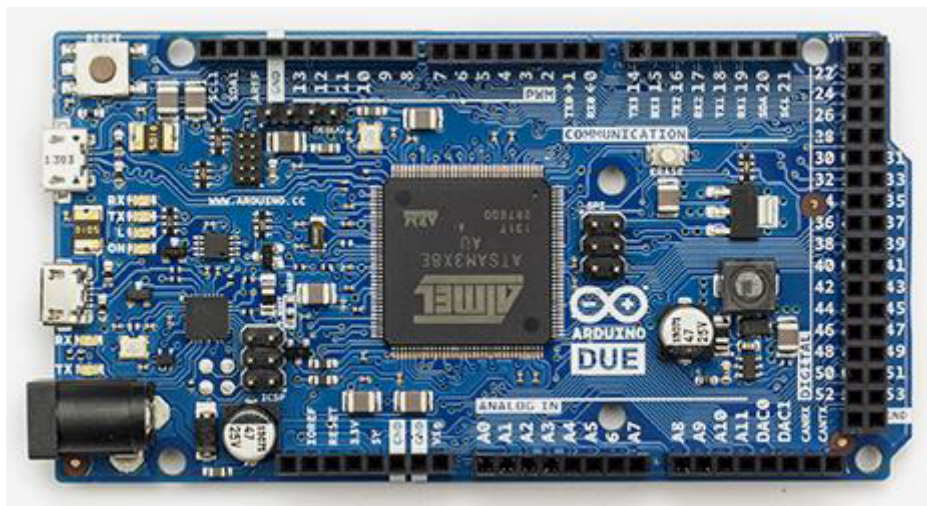
A. Arduino Due & Zero

a. Arduino Due

Arduino Due adalah papan mikrokontroler berdasarkan CPU Atmel SAM3X8E ARM Cortex-M3. Ini adalah papan Arduino pertama yang didasarkan pada mikrokontroler inti ARM 32-bit.

Fitur penting:

- Ini memiliki 54 pin input/output digital (12 di antaranya dapat digunakan sebagai output PWM)
- 12 input analog
- 4 UART (port serial perangkat keras)
- Jam 84 MHz, koneksi USB OTG yang mampu
- 2 DAC (digital ke analog), 2 TWI, colokan listrik, header SPI, header JTAG
- Tombol reset dan tombol hapus



Gambar 3. 1 Papan Arduino Due

Karakteristik Papan Arduino Due:

Volt operasi	kecepatan CPU	Masuk/keluar analog	IO/PWM Digital	EEPROM [KB]	SRAM [KB]	Kilatan [KB]	USB	UART
3,3 Volt	84Mhz	12/2	54/12	-	96	512	2 mikro	4

Komunikasi:

- 4 UART Perangkat Keras
- 2 I2C
- 1 Antarmuka CAN (Protokol komunikasi otomotif)
- 1 SPI
- 1 Antarmuka JTAG (10 pin)
- 1 Host USB (seperti Leonardo)
- 1 Pelabuhan Pemrograman

Tidak seperti kebanyakan papan Arduino, papan Arduino Due beroperasi pada 3.3V. Tegangan maksimum yang dapat ditoleransi oleh pin I/O adalah 3,3V. Menerapkan tegangan lebih tinggi dari 3,3V ke pin I/O mana pun dapat merusak board.

Papan berisi semua yang diperlukan untuk mendukung mikrokontroler. Anda cukup menghubungkannya ke komputer dengan kabel micro-USB atau menyalakannya dengan adaptor AC-DC atau baterai untuk memulai. Due kompatibel dengan semua pelindung Arduino yang bekerja pada 3.3V.

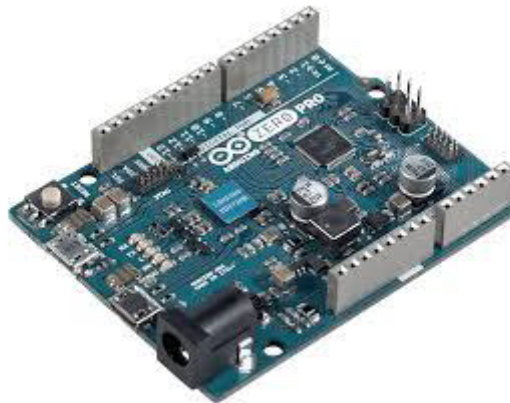
b. Arduino Zero

Arduino Zero adalah perpanjangan 32-bit yang sederhana dan kuat dari platform yang dibuat oleh UNO. Papan Zero memperluas keluarga dengan memberikan peningkatan kinerja, memungkinkan berbagai peluang proyek untuk perangkat, dan bertindak sebagai alat pendidikan yang hebat untuk mempelajari tentang pengembangan aplikasi 32-bit.

Fitur penting:

- Aplikasi Zero mencakup perangkat IoT pintar, teknologi yang dapat dikenakan, otomatisasi teknologi tinggi, hingga robotika gila.
- Board ini didukung oleh SAMD21 MCU Atmel, yang dilengkapi inti ARM Cortex® M0+ 32-bit.

- Salah satu fitur terpentingnya adalah Embedded Debugger (EDBG) Atmel, yang menyediakan antarmuka debug lengkap tanpa memerlukan perangkat keras tambahan, sehingga secara signifikan meningkatkan kemudahan penggunaan untuk debugging perangkat lunak.
- EDBG juga mendukung port COM virtual yang dapat digunakan untuk pemrograman perangkat dan bootloader.



Gambar 3. 2 Papan Arduino Zero

Karakteristik papan Arduino Zero:

Volt operasi	kecepatan CPU	Masuk/keluar analog	IO/PWM Digital	EEPROM [KB]	SRAM [KB]	Kilatan [KB]	USB	UART
3,3 Volt	48Mhz	6/1	14/10	-	32	256	2 mikro	2

Tidak seperti kebanyakan papan Arduino dan Genuino, Zero beroperasi pada 3.3V. Tegangan maksimum yang dapat ditoleransi oleh pin I/O adalah 3,3V. Menerapkan tegangan lebih tinggi dari 3,3V ke pin I/O mana pun dapat merusak board.

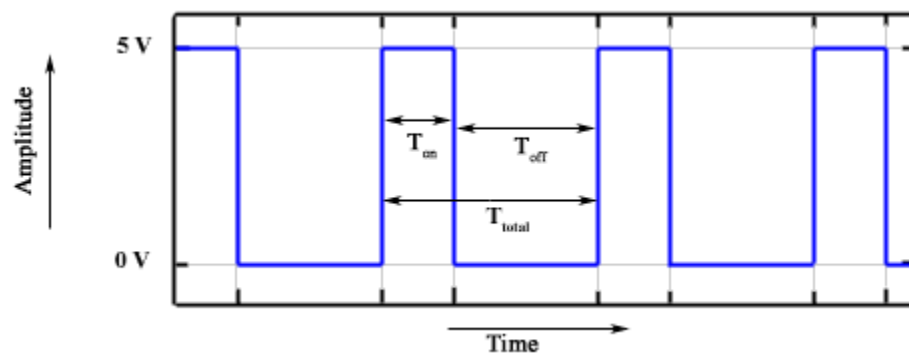
Papan berisi semua yang diperlukan untuk mendukung mikrokontroler. Anda cukup menghubungkannya ke komputer dengan kabel micro-USB atau menyalakannya dengan adaptor AC-DC atau baterai untuk memulai. Zero kompatibel dengan semua pelindung yang bekerja pada 3.3V.

B. Pulse Width Modulation

Modulasi Lebar Pulsa atau PWM adalah teknik umum yang digunakan untuk memvariasikan lebar pulsa dalam rangkaian pulsa. PWM memiliki banyak aplikasi seperti mengendalikan servo dan pengontrol kecepatan, membatasi daya efektif motor dan LED.

a. Prinsip Dasar PWM

Modulasi lebar pulsa pada dasarnya adalah gelombang persegi dengan waktu tinggi dan rendah yang bervariasi. Sinyal PWM dasar ditunjukkan pada gambar berikut.



Gambar 3. 3 Sinyal PWM

Ada berbagai istilah yang terkait dengan PWM:

- **On-Time-** Durasi sinyal waktu tinggi.
- **Off-Time-** Durasi sinyal rendah.
- **Period** - Ini direpresentasikan sebagai jumlah sinyal PWM tepat waktu dan tidak aktif.
- **Duty Cycle-** Ini direpresentasikan sebagai persentase sinyal waktu yang tetap menyala selama periode sinyal PWM.

b. Period

Seperti yang ditunjukkan pada gambar, T_{on} menunjukkan waktu aktif dan T_{off} menunjukkan waktu mati sinyal. Periode adalah jumlah waktu hidup dan mati dan dihitung seperti yang ditunjukkan pada persamaan berikut:

$$T_{total} = T_{hidup} + T_{mati}$$

c. Siklus

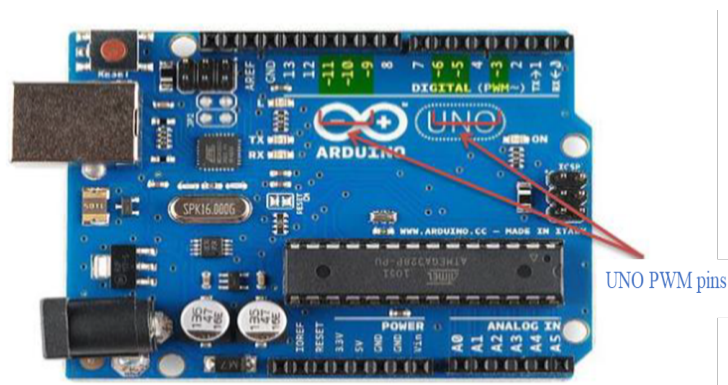
Siklus kerja dihitung sebagai ketepatan waktu dalam jangka waktu tertentu. Dengan menggunakan periode yang dihitung di atas, siklus kerja dihitung sebagai

$$D = \frac{T_{on}}{T_{on} + T_{off}} = \frac{T_{on}}{T_{total}}$$

d. Fungsi analogWrite()

Fungsi **analogWrite()** menulis nilai analog (gelombang PWM) ke pin. Ini dapat digunakan untuk menyalakan LED dengan kecerahan berbeda-beda atau menggerakkan motor dengan berbagai kecepatan. Setelah pemanggilan fungsi **analogWrite()**, pin akan menghasilkan gelombang persegi stabil dari siklus kerja yang ditentukan hingga panggilan berikutnya ke **analogWrite()** atau panggilan ke **digitalRead()** atau **digitalWrite()** pada pin yang sama. Frekuensi sinyal PWM pada sebagian besar pin kira-kira 490 Hz. Pada Uno dan papan serupa, pin 5 dan 6 memiliki frekuensi sekitar 980 Hz. Pin 3 dan 11 pada Leonardo juga beroperasi pada 980 Hz.

Pada sebagian besar papan Arduino (yang memiliki ATmega168 atau ATmega328), fungsi ini berfungsi pada pin 3, 5, 6, 9, 10, dan 11. Pada Arduino Mega, fungsi ini berfungsi pada pin 2 - 13 dan 44 - 46. Arduino lama papan dengan ATmega8 hanya mendukung **analogWrite()** pada pin 9, 10, dan 11.



Gambar 3. 4 Posisi pin untuk PWM

Arduino Due mendukung **analogWrite()** pada pin 2 hingga 13, dan pin DAC0 dan DAC1. Berbeda dengan pin PWM, DAC0 dan DAC1 adalah konverter Digital ke Analog, dan bertindak sebagai keluaran analog sebenarnya.

Anda tidak perlu memanggil `pinMode()` untuk menyetel pin sebagai output sebelum memanggil `analogWrite()`.

Fungsi sintaks `analogWrite()`:

```
analogWrite ( pin , value ) ;
```

value - siklus kerja: antara 0 (selalu mati) dan 255 (selalu aktif).

Contoh:

```
int ledPin = 9; // LED terhubung ke pin digital 9
int analogPin = 3; // potensiometer terhubung ke pin analog 3
int val = 0; // variabel untuk menyimpan nilai baca
void setup() {
    pinMode(ledPin, OUTPUT); // Mengatur pin sebagai output
}
void loop() {
    val = analogRead(analogPin); // Baca Pin Input
    analogWrite(ledPin, (val / 4)); // nilai analogRead berubah dari 0
    // analogTulis nilai dari 0 hingga 255
}
```

C. Random Number

Untuk menghasilkan angka acak, Anda dapat menggunakan fungsi angka acak Arduino. Kami memiliki dua fungsi yaitu `randomSeed(seed)` dan `random()`.

a. `randomSeed(seed)`

Fungsi `randomSeed(seed)` mengatur ulang generator nomor pseudorandom Arduino. Meskipun distribusi angka yang dikembalikan oleh `random()` pada dasarnya acak, urutannya dapat diprediksi. Anda harus mengatur ulang generator ke nilai acak. Jika Anda memiliki pin analog yang tidak terhubung, pin tersebut mungkin menangkap suara acak dari lingkungan sekitar. Ini bisa berupa gelombang radio, sinar kosmik, interferensi elektromagnetik dari ponsel, lampu neon, dan sebagainya.

Contoh:

```
randomSeed(analogRead(5)); // randomize using noise from analog pin 5
```

b. random()

Fungsi acak menghasilkan angka pseudo-acak. Berikut ini adalah sintaksnya.

Pernyataan sintaks random()

```
long random(max) // Ini menghasilkan angka acak dari 0 hingga maks  
long random(min, max) // Ini menghasilkan angka acak dari Min ke Max
```

Contoh:

```
long randNumber;  
void setup() {  
    Serial.begin(9600);  
    // if analog input pin 0 is unconnected, random analog  
    // noise will cause the call to randomSeed() to generate  
    // different seed numbers each time the sketch runs.  
    // randomSeed() will then shuffle the random function.  
    randomSeed(analogRead(0));  
}  
void loop() {  
    // print a random number from 0 to 299  
    Serial.print("random1=");  
    randNumber = random(300);  
    Serial.println(randNumber); // print a random number from 0 to 299  
    Serial.print("random2=");  
    randNumber = random(10, 20); // print a random number from 10 to 19  
    Serial.println (randNumber);  
    delay(50);  
}
```

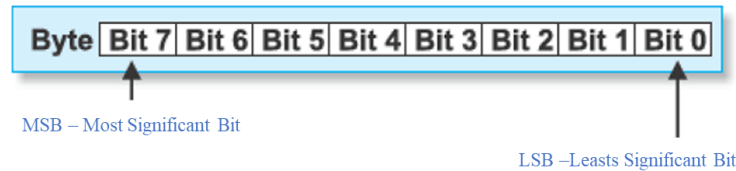
Sekarang mari kita menyegarkan kembali pengetahuan kita tentang beberapa konsep dasar seperti bits dan bytes.

c. Bits

Bit hanyalah angka biner.

- Sistem biner menggunakan dua digit, 0 dan 1.
- Mirip dengan sistem bilangan desimal, di mana digit suatu bilangan tidak mempunyai nilai yang sama, 'signifikansi' suatu bit bergantung pada

posisinya dalam bilangan biner. Misalnya angka desimal 666 sama, tetapi nilainya berbeda.



Gambar 3. 5 Posisi signifikansi bit dalam bilangan biner

d. Byte

Satu byte terdiri dari delapan bit.

- Jika bit adalah sebuah digit, maka logis bahwa byte mewakili angka.
- Semua operasi matematika dapat dilakukan pada mereka.
- Digit dalam satu byte juga tidak memiliki arti yang sama.
- Bit paling kiri mempunyai nilai terbesar yang disebut Most Significant Bit (MSB).
- Bit paling kanan memiliki nilai terkecil dan oleh karena itu disebut Least Significant Bit (LSB).
- Karena delapan angka nol dan satu dari satu byte dapat digabungkan dalam 256 cara berbeda, angka desimal terbesar yang dapat diwakili oleh satu byte adalah 255 (satu kombinasi mewakili angka nol).

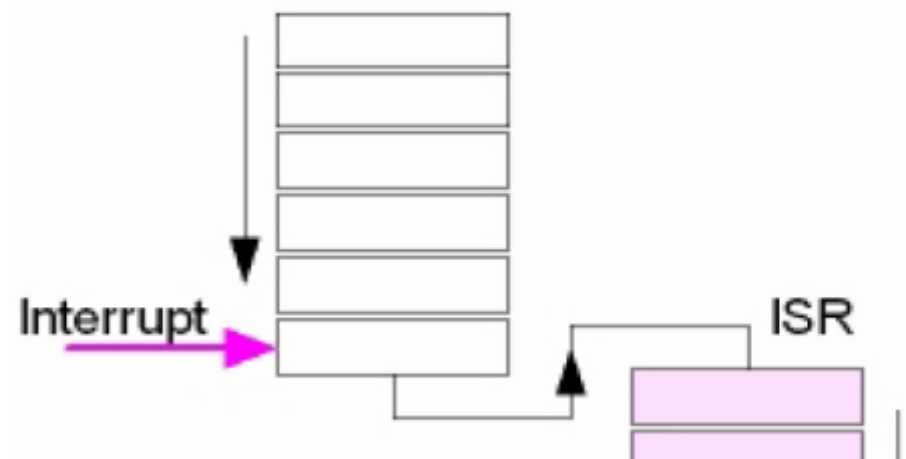
D. Interrupts

Interupsi menghentikan pekerjaan Arduino saat ini sehingga beberapa pekerjaan lain dapat diselesaikan.

Misalkan Anda sedang duduk di rumah, mengobrol dengan seseorang. Tiba-tiba telepon berdering. Anda berhenti mengobrol, dan mengangkat telepon untuk berbicara dengan penelepon. Ketika Anda telah menyelesaikan percakapan lewat telepon, Anda kembali mengobrol dengan orang tersebut sebelum telepon berdering.

Demikian pula, Anda dapat menganggap rutinitas utama sebagai mengobrol dengan seseorang, dering telepon menyebabkan Anda berhenti mengobrol. Rutinitas layanan interupsi adalah proses pembicaraan di telepon. Ketika percakapan telepon berakhir, Anda kemudian kembali ke rutinitas utama Anda mengobrol. Contoh ini menjelaskan dengan tepat bagaimana interupsi menyebabkan prosesor bertindak.

Program utama sedang berjalan dan menjalankan beberapa fungsi dalam suatu rangkaian. Namun, jika terjadi interupsi, program utama akan berhenti sementara rutinitas lainnya dijalankan. Ketika rutinitas ini selesai, prosesor kembali ke rutinitas utama lagi.



Gambar 3. 6 Gambaran intrupsi bekerja

Berikut beberapa fitur penting tentang interupsi:

- Interupsi bisa datang dari berbagai sumber. Dalam hal ini, kami menggunakan interupsi perangkat keras yang dipicu oleh perubahan status pada salah satu pin digital.
- Kebanyakan desain Arduino memiliki dua interupsi perangkat keras (disebut sebagai "interrupt0" dan "interrupt1") yang masing-masing dihubungkan ke pin I/O digital 2 dan 3.
- Arduino Mega memiliki enam interupsi perangkat keras termasuk interupsi tambahan ("interrupt2" hingga "interrupt5") pada pin 21, 20, 19, dan 18.
- Anda dapat mendefinisikan suatu rutin menggunakan fungsi khusus yang disebut sebagai "Interrupt Service Routine" (biasanya dikenal sebagai ISR).
- Anda dapat menentukan rutinitas dan menentukan kondisi pada tepi naik, tepi turun, atau keduanya. Pada kondisi spesifik ini, interupsi akan dilayani.
- Fungsi tersebut dapat dijalankan secara otomatis, setiap kali peristiwa terjadi pada pin input.

Selain itu terdapat dua jenis dalam intrupsi yaitu:

- **Hardware Interrupts**- Terjadi sebagai respons terhadap peristiwa eksternal, seperti pin interupsi eksternal menjadi tinggi atau rendah.
- **Software Interrupts** - Terjadi sebagai respons terhadap instruksi yang dikirim dalam perangkat lunak. Satu-satunya jenis interupsi yang didukung oleh “bahasa Arduino” adalah fungsi `attachInterrupt()`.

Menggunakan intrupsi di Arduino

Interupsi sangat berguna dalam program Arduino karena membantu dalam memecahkan masalah waktu. Penerapan interupsi yang baik adalah membaca encoder putar atau mengamati masukan pengguna. Umumnya, ISR harus dibuat sesingkat dan secepat mungkin. Jika sketsa Anda menggunakan beberapa ISR, hanya satu yang dapat dijalankan dalam satu waktu. Interupsi lain akan dieksekusi setelah interupsi saat ini selesai dalam urutan yang bergantung pada prioritas yang dimilikinya.

Biasanya, variabel global digunakan untuk meneruskan data antara ISR dan program utama. Untuk memastikan variabel yang dibagikan antara ISR dan program utama diperbarui dengan benar, nyatakan variabel tersebut sebagai variabel yang mudah menguap.

Pernyataan sintaks `attachInterrupt`

```
attachInterrupt(digitalPinToInterrupt(pin), ISR, mode);
//Direkomendasikan untuk papan Arduino
attachInterrupt(pin, ISR, mode) ; //direkomendasikan Arduino Due, Zero
only
//Pin argumen: nomor pin
//argumen ISR: ISR untuk memanggil ketika interupsi terjadi;
    //Fungsi ini tidak boleh mengambil parameter dan tidak mengembalikan
    apa pun.
    //Fungsi ini kadang-kadang disebut sebagai rutinitas layanan
    interupsi.
//Mode argumen: menentukan kapan interupsi harus dipicu.
```

Tiga konstanta berikut telah ditentukan sebelumnya sebagai nilai valid

- **LOW** untuk memicu interupsi setiap kali pin rendah.
- **CHANGE** untuk memicu interupsi setiap kali pin berubah nilai.
- **JATUH** setiap kali pin berpindah dari tinggi ke rendah.

Contoh:

```

int pin = 2; //Tentukan pin interupsi ke 2
volatile int state = LOW; // Untuk memastikan variabel dibagi antara ISR
//Program utama diperbarui dengan benar, menyatakannya sebagai volatile.

void setup() {
    pinMode(13, OUTPUT); //Setel pin 13 sebagai output
    attachInterrupt(digitalPinToInterrupt(pin), blink, CHANGE);
    //interupsi pada pin 2 berkedip ISR saat pin untuk mengubah nilai
}

void loop() {
    digitalWrite(13, state); //Pin 13 sama dengan nilai negara
}

void blink() {
    //ISR (Fungsi ISR)
    state = !state; //Alihkan status saat interupsi terjadi
}

```

E. Komunikasi

Ratusan protokol komunikasi telah ditetapkan untuk mencapai pertukaran data ini. Setiap protokol dapat dikategorikan ke dalam salah satu dari dua kategori: paralel atau serial.

a. Komunikasi Paralel

Koneksi paralel antara Arduino dan periferal melalui port input/output adalah solusi ideal untuk jarak pendek hingga beberapa meter. Namun, dalam kasus lain ketika diperlukan untuk menjalin komunikasi antara dua perangkat untuk jarak yang lebih jauh, koneksi paralel tidak dapat digunakan. Antarmuka paralel mentransfer banyak bit secara bersamaan. Mereka biasanya memerlukan bus data - transmisi melalui delapan, enam belas, atau lebih kabel. Data ditransfer dalam gelombang 1 dan 0 yang sangat besar.

Gambar 3. 7 Gambaran komunikasi paralel

b. Kelebihan dan Kekurangan Komunikasi Paralel

Komunikasi paralel tentu mempunyai kelebihan. Ini lebih cepat daripada serial, lugas, dan relatif mudah diimplementasikan. Namun, memerlukan banyak port dan jalur input/output (I/O). Jika Anda pernah harus memindahkan proyek dari Arduino Uno dasar ke Mega, Anda tahu bahwa jalur I/O pada mikroprosesor bisa sangat berharga dan sedikit. Oleh karena itu, kami lebih memilih komunikasi serial, mengorbankan potensi kecepatan untuk pin real estate.

c. Modul Komunikasi Serial

Saat ini, sebagian besar papan Arduino dibuat dengan beberapa sistem berbeda untuk komunikasi serial sebagai perlengkapan standar.

Sistem mana yang digunakan bergantung pada faktor-faktor berikut -

- Berapa banyak perangkat yang dimiliki mikrokontroler untuk bertukar data?
- Seberapa cepat pertukaran data harus dilakukan?
- Berapa jarak antara perangkat ini?
- Apakah perlu mengirim dan menerima data secara bersamaan?

Salah satu hal terpenting mengenai komunikasi serial adalah Protokol, yang harus dipatuhi dengan ketat. Ini adalah seperangkat aturan, yang harus diterapkan sedemikian rupa sehingga perangkat dapat menafsirkan data yang saling dipertukarkan dengan benar. Untungnya, Arduino secara otomatis menangani hal ini, sehingga pekerjaan programmer/pengguna hanya sekedar menulis (data yang akan dikirim) dan membaca (data yang diterima).

d. Jenis Komunikasi Serial

Komunikasi serial selanjutnya dapat diklasifikasikan sebagai -

- **Synchronous** - Perangkat yang disinkronkan menggunakan jam yang sama dan waktunya sinkron satu sama lain.
- **Asynchronous** - Perangkat yang asynchronous memiliki jamnya sendiri dan dipicu oleh output dari keadaan sebelumnya.

Sangat mudah untuk mengetahui apakah suatu perangkat sinkron atau tidak. Jika jam yang sama diberikan ke semua perangkat yang terhubung, maka perangkat tersebut sinkron. Jika tidak ada garis jam, itu tidak sinkron.

Misalnya, modul UART (Universal Asynchronous Receiver Transmitter) bersifat asynchronous.

Protokol serial asinkron memiliki sejumlah aturan bawaan. Aturan-aturan ini hanyalah mekanisme yang membantu memastikan transfer data yang kuat dan bebas kesalahan. Mekanisme ini, yang kita peroleh untuk menghindari sinyal jam eksternal, adalah -

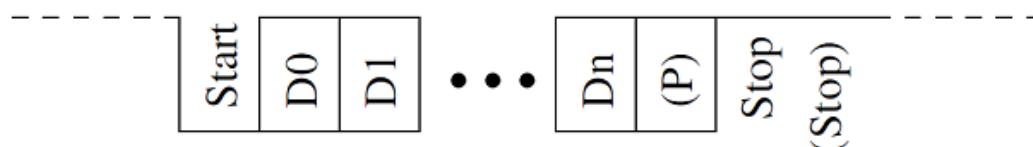
- Bit sinkronisasi
- Bit data
- Bit paritas
- Tingkat baud

e. Bit Sinkronisasi

Bit sinkronisasi adalah dua atau tiga bit khusus yang ditransfer dengan setiap paket data. Mereka adalah bit awal dan bit penghentian. Sesuai dengan namanya, bit-bit ini masing-masing menandai awal dan akhir sebuah paket.

Selalu hanya ada satu bit awal, tetapi jumlah bit penghentian dapat dikonfigurasi menjadi satu atau dua (walaupun biasanya dibiarkan satu).

Bit awal selalu ditandai dengan jalur data idle yang bergerak dari 1 ke 0, sedangkan bit stop akan bertransisi kembali ke keadaan idle dengan menahan garis pada 1.



Gambar 3. 8 Jalur data Bit

f. Bit Data

Jumlah data dalam setiap paket dapat diatur ke berbagai ukuran mulai dari 5 hingga 9 bit. Tentu saja, ukuran data standar adalah byte 8-bit dasar Anda, tetapi ukuran lain juga memiliki kegunaannya. Paket data 7-bit bisa lebih efisien daripada 8, terutama jika Anda hanya mentransfer karakter ASCII 7-bit.

g. Bit Paritas

Pengguna dapat memilih apakah harus ada bit paritas atau tidak, dan jika ya, apakah paritas tersebut harus ganjil atau genap. Bit paritas adalah 0 jika jumlah 1 di antara bit-bit datanya genap. Paritas ganjil justru sebaliknya.

h. Baud rate

Istilah baud rate digunakan untuk menunjukkan jumlah bit yang ditransfer per detik [bps]. Perhatikan bahwa ini mengacu pada bit, bukan byte. Biasanya protokol mengharuskan setiap byte ditransfer bersama dengan beberapa bit kontrol. Artinya satu byte dalam aliran data serial dapat terdiri dari 11 bit. Misalnya, jika baud rate adalah 300 bps maka maksimum 37 dan minimum 27 byte dapat ditransfer per detik.

i. Arduino UART

Kode berikut akan membuat Arduino mengirim hello world saat dijalankan.

```
void setup() {  
    Serial.begin(9600); //Menyiapkan baud rate Serial Library ke 9600  
    Serial.println("hello world"); //Cetak Hello World  
}  
void loop() {  
  
}
```

Setelah sketsa Arduino diunggah ke Arduino, buka Serial monitordi bagian kanan atas Arduino IDE.

Ketik apa pun ke kotak atas Serial Monitor dan tekan kirim atau enter pada keyboard Anda. Ini akan mengirimkan serangkaian byte ke Arduino.

Kode berikut mengembalikan apa pun yang diterimanya sebagai masukan.

Kode berikut akan membuat Arduino mengirimkan output tergantung pada input yang diberikan.

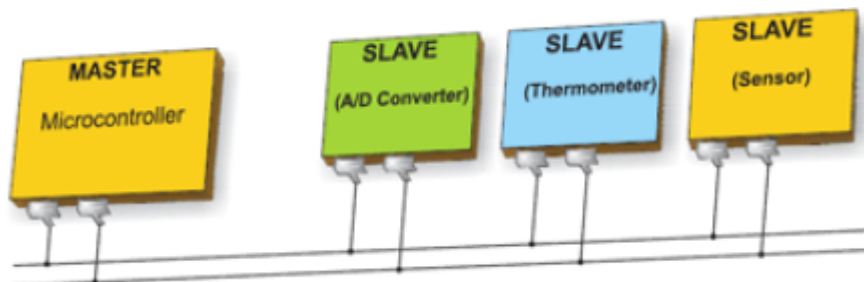
```
void setup() {  
    Serial.begin(9600); //Menyiapkan baud rate Serial Library ke 9600  
}  
void loop() {  
    if(Serial.available()) //jika jumlah byte (karakter) tersedia untuk  
    reading from {  
        serial port  
        Serial.print("I received:"); //cetak yang saya terima  
        Serial.write(Serial.read()); //Kirim apa yang Anda baca  
    }  
}
```

Perhatikan bahwa **Serial.print** dan **Serial.println** akan mengirimkan kembali kode ASCII sebenarnya, sedangkan **Serial.write** akan mengirimkan kembali teks sebenarnya. Lihat kode ASCII untuk informasi lebih lanjut.

F. Inter Integrated Circuit

Sirkuit antar-terintegrasi (I2C) adalah sistem pertukaran data serial antara mikrokontroler dan sirkuit terpadu khusus generasi baru. Ini digunakan ketika jarak antara keduanya pendek (penerima dan pemancar biasanya berada pada papan cetak yang sama). Koneksi dibuat melalui dua konduktor. Satu digunakan untuk transfer data dan yang lainnya digunakan untuk sinkronisasi (sinyal jam).

Seperti terlihat pada gambar berikut, satu perangkat selalu menjadi master. Ia melakukan pengalamatan pada satu chip budak sebelum komunikasi dimulai. Dengan cara ini, satu mikrokontroler dapat berkomunikasi dengan 112 perangkat berbeda. Baud rate biasanya 100 Kb/detik (mode standar) atau 10 Kb/detik (mode baud rate lambat). Sistem dengan baud rate 3,4 Mb/detik baru-baru ini muncul. Jarak antar perangkat yang berkomunikasi melalui bus I2C dibatasi hingga beberapa meter.



Gambar 3. 9 Cara kerja inter integrated circuit

a. Pin Papan I2C

Bus I2C terdiri dari dua sinyal - SCL dan SDA. SCL adalah sinyal clock, dan SDA adalah sinyal data. Master bus saat ini selalu menghasilkan sinyal jam. Beberapa perangkat slave mungkin memaksa jam menjadi rendah pada waktu-waktu tertentu untuk menunda master mengirimkan lebih banyak data (atau memerlukan lebih banyak waktu untuk menyiapkan data sebelum master mencoba untuk mengeluarkannya). Ini dikenal sebagai “peregangan jam”.

Berikut adalah pin untuk papan Arduino yang berbeda

- Uno, Pro Mini A4 (SDA), A5 (SCL)
- Mega, Due 20 (SDA), 21 (SCL)
- Leonardo, Yun 2 (SDA), 3 (SCL)

b. Arduino I2C

Kami memiliki dua mode - kode master dan kode budak - untuk menghubungkan dua papan Arduino menggunakan I2C. dua mode I2C tersebut adalah:

c. Master Transmitter / Slave Receiver

Sekarang mari kita lihat apa itu master transmitter dan slave receiver:

a. Master Transmitter

Fungsi berikut digunakan untuk menginisialisasi perpustakaan Wire dan bergabung dengan bus I2C sebagai master atau slave. Ini biasanya dipanggil hanya sekali.

- **Wire.begin(address)** - Alamat adalah alamat slave 7-bit dalam kasus kami karena master tidak ditentukan dan akan bergabung dengan bus sebagai master.

- **Wire.beginTransmission(address)** - Mulailah transmisi ke perangkat slave I2C dengan alamat yang diberikan.
- **Wire.write(value)** - Mengantri byte untuk transmisi dari perangkat master ke perangkat slave (di antara panggilan ke BeginTransmission() dan endTransmission()).
- **Wire.endTransmission()** - Mengakhiri transmisi ke perangkat budak yang dimulai oleh BeginTransmission() dan mengirimkan byte yang diantrekan oleh wire.write().

Contoh:

```
#include <Wire.h> //Sertakan pustaka kawat
void setup() //Ini hanya akan berjalan sekali
{
    Wire.begin(); // Bergabunglah dengan i2C Bus sebagai Master
}
short age = 0;
void loop() {
    Wire.beginTransmission(2);
    // transmit to device #2
    Wire.write("age is = ");
    Wire.write(age); // mengirim satu byte
    Wire.endTransmission(); // Hentikan transmisi
    delay(1000);
}
```

b. Slave receiver

Pada slave receiver memiliki fungsi yang digunakan sebagai berikut:

- **Wire.begin(address)** - Alamat adalah alamat budak 7-bit.
- **Wire.onReceive(received data handler)** - Fungsi yang dipanggil saat perangkat budak menerima data dari master.
- **Wire.available()** - Mengembalikan jumlah byte yang tersedia untuk diambil dengan Wire.read(). Ini harus dipanggil di dalam handler Wire.onReceive().

Contoh:

```
#include <Wire.h> //Sertakan pustaka kawat
void setup() { //Ini hanya akan berjalan sekali
    Wire.begin(2); // Bergabunglah dengan bus i2C dengan alamat #2
    Wire.onReceive(receiveEvent); // panggilan terimaEvent ketika master
    mengirim sesuatu
}
```

```

    Serial.begin(9600); // Mulai serial untuk output untuk mencetak apa
    yang kita terima
}
void loop() {
    delay(250);
}
//-----Fungsi ini akan dijalankan setiap kali data diterima dari master-
-----//
void receiveEvent(int howMany) {
    while (Wire.available() > 1) // Ulangi semua kecuali yang terakhir
    {
        char c = Wire.read(); // menerima byte sebagai karakter
        Serial.print(c); // print the character
    }
}

```

c. Master receiver / Slave transmitter

Sekarang mari kita lihat apa itu master receiver and slave transmitter:

a. Master receiver

Master, diprogram untuk meminta, dan kemudian membaca byte data yang dikirim dari Slave Arduino yang dialamatkan secara unik.

Fungsi berikut digunakan:

- **Wire.requestFrom(address, number of bytes)** - Digunakan oleh master untuk meminta byte dari perangkat slave. Byte kemudian dapat diambil dengan fungsi fungsi `wire.available()` dan `wire.read()`.

Contoh:

```

#include <Wire.h> //memasukkan wire library void setup()
{
    Wire.begin(); // Bergabunglah dengan bus i2C (alamat opsional untuk
    Master)
    Serial.begin(9600); // Mulai serial untuk output
}

void loop() {
    Wire.requestFrom(2, 1); // Minta 1 byte dari perangkat slave #2
    while (Wire.available()) // slave dapat mengirim kurang dari yang
    diminta
    {

```

```

        char c = Wire.read(); // menerima byte sebagai karakter
        Serial.print(c); // Mencetak karakter
    }
    delay(500);
}

```

b. Slave transmitter

Pada slave transmitter memiliki fungsi yang digunakan sebagai berikut:

- **Wire.onRequest(handler)** - Sebuah fungsi dipanggil ketika master meminta data dari perangkat budak ini.

Contoh:

```

#include <Wire.h>

void setup() {
    Wire.begin(2); // Bergabunglah dengan bus i2C dengan alamat #2
    Wire.onRequest(requestEvent); // Daftarkan event
}

Byte x = 0;

void loop() {
    delay(100);
}

// fungsi yang dijalankan setiap kali data diminta oleh master
// Fungsi ini terdaftar sebagai event, see setup()

void requestEvent() {
    Wire.write(x); // Tanggapi dengan pesan 1 byte seperti yang diharapkan
    oleh master
    x++;
}

```

F. Serial Peripheral Interface

Bus Serial Peripheral Interface (SPI) adalah sistem komunikasi serial, yang menggunakan hingga empat konduktor, biasanya tiga. Satu konduktor digunakan untuk menerima data, satu untuk mengirim data, satu untuk sinkronisasi dan satu lagi sebagai alternatif untuk memilih perangkat yang akan diajak berkomunikasi.

Ini adalah koneksi dupleks penuh, artinya data dikirim dan diterima secara bersamaan. Baud rate maksimum lebih tinggi daripada sistem komunikasi I2C.

a. Papan Pin SPI

SPI menggunakan empat kabel berikut:

- **SCK** - Ini adalah jam serial yang digerakkan oleh master.
- **MOSI** - Ini adalah keluaran master / masukan budak yang digerakkan oleh master.
- **MISO** - Ini adalah input master / output slave yang digerakkan oleh master.
- **SS** - Ini adalah kabel pemilihan budak.

Fungsi-fungsi berikut digunakan. Anda harus menyertakan SPI.h.

- **SPI.begin()** - Menginisialisasi bus SPI dengan mengatur SCK, MOSI, dan SS ke output, menarik SCK dan MOSI rendah, dan SS tinggi.
- **SPI.setClockDivider(divider)** - Untuk mengatur pembagi jam SPI relatif terhadap jam sistem. Pada papan berbasis AVR, pembagi yang tersedia adalah 2, 4, 8, 16, 32, 64 atau 128. Pengaturan default adalah SPI_CLOCK_DIV4, yang mengatur jam SPI ke seperempat frekuensi jam sistem (5 Mhz untuk papan pada 20 MHz).
- **Divider**- Bisa jadi (SPI_CLOCK_DIV2, SPI_CLOCK_DIV4, SPI_CLOCK_DIV8, SPI_CLOCK_DIV16, SPI_CLOCK_DIV32, SPI_CLOCK_DIV64, SPI_CLOCK_DIV128).
- **SPI.transfer(val)** - Transfer SPI didasarkan pada pengiriman dan penerimaan secara bersamaan: data yang diterima dikembalikan dalam receiverVal.
- **SPI.beginTransaction(SPISettings(speedMaximum, dataOrder, dataMode))** – speedMaximum adalah jam, dataOrder(MSBFIRST atau LSBFIRST), dataMode(SPI_MODE0, SPI_MODE1, SPI_MODE2, atau SPI_MODE3).

Kami memiliki empat mode operasi di SPI sebagai berikut:

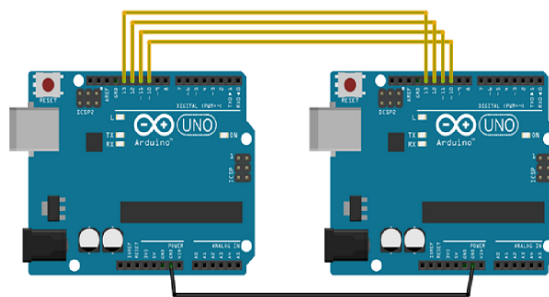
- **Mode 0 (default)** - Jam biasanya rendah (CPOL = 0), dan data diambil sampelnya pada transisi dari rendah ke tinggi (leading edge) (CPHA = 0).
- **Mode 1** - Jam biasanya rendah (CPOL = 0), dan data diambil sampelnya pada transisi dari tinggi ke rendah (tepi belakang) (CPHA = 1).
- **Mode 2** - Jam biasanya tinggi (CPOL = 1), dan data diambil sampelnya pada transisi dari tinggi ke rendah (leading edge) (CPHA = 0).
- **Mode 3** - Jam biasanya tinggi (CPOL = 1), dan data diambil sampelnya pada transisi dari rendah ke tinggi (tepi belakang) (CPHA = 1).

- **SPI.attachInterrupt(handler)** - Fungsi yang dipanggil saat perangkat budak menerima data dari master.

Sekarang, kita akan menghubungkan dua papan Arduino UNO bersama-sama; yang satu sebagai tuan dan yang lainnya sebagai budak.

- (SS): pin 10
- (MOSI): pin 11
- (MISO): pin 12
- (SCK) : pin 13

Ground-nya umum. Berikut adalah representasi diagramatik dari koneksi antara kedua papan arduino.



Gambar 3. 10 Koneksi antar arduino

Mari kita lihat contoh SPI sebagai Master dan SPI sebagai Slave.

b. SPI sebagai MASTER

Contoh:

```
#include <SPI.h>

void setup (void) {
    Serial.begin(115200); //Tetapkan baud rate ke 115200 untuk USART
    digitalWrite(SS, HIGH); // nonaktifkan Pilih Slave
    SPI.begin ();
    SPI.setClockDivider(SPI_CLOCK_DIV8); //Bagilah jam dengan 8
}

void loop (void) {
    char c;
    digitalWrite(SS, LOW); // aktifkan Slave Select
    // send test string
    for (const char * p = "Hello, world!\r" ; c = *p; p++) {
        SPI.transfer (c);
        Serial.print(c);
    }
}
```

```

    }
    digitalWrite(SS, HIGH); // nonaktifkan Pilih Slave
    delay(2000);
}

```

c. SPI sebagai Slave

Contoh:

```

#include <SPI.h>
char buff [50];
volatile byte indx;
volatile boolean process;

void setup (void) {
    Serial.begin (115200);
    pinMode(MISO, OUTPUT); // have to send on master in so it set as
    output
    SPCR |= _BV(SPE); // mengaktifkan SPI dalam mode slave
    indx = 0; // buffer kosong
    process = false;
    SPI.attachInterrupt(); // Mengaktifkan interupsi
}
ISR (SPI_STC_vect) // SPI interupsi rutin
{
    byte c = SPDR; // baca byte dari SPI Data Register
    if (indx < sizeof buff) {
        buff [indx++] = c; // Simpan data di indeks berikutnya di buff
        array
        if (c == '\r') //Periksa akhir kata
            process = true;
    }
}

void loop (void) {
    if (process) {
        process = false; //Setel ulang proses
        Serial.println (buff); //Cetak array pada monitor serial
        indx= 0; //Tombol reset ke nol
    }
}

```

BAB 4

PROYEK ARDUINO

A. LED Berkedip

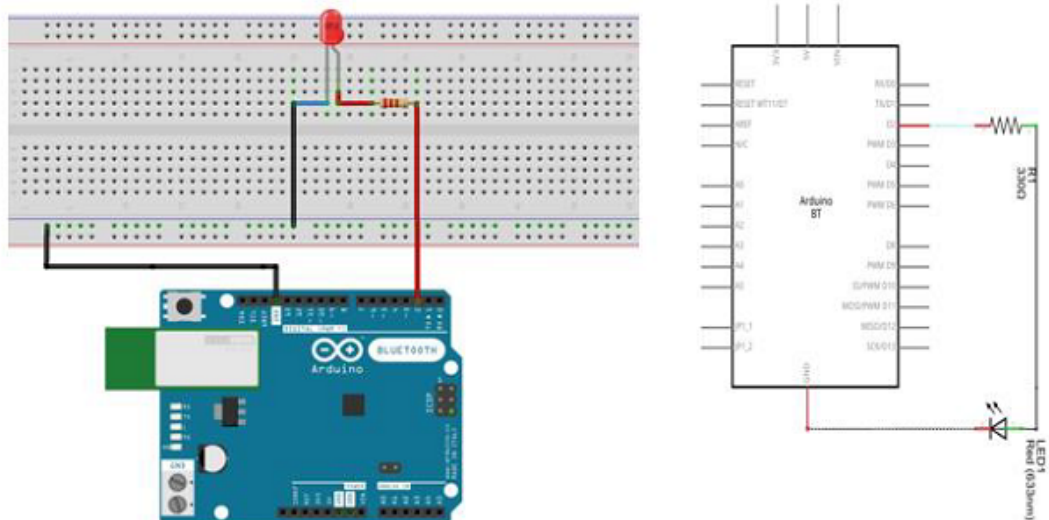
LED adalah lampu kecil dan kuat yang digunakan dalam berbagai aplikasi. Untuk memulai, kita akan mengedipkan LED, mikrokontroler Hello World. Ini sederhana menyalakan dan mematikan lampu. Menetapkan dasar penting ini akan memberi Anda dasar yang kuat saat kami berupaya menuju eksperimen yang lebih kompleks.

Komponen yang diperlukan:

- 1 × *Breadboard*
- 1×Arduino Uno R3
- 1×LED
- Resistor 1 × 330Ω
- 2 × Kabel Jumper

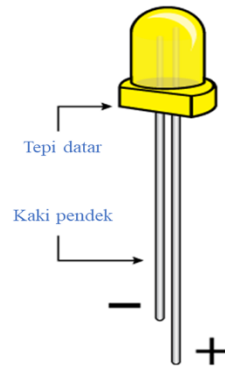
Prosedur:

Ikuti diagram sirkuit dan sambungkan komponen pada *Breadboard* seperti yang ditunjukkan pada gambar di bawah.



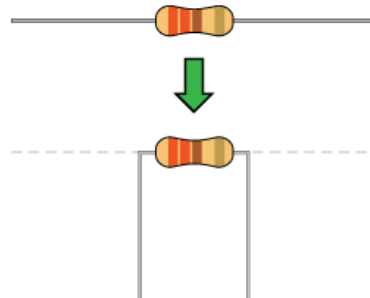
Gambar 4. 1 Rangkaian LED berkedip

Catatan – Untuk mengetahui polaritas LED, perhatikan dengan saksama. Kaki yang lebih pendek dari keduanya, ke arah tepi datar bohlam menunjukkan terminal negatif.



Gambar 4. 2 Pin pada LED

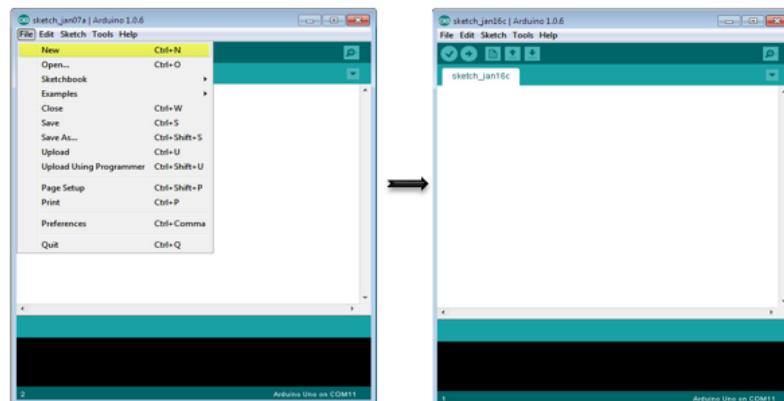
Komponen seperti resistor harus memiliki terminal yang ditekuk hingga sudut 90° agar dapat memasang soket *Breadboard* dengan benar. Anda juga dapat memperpendek terminalnya.



Gambar 4. 3 Resistor yang ditekuk

Sketch:

Buka perangkat lunak Arduino IDE di komputer Anda. Pengkodean dalam bahasa Arduino akan mengontrol sirkuit Anda. Buka File sketsa baru dengan mengklik Baru.



Gambar 4. 4 Membuka *new sketch* di Arduino IDE

Program Arduino:

```

/* Blink. Menyalakan LED selama satu detik, lalu mati selama satu detik,
berulang kali.*/
// Fungsi pengaturan berjalan sekali ketika Anda menekan reset atau
menyalakan papan
void setup() { // Inisialisasi PIN Digital 13 sebagai output.
    pinMode(2, OUTPUT);
}
// Fungsi loop berjalan berulang-ulang selamanya
void loop() {
    digitalWrite(2, HIGH); // nyalakan LED (TINGGI adalah level tegangan)
    delay(1000); // tunggu sebentar
    digitalWrite(2, LOW); // matikan LED dengan membuat tegangan RENDAH
    delay(1000); // tunggu sebentar
}

```

pinMode(2, OUTPUT) - Sebelum Anda dapat menggunakan salah satu pin Arduino, Anda perlu memberi tahu Arduino Uno R3 apakah itu INPUT atau OUTPUT. Kami menggunakan “fungsi” bawaan yang disebut pinMode() untuk melakukan ini.

digitalWrite(2, HIGH) - Saat Anda menggunakan pin sebagai OUTPUT, Anda dapat memerintahkannya menjadi HIGH (output 5 volt), atau LOW (output 0 volt).

Hasil:

Anda akan melihat LED Anda menyala dan mati. Jika output yang diperlukan tidak terlihat, pastikan Anda telah merakit sirkuit dengan benar, dan memverifikasi serta mengunggah kode ke board Anda.

B. LED Redup

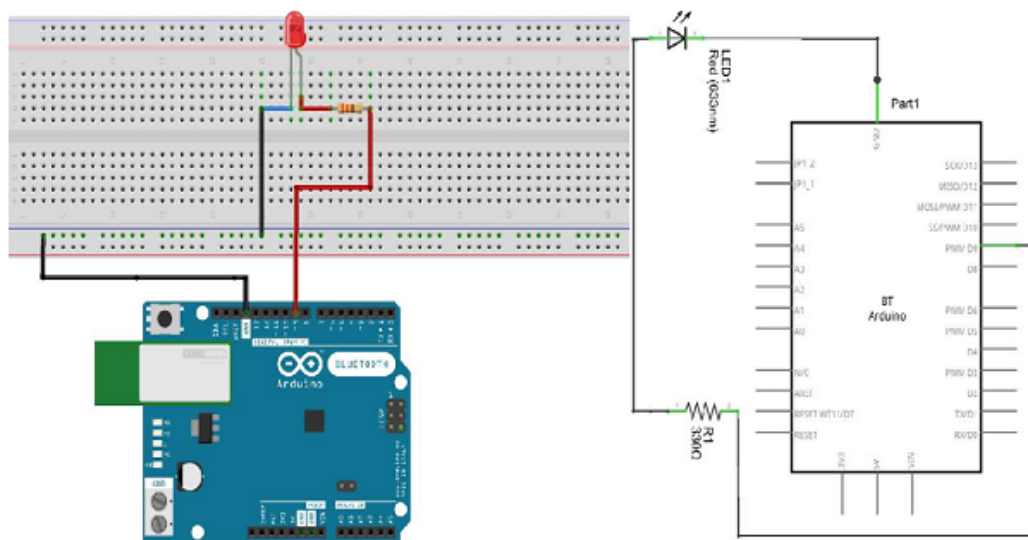
Contoh ini mendemonstrasikan penggunaan fungsi `analogWrite()` dalam mematikan LED. `AnalogWrite` menggunakan modulasi lebar pulsa (PWM), menghidupkan dan mematikan pin digital dengan sangat cepat dengan rasio berbeda antara hidup dan mati, untuk menciptakan efek memudar.

Komponen yang Diperlukan:

- 1 × Breadboard
- 1×Arduino Uno R3
- 1×LED
- Resistor 1 × 330Ω
- 2 × Kabel Jumper

Prosedur:

Ikuti diagram sirkuit dan sambungkan komponen pada *Breadboard* seperti yang ditunjukkan pada gambar di bawah.



Gambar 4. 5 Rangkaian LED Redup

Catatan- Untuk mengetahui polaritas sebuah LED, perhatikan baik-baik. Kaki yang lebih pendek, mengarah ke tepi datar bohlam menunjukkan terminal negatif. Seperti pada Gambar 4.2.

Komponen seperti resistor harus memiliki terminal yang ditebuk hingga sudut 90° agar dapat memasang soket *Breadboard* dengan benar. Anda juga dapat memperpendek terminalnya. Hal tersebut terlihat seperti pada Gambar 4.3.

Sketch:

Buka perangkat lunak Arduino IDE di komputer Anda. Pengkodean dalam bahasa Arduino akan mengontrol sirkuit Anda. Buka File sketsa baru dengan mengklik Baru seperti pada Gambar 4.4.

Program Arduino:

```
/* Fade
   Contoh ini menunjukkan cara memudarkan LED pada pin 9 menggunakan
   fungsi analogWrite().
   Fungsi analogWrite() menggunakan PWM, jadi jika Anda ingin mengubah
   pin yang Anda gunakan, jadilah
   yakin untuk menggunakan pin berkemampuan PWM lainnya. Pada kebanyakan
   Arduino, pin PWM diidentifikasi dengan
   a "~" sign, like ~3, ~5, ~6, ~9, ~10 and ~11.
*/
int led = 9; // pin PWM yang terpasang pada LED
int brightness = 0; // seberapa terang LED-nya
int fadeAmount = 5; // berapa banyak poin untuk memudarkan LED oleh
// Rutinitas penyiapan berjalan sekali saat Anda menekan Reset:
void setup() {
  // Nyatakan pin 9 sebagai output:
  pinMode(led, OUTPUT);
}
// Rutinitas loop berjalan berulang-ulang selamanya:
void loop() {
  // Atur kecerahan Pin 9:
  analogWrite(led, brightness);
  // Ubah kecerahan untuk waktu berikutnya melalui loop:
  brightness = brightness + fadeAmount;
  // Balikkan arah memudar di ujung pudar:
  if (brightness == 0 || brightness == 255) {
    fadeAmount = -fadeAmount ;
  }
  // Tunggu selama 30 milidetik untuk melihat efek peredupan
  delay(300);
}
```

Setelah mendeklarasikan pin 9 sebagai pin LED Anda, tidak ada yang bisa dilakukan dalam fungsi `setup()` kode Anda. Fungsi `analogWrite()` yang akan Anda gunakan dalam loop utama kode Anda memerlukan dua argumen: Satu, memberi tahu fungsi ke pin mana untuk menulis dan yang lainnya menunjukkan nilai PWM yang akan ditulis.

Untuk memudahkan LED mati dan hidup, naikkan nilai PWM secara bertahap dari 0 (mati sepenuhnya) menjadi 255 (hidup penuh), lalu kembali ke 0, untuk menyelesaikan siklus. Pada sketsa di atas, nilai PWM diatur menggunakan variabel yang disebut kecerahan. Setiap kali melewati loop, nilai tersebut bertambah sebesar nilai variabel **fadeAmount**.

Jika kecerahan berada pada nilai ekstremnya (0 atau 255), maka `fadeAmount` diubah menjadi negatif. Dengan kata lain, jika `fadeAmount` adalah 5, maka disetel ke -5. Jika -5, maka disetel ke 5. Pada loop berikutnya, perubahan ini menyebabkan kecerahan berubah arah juga.

analogWrite() dapat mengubah nilai PWM dengan sangat cepat, sehingga penundaan di akhir sketsa mengontrol kecepatan fade. Coba ubah nilai penundaannya dan lihat bagaimana perubahannya pada efek pemudaran.

Hasil:

Anda akan melihat kecerahan LED Anda berubah secara bertahap.

C. Membaca Tegangan Analog

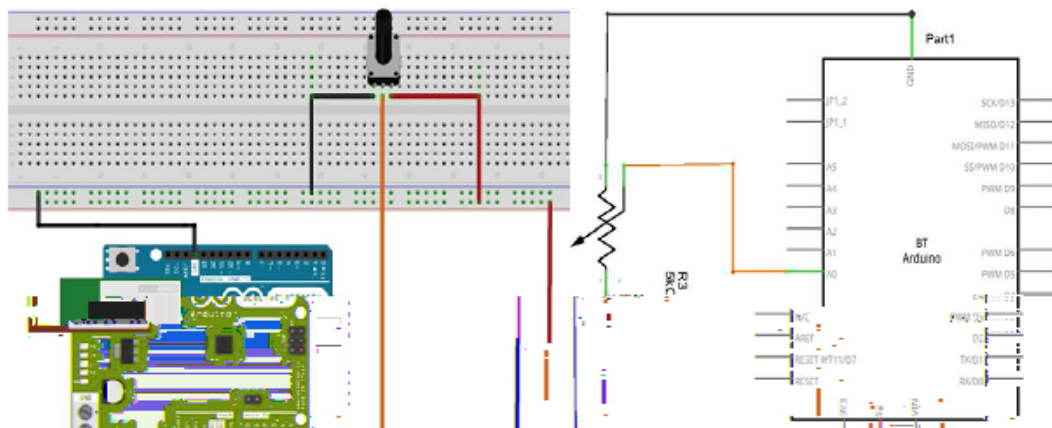
Contoh ini akan menunjukkan cara membaca input analog pada pin analog 0. Input diubah dari `analogRead()` menjadi tegangan, dan dicetak ke monitor serial Perangkat Lunak Arduino (IDE).

Komponen yang Diperlukan:

- 1 × *Breadboard*
- 1×Arduino Uno R3
- Resistor variabel 1 × 5K (potensiometer)
- 2 × Kabel Jumper

Prosedur:

Ikuti diagram rangkaian dan hubungkan komponen-komponen pada *Breadboard* seperti yang ditunjukkan pada gambar di bawah ini.

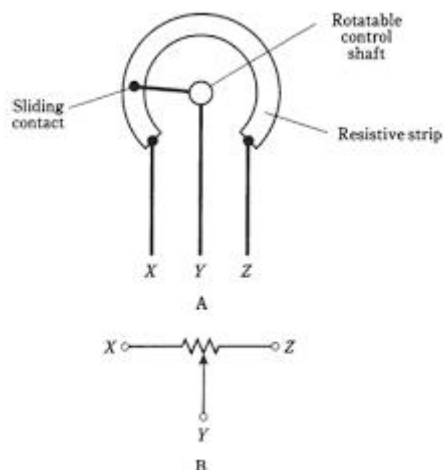


Gambar 4. 6 Rangkaian membaca tegangan analog dari potensiometer

Potensiometer

Potensiometer (atau pot) adalah transduser elektro-mekanis sederhana. Ini mengubah gerakan putar atau linier dari operator input menjadi perubahan resistansi. Perubahan ini (atau dapat) digunakan untuk mengontrol apa pun mulai dari volume sistem hi-fi hingga arah kapal kontainer besar.

Panci seperti yang kita kenal pada awalnya dikenal sebagai rheostat (pada dasarnya adalah resistor wirewound variabel). Variasi pot yang tersedia saat ini cukup mencengangkan, dan akan sangat sulit bagi pemula (khususnya) untuk menentukan jenis pot mana yang cocok untuk tugas tertentu. Beberapa jenis pot berbeda, yang semuanya dapat digunakan untuk tugas yang sama membuat pekerjaan menjadi lebih sulit.



Gambar 4. 7 Skema standar potensiometer dan potensiometer

Sketsa:

Buka perangkat lunak Arduino IDE di komputer Anda. Pengkodean dalam bahasa Arduino akan mengontrol sirkuit Anda. Buka File sketsa baru dengan mengklik Baru. Seperti terlihat pada Gambar 4.4.

Program Arduino:

```
/* BacaAnalogTegangan. Membaca input analog pada pin 0, mengubahnya
menjadi tegangan, dan mencetak hasilnya ke monitor serial.

Representasi grafis tersedia menggunakan plotter serial (Alat > menu
Plotter Serial)

Pasang pin tengah potensiometer ke pin A0, dan pin luar ke +5V dan
ground.*/
// Rutinitas penyiapan berjalan sekali saat Anda menekan Reset:
void setup() {
    // Inisialisasi komunikasi serial pada 9600 bit per detik:
    Serial.begin(9600);
}
// Rutinitas loop berjalan berulang-ulang selamanya:
void loop() {
    // Baca input pada pin analog 0:
    int sensorValue = analogRead(A0);
    // Ubah pembacaan analog (yang berlangsung dari 0 - 1023) menjadi
    tegangan (0 - 5V):
    float voltage = sensorValue * (5.0 / 1023.0);
    // Cetak nilai yang Anda baca:
    Serial.println(voltage);
}
```

Dalam program atau sketsa yang diberikan di bawah ini, hal pertama yang Anda lakukan dalam fungsi pengaturan adalah memulai komunikasi serial, pada 9600 bit per detik, antara board Anda dan komputer Anda dengan saluran:

```
Serial.begin(9600);
```

Di loop utama kode Anda, Anda perlu membuat variabel untuk menyimpan nilai resistansi (antara 0 dan 1023, cocok untuk tipe data int) yang berasal dari potensiometer Anda:

```
int sensorValue = analogRead(A0);
```

Untuk mengubah nilai dari 0-1023 ke kisaran yang sesuai dengan tegangan yang dibaca pin, Anda perlu membuat variabel lain, float, dan melakukan sedikit

perhitungan. Untuk menskalakan angka antara 0,0 dan 5,0, bagi 5,0 dengan 1023,0 dan kalikan dengan sensorValue:

```
float voltage= sensorValue * (5.0 / 1023.0);
```

Terakhir, Anda perlu mencetak informasi ini ke jendela serial Anda. Anda dapat melakukan ini dengan perintah `Serial.println()` di baris kode terakhir Anda:

```
Serial.println(voltage);
```

Sekarang, buka Serial Monitor di Arduino IDE dengan mengklik ikon di sisi kanan bilah hijau atas atau menekan `Ctrl+Shift+M`.

Gambar 4. 8 Menunjukkan letak icon serial monitor

Hasil:

Anda akan melihat aliran angka yang berkisar antara 0,0 - 5,0. Saat Anda memutar pot, nilainya akan berubah, sesuai dengan tegangan pada pin A0.

D. Grafik Batang LED

Contoh ini menunjukkan cara membaca input analog pada pin analog 0, mengubah nilai dari `analogRead()` menjadi tegangan, dan mencetaknya ke monitor serial Perangkat Lunak Arduino (IDE).

Komponen yang Diperlukan:

- 1 × *Breadboard*
- 1×Arduino Uno R3
- Resistor variabel 1 × 5k ohm (potensiometer)
- 2 × Kabel Jumper
- 8 × LED atau Anda dapat menggunakan (tampilan grafik batang LED seperti yang ditunjukkan pada gambar di bawah)

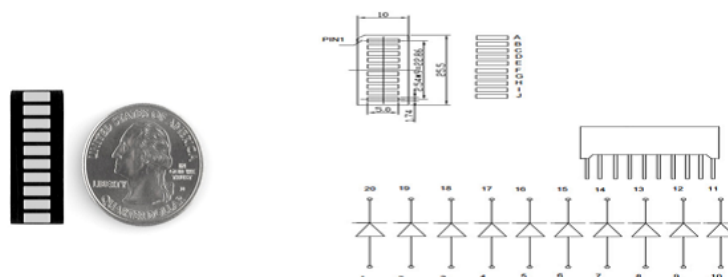
Prosedur:

Ikuti diagram sirkuit dan sambungkan komponen pada *Breadboard* seperti yang ditunjukkan pada gambar di bawah.

Gambar 4. 9 Rangkaian grafik batang LED

Sketsa:

Buka perangkat lunak Arduino IDE di komputer Anda. Pengkodean dalam bahasa Arduino akan mengontrol sirkuit Anda. Buka File sketsa baru dengan mengklik Baru. Seperti terlihat pada Gambar 4.4.



Gambar 4. 10 Grafik batang LED 10 segment

LED grafik batang 10 segmen ini memiliki banyak kegunaan. Dengan tapak yang ringkas, pemasangan yang sederhana, mudah untuk membuat prototipe atau produk jadi. Pada dasarnya, mereka adalah 10 LED biru individual yang ditempatkan bersama, masing-masing dengan koneksi anoda dan katoda individual.

Mereka juga tersedia dalam warna kuning, merah, dan hijau.

Catatan- Pin out pada grafik batang ini mungkin berbeda dari yang tercantum di lembar data. Memutar perangkat 180 derajat akan memperbaiki perubahan tersebut, menjadikan pin 11 sebagai pin pertama dalam baris.

Program Arduino:

```
/*
```

```

    Grafik batang LED
    Menyalakan serangkaian LED berdasarkan nilai sensor analog.
    Ini adalah cara sederhana untuk membuat tampilan grafik batang.
    Meskipun grafik ini menggunakan 8LED, Anda dapat menggunakan angka
    apa pun dengan
        mengubah jumlah LED dan pin dalam array.
    Metode ini dapat digunakan untuk mengontrol serangkaian digital
        output yang bergantung pada input analog.
*/
// Konstanta ini tidak akan berubah:
const int analogPin = A0; // pin tempat potensiometer terpasang
const int ledCount = 8; // jumlah LED dalam grafik batang
int ledPins[] = {2, 3, 4, 5, 6, 7, 8, 9}; // array nomor pin tempat LED
terpasang
void setup() {
    // Ulangi array pin dan atur semuanya ke output:
    for (int thisLed = 0; thisLed < ledCount; thisLed++) {
        pinMode(ledPins[thisLed], OUTPUT);
    }
}
void loop() {
    // Baca potensiometer:
    int sensorReading = analogRead(analogPin);
    // memetakan hasilnya ke rentang dari 0 hingga jumlah LED:
    int ledLevel = map(sensorReading, 0, 1023, 0, ledCount);
    // loop di atas array LED:
    for (int thisLed = 0; thisLed < ledCount; thisLed++) {
        // jika indeks elemen array kurang dari ledLevel,
        // Aktifkan Pin untuk elemen ini:
        if (thisLed < ledLevel) {
            digitalWrite(ledPins[thisLed], HIGH);
        } else { // matikan semua pin yang lebih tinggi dari ledLevel:
            digitalWrite(ledPins[thisLed], LOW);
        }
    }
}
}

```

Sketsa tersebut bekerja seperti ini: pertama, Anda membaca input. Anda memetakan nilai input ke rentang output, dalam hal ini sepuluh LED. Kemudian Anda menyiapkan for **-loop** untuk mengulangi output. Jika angka output dalam

rangkaian lebih rendah dari rentang input yang dipetakan, Anda menyalakannya. Jika tidak, Anda mematikannya.

Hasil:

Anda akan melihat LED menyala satu per satu saat nilai pembacaan analog bertambah dan mati satu per satu saat nilai pembacaan menurun.

E. Keyboard logout

Contoh ini menggunakan perpustakaan Keyboard untuk mengeluarkan Anda dari sesi pengguna di komputer Anda ketika pin 2 pada ARDUINO UNO ditarik ke ground. Sketsa tersebut mensimulasikan penekanan tombol secara berurutan dari dua atau tiga tombol pada saat yang sama dan setelah beberapa saat, ia melepaskannya.

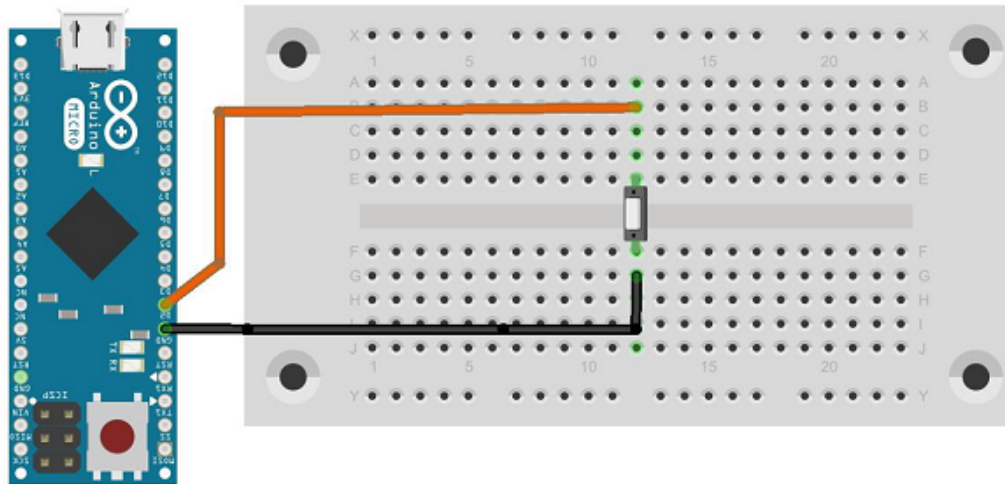
Warning- Saat Anda menggunakan perintah **Keyboard.print()** , Arduino mengambil alih keyboard komputer Anda. Untuk memastikan Anda tidak kehilangan kendali atas komputer Anda saat menjalankan sketsa dengan fungsi ini, siapkan sistem kontrol yang andal sebelum Anda memanggil **Keyboard.print()**. Sketsa ini dirancang untuk hanya mengirimkan perintah Keyboard setelah pin ditarik ke ground.

Komponen yang Diperlukan:

- 1 × *Breadboard*
- 1 × papan Arduino Leonardo, Micro, atau Due
- 1 × tombol tekan
- 1 × Kabel Jumper

Prosedur:

Ikuti diagram rangkaian dan hubungkan komponen-komponen pada *Breadboard* seperti yang ditunjukkan pada gambar di bawah ini.

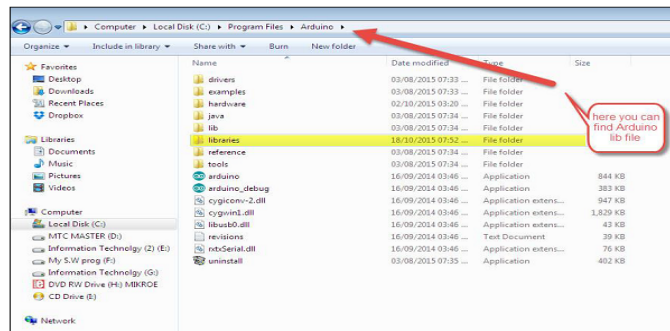


Gambar 4. 11 Rangkaian kerboard logout

Sketsa:

Buka perangkat lunak Arduino IDE di komputer Anda. Pengodean dalam bahasa Arduino akan mengendalikan sirkuit Anda. Buka File sketsa baru dengan mengeklik Baru. Seperti terlihat pada Gambar 4.4. Untuk contoh ini, Anda perlu menggunakan Arduino IDE 1.6.7

Catatan- Anda harus menyertakan perpustakaan keyboard di file perpustakaan Arduino Anda. Salin dan tempel file perpustakaan keypad di dalam file dengan nama perpustakaan (disorot) seperti yang ditunjukkan pada gambar layar berikut.



Gambar 4. 12 Letak folder *library* untuk menambahkan ke Arduino IDE

Program Arduino:

```
/*
Keluar dari keyboard
Sketsa ini menunjukkan perpustakaan Keyboard.
Saat Anda menghubungkan pin 2 ke ground, ia melakukan logout.
Ini menggunakan kombinasi keyboard untuk melakukan ini, sebagai
berikut:
Pada Windows, CTRL-ALT-DEL diikuti oleh ALT-1
```

```

    Di Ubuntu, CTRL-ALT-DEL, dan ENTER
    Pada OSX, CMD-SHIFT-q
    Untuk membangunkan: Spasi.
    Sirkuit:
    * Arduino Leonardo atau Micro
    * kawat untuk menghubungkan D2 ke ground.
*/

#define OSX 0
#define WINDOWS 1
#define UBUNTU 2

#include "Keyboard.h"

// Ubah ini agar sesuai dengan platform Anda:
int platform = WINDOWS;

void setup() {
    // Jadikan PIN 2 sebagai input dan nyalakan
    // resistor pullup sehingga menjadi tinggi kecuali
    // terhubung ke ground:

    pinMode(2, INPUT_PULLUP);
    Keyboard.begin();
}

void loop() {
    while (digitalRead(2) == HIGH) {
        // Tidak melakukan apa pun sampai pin 2 menjadi rendah
        delay(500);
    }

    delay(1000);

    switch (platform) {
        case OSX:
            Keyboard.press(KEY_LEFT_GUI);

            // Shift-Q logs out:
            Keyboard.press(KEY_LEFT_SHIFT);

```

```

Keyboard.press('Q');
delay(100);

// enter:
Keyboard.write(KEY_RETURN);
break;

case WINDOWS:
// CTRL-ALT-DEL:
Keyboard.press(KEY_LEFT_CTRL);
Keyboard.press(KEY_LEFT_ALT);
Keyboard.press(KEY_DELETE);
delay(100);
Keyboard.releaseAll();

//ALT-l:
delay(2000);
Keyboard.press(KEY_LEFT_ALT);
Keyboard.press('l');
Keyboard.releaseAll();
break;

case UBUNTU:
// CTRL-ALT-DEL:
Keyboard.press(KEY_LEFT_CTRL);
Keyboard.press(KEY_LEFT_ALT);
Keyboard.press(KEY_DELETE);
delay(1000);
Keyboard.releaseAll();
// Enter untuk mengonfirmasi logout:
Keyboard.write(KEY_RETURN);
break;
}
// Tidak melakukan apa-apa:
while (true);
}

Keyboard.releaseAll();

// enter:

```

```

        Keyboard.write(KEY_RETURN);
        break;
        case WINDOWS:

// CTRL-ALT-DEL:
        Keyboard.press(KEY_LEFT_CTRL);
        Keyboard.press(KEY_LEFT_ALT);
        Keyboard.press(KEY_DELETE);
        delay(100);
        Keyboard.releaseAll();

//ALT-l:
        delay(2000);
        Keyboard.press(KEY_LEFT_ALT);
        Keyboard.press('l');
        Keyboard.releaseAll();
        break;

        case UBUNTU:
            // CTRL-ALT-DEL:
            Keyboard.press(KEY_LEFT_CTRL);
            Keyboard.press(KEY_LEFT_ALT);
            Keyboard.press(KEY_DELETE);
            delay(1000);
            Keyboard.releaseAll();
            // Enter to confirm logout:
            Keyboard.write(KEY_RETURN);
            break;
    }
    // Tidak melakukan apa-apa:
    while (true);
}

```

Sebelum Anda mengunggah program ke board Anda, pastikan Anda menetapkan OS yang benar yang sedang Anda gunakan ke variabel platform.

Saat sketsa sedang berjalan, menekan tombol akan menghubungkan pin 2 ke ground dan papan akan mengirimkan urutan logout ke PC yang terhubung USB.

Hasil:

Saat Anda menghubungkan pin 2 ke ground, ia melakukan operasi logout.

Ia menggunakan kombinasi keyboard berikut untuk keluar

- Di **Windows** , CTRL-ALT-DEL diikuti oleh ALT-l
- Di **Ubuntu** , CTRL-ALT-DEL, dan ENTER
- Di **OSX** , CMD-SHIFT-q

F. Keyboard Message

Dalam contoh ini, ketika tombol ditekan, string teks dikirim ke komputer sebagai input keyboard. String tersebut melaporkan berapa kali tombol ditekan. Setelah Leonardo diprogram dan dihubungkan, buka editor teks favorit Anda untuk melihat hasilnya.

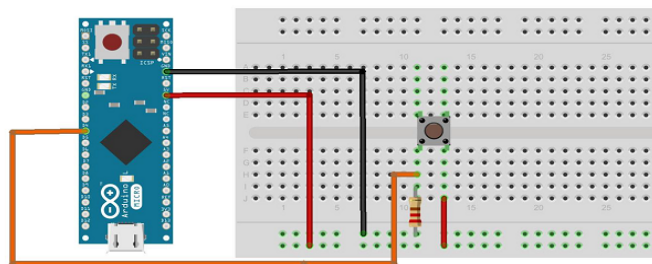
Warning- Saat Anda menggunakan perintah **Keyboard.print()** , Arduino mengambil alih keyboard komputer Anda. Untuk memastikan Anda tidak kehilangan kendali atas komputer saat menjalankan sketsa dengan fungsi ini, siapkan sistem kontrol yang andal sebelum Anda memanggil **Keyboard.print()** . Sketsa ini menyertakan tombol untuk mengaktifkan keyboard, sehingga hanya berjalan setelah tombol ditekan.

Komponen yang Diperlukan:

- 1 × *Breadboard*
- 1 × papan Arduino Leonardo, Micro, atau Due
- 1 × tombol tekan sesaat
- Resistor 1 × 10k ohm

Prosedur:

Ikuti diagram rangkaian dan hubungkan komponen-komponen pada *Breadboard* seperti yang ditunjukkan pada gambar di bawah ini.



Gambar 4. 13 Rangkaian *keyboard message*

Sketsa:

Buka perangkat lunak Arduino IDE di komputer Anda. Pengkodean dalam bahasa Arduino akan mengontrol sirkuit Anda. Buka File sketsa baru dengan mengklik Baru. Seperti pada Gambar 4.4.

Program Arduino:

```

/*
  Tes Pesan Keyboard Untuk Arduino Leonardo dan Micro,
  Mengirim string teks saat tombol ditekan.
  Sirkuit:
  * tombol tekan terpasang dari pin 4 ke + 5V
  * Resistor 10-kilohm terpasang dari pin 4 ke ground
*/
#include "Keyboard.h"
const int buttonPin = 4; // Pin input untuk tombol tekan
int previousButtonState = HIGH; // untuk memeriksa status pushButton
int counter = 0; // tombol tekan counter
void setup() {
  pinMode(buttonPin, INPUT); // jadikan pin pushButton sebagai input:
  Keyboard.begin(); // Menginisialisasi kontrol atas keyboard:
}
void loop() {
  int buttonState = digitalRead(buttonPin); // Baca tombol tekan:
  if ((buttonState != previousButtonState)&& (buttonState == HIGH)) //
  dan saat ini ditekan:
  {
    // menambah penghitung tombol
    counter++;
    // Mengetik pesan
    Keyboard.print("You pressed the button ");
    Keyboard.print(counter);
    Keyboard.println(" times.");
  }
  // Simpan status tombol saat ini untuk perbandingan lain kali:
  previousButtonState = buttonState;
}

```

Hubungkan satu terminal tombol tekan ke pin 4 pada Arduino. Hubungkan pin lainnya ke 5V. Gunakan resistor sebagai penarik, yang menyediakan referensi ke ground, dengan menghubungkannya dari pin 4 ke ground.

Setelah Anda memprogram papan Anda, cabut kabel USB, buka editor teks dan letakkan kursor teks di area pengetikan. Hubungkan papan ke komputer Anda melalui USB lagi dan tekan tombol untuk menulis di dokumen.

Hasil:

Dengan menggunakan editor teks apa pun, ini akan menampilkan teks yang dikirim melalui Arduino.

G. Mouse Button Control

Dengan menggunakan perpustakaan Mouse, Anda dapat mengontrol kursor di layar komputer dengan Arduino Leonardo, Micro, atau Due.

Contoh khusus ini menggunakan lima tombol untuk menggerakkan kursor pada layar. Empat tombolnya bersifat arah (atas, bawah, kiri, kanan) dan satu lagi untuk klik kiri mouse. Pergerakan kursor dari Arduino selalu relatif. Setiap kali input dibaca, posisi kursor diperbarui relatif terhadap posisinya saat ini.

Setiap kali salah satu tombol arah ditekan, Arduino akan menggerakkan mouse, memetakan masukan TINGGI ke rentang 5 ke arah yang sesuai.

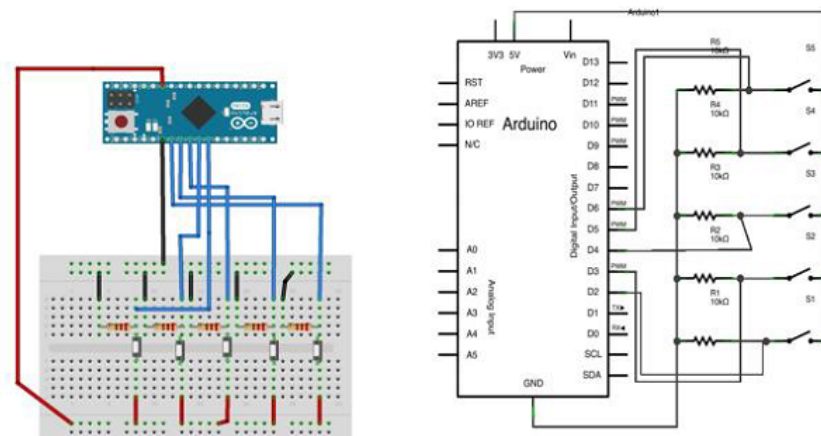
Tombol kelima berfungsi untuk mengendalikan klik kiri dari tetikus. Saat tombol dilepaskan, komputer akan mengenali kejadian tersebut.

Komponen yang Diperlukan:

- 1 × *Breadboard*
- 1 × Papan Arduino Leonardo, Micro atau Due
- Resistor 5 × 10k ohm
- 5 × tombol tekan sesaat

Prosedur:

Ikuti diagram rangkaian dan hubungkan komponen-komponen pada *Breadboard* seperti yang ditunjukkan pada gambar di bawah ini.



Gambar 4. 14 Rangkaian mouse button control

Sketsa:

Buka perangkat lunak Arduino IDE di komputer Anda. Pengkodean dalam bahasa Arduino akan mengontrol sirkuit Anda. Buka File sketsa baru dengan mengklik

Baru. Seperti terlihat pada Gambar 4.4. Untuk contoh ini, Anda perlu menggunakan Arduino IDE 1.6.7

Program Arduino:

```
/*
  Kontrol Tombol Mouse
  Hanya untuk papan Leonardo dan Due . Mengontrol mouse dari
  lima tombol tekan pada Arduino Leonardo, Micro atau Due.
  Perangkat keras:
  * 5 tombol tekan terpasang pada D2, D3, D4, D5, D6
  Gerakan mouse selalu relatif. Sketsa ini berbunyi
  empat tombol tekan, dan menggunakannya untuk mengatur pergerakan
  mouse.
  PERINGATAN: Saat Anda menggunakan perintah Mouse.move(), Arduino
  mengambil
  di atas mouse Anda! Pastikan Anda memiliki kontrol sebelum menggunakan
  perintah mouse.
*/

#include "Mouse.h"
// Mengatur nomor pin untuk lima tombol:
const int upButton = 2;
const int downButton = 3;
const int leftButton = 4;
const int rightButton = 5;
const int mouseButton = 6;
int range = 5; // rentang keluaran gerakan X atau Y; mempengaruhi
kecepatan gerakan
int responseDelay = 10; // Penundaan respons mouse, dalam ms
void setup() {
  // Inisialisasi input tombol:
  pinMode(upButton, INPUT);
  pinMode(downButton, INPUT);
  pinMode(leftButton, INPUT);
  pinMode(rightButton, INPUT);
  pinMode(mouseButton, INPUT);
  // Inisialisasi kontrol mouse:
  Mouse.begin();
}

void loop() {
  // Baca tombolnya:
```

```

int upState = digitalRead(upButton);
int downState = digitalRead(downButton);
int rightState = digitalRead(rightButton);
int leftState = digitalRead(leftButton);
int clickState = digitalRead(mouseButton);
// Hitung jarak gerakan berdasarkan status tombol:
int xDistance = (leftState - rightState) * range;
int yDistance = (upState - downState) * range;
// jika X atau Y bukan nol, pindahkan:
if ((xDistance != 0) || (yDistance != 0)) {
    Mouse.move(xDistance, yDistance, 0);
}

// Jika tombol mouse ditekan:
if (clickState == HIGH) {
    // Jika mouse tidak ditekan, tekan:
    if (!Mouse.isPressed(MOUSE_LEFT)) {
        Mouse.press(MOUSE_LEFT);
    }
} else { // jika tidak, tombol mouse tidak ditekan:
    // Jika mouse ditekan, lepaskan:
    if (Mouse.isPressed(MOUSE_LEFT)) {
        Mouse.release(MOUSE_LEFT);
    }
}

// penundaan agar mouse tidak bergerak terlalu cepat:
delay(responseDelay);
}

```

Hubungkan papan Anda ke komputer dengan kabel micro-USB. Tombol-tombol tersebut terhubung ke input digital dari pin 2 hingga 6. Pastikan Anda menggunakan resistor pull-down 10k.

H. Keyboard Serial

Contoh ini mendengarkan byte yang datang dari port serial. Saat diterima, papan mengirimkan penekanan tombol kembali ke komputer. Penekanan tombol yang dikirim lebih tinggi satu dari yang diterima, jadi jika Anda mengirim "a" dari monitor serial, Anda akan menerima "b" dari papan yang terhubung ke komputer. "1" akan mengembalikan "2" dan seterusnya.

Warning- Saat Anda menggunakan perintah `Keyboard.print()` , papan Leonardo, Micro, atau Due akan mengambil alih keyboard komputer Anda. Untuk memastikan Anda tidak kehilangan kendali atas komputer Anda saat menjalankan sketsa dengan fungsi ini, siapkan sistem kontrol yang andal sebelum Anda memanggil `Keyboard.print()`. Sketsa ini dirancang untuk hanya mengirim perintah Keyboard setelah papan menerima byte melalui port serial.

Komponen yang Diperlukan:

- 1 × Papan Arduino Leonardo, Micro, atau Due

Prosedur:

Cukup sambungkan papan Anda ke komputer menggunakan kabel USB.

Sketsa:

Buka perangkat lunak Arduino IDE di komputer Anda. Pengkodean dalam bahasa Arduino akan mengontrol sirkuit Anda. Buka File sketsa baru dengan mengklik Baru. Seperti pada Gambar 4.4.

Catatan – Anda harus menyertakan pustaka keypad dalam berkas pustaka Arduino Anda. Salin dan tempel berkas pustaka keypad di dalam berkas dengan nama 'libraries' yang disorot dengan warna kuning. Seperti pada Gambar 4.12.

Program Arduino:

```
/*
  Tes keyboard
  Untuk Arduino Leonardo, Micro atau Due Membaca
  byte dari port serial, mengirim penekanan tombol kembali.
  Keystroke yang dikirim adalah satu lebih tinggi dari apa yang
  diterima, misalnya jika Anda mengirim a, Anda mendapatkan b, kirim
  A Anda mendapatkan B, dan sebagainya.
  Sirkuit:
  * tidak
*/
#include "Keyboard.h"
void setup() {
  // Buka port serial:
  Serial.begin(9600);
  // Menginisialisasi kontrol atas keyboard:
  Keyboard.begin();
}
void loop() {
  // Periksa data seri yang masuk:
  if (Serial.available() > 0) {
```

```
// Baca data seri yang masuk:  
char inChar = Serial.read();  
// Ketik nilai ASCII berikutnya dari apa yang Anda terima:  
Keyboard.write(inChar + 1);  
}  
}
```

Setelah diprogram, buka monitor serial Anda dan kirim satu byte. Dewan akan membalas dengan menekan tombol, yaitu satu angka lebih tinggi.

Hasil:

Papan akan membalas dengan penekanan tombol yang satu angka lebih tinggi pada monitor serial Arduino IDE saat Anda mengirim satu byte.

BAB 5

SENSOR ARDUINO

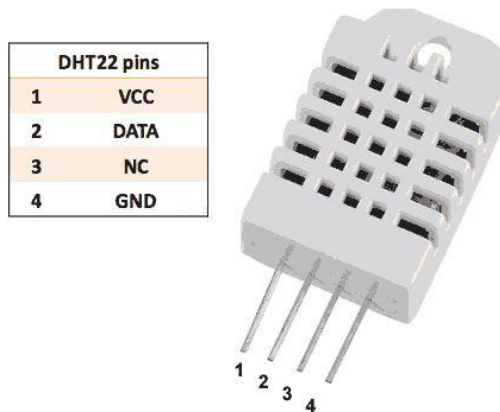
Bagian ini, kita akan mempelajari cara menghubungkan papan Arduino kita dengan sensor yang berbeda. Kami akan membahas sensor berikut:

- Sensor Kelembaban (DHT22)
- Sensor suhu (LM35)
- Sensor detektor air (Simple Water Trigger)
- SENSOR PIR
- SENSOR ULTRASONIK
- Saklar Penghubung

A. Sensor Kelembaban

DHT-22 (juga dinamai AM2302) adalah keluaran digital, kelembaban relatif, dan sensor suhu. Ia menggunakan sensor kelembaban kapasitif dan termistor untuk mengukur udara di sekitarnya, dan mengirimkan sinyal digital pada pin data.

Dalam contoh ini, Anda akan mempelajari cara menggunakan sensor ini dengan Arduino UNO. Suhu dan kelembaban ruangan akan dicetak ke monitor serial.



Gambar 5. 1 Sensor DHT-22

Koneksinya sederhana. Pin pertama di sebelah kiri untuk daya 3-5V, pin kedua untuk pin input data, dan pin paling kanan untuk ground.

Detail teknis:

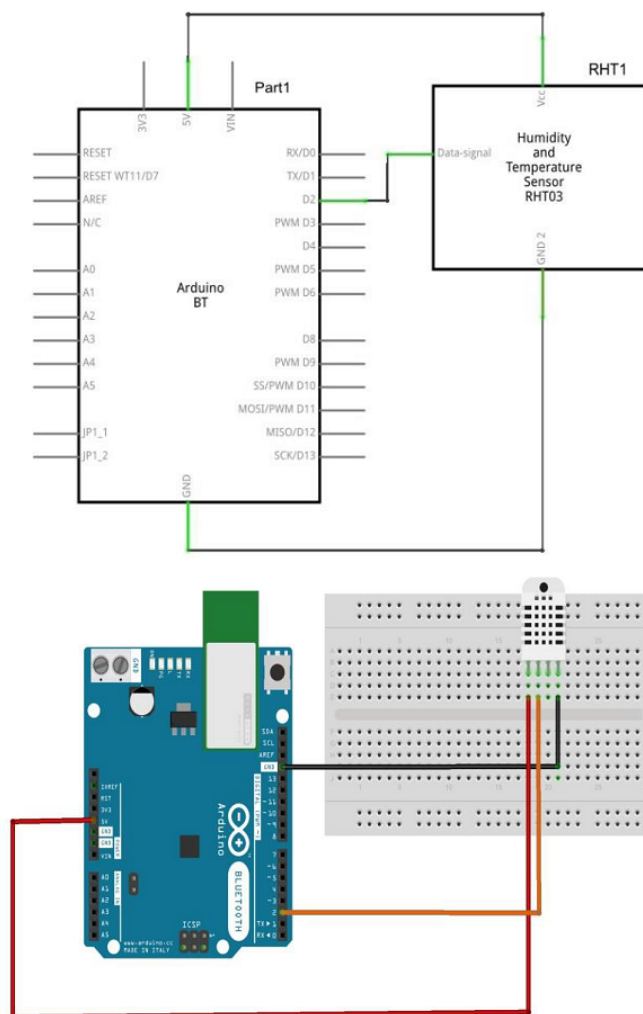
- Daya – 3-5V
- Arus Maks - 2,5mA

- **Kelembaban** – 0-100%, akurasi 2-5%
- **Suhu** – 40 hingga 80°C, akurasi $\pm 0.5^{\circ}\text{C}$

Komponen yang Diperlukan:

- 1 $\times$ Breadboard
- 1 $\times$ Arduino Uno R3
- 1 $\times$ DHT22
- 1 $\times$ resistor 10k ohm

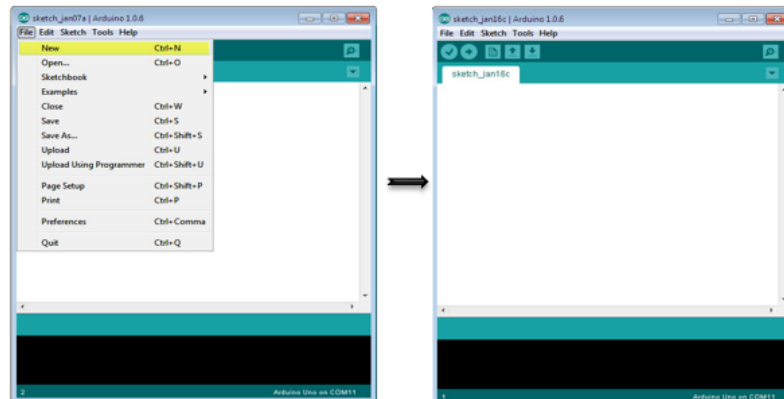
Prosedur:



Gambar 5. 2 Rangkaian proyek sensor kelembaban

Sketsa:

Buka perangkat lunak Arduino IDE di komputer Anda. Pengodean dalam bahasa Arduino akan mengendalikan sirkuit Anda. Buka File sketsa baru dengan mengklik Baru. Seperti terlihat pada Gambar 5.3.



Gambar 5. 3 Membuka *Sketch* baru di Arduino IDE

Program Arduino:

```
// Contoh sketsa pengujian untuk berbagai sensor kelembaban / suhu DHT
#include "DHT.h"
#define DHTPIN 2 // Pin digital apa yang terhubung dengan kami
// Hapus tanda komentar apa pun jenis yang Anda gunakan!
// #define DHTTYPE DHT11 // DHT 11
#define DHTTYPE DHT22 // DHT 22 (AM2302), AM2321
// #define DHTTYPE DHT21 // DHT 21 (AM2301)
// Hubungkan pin 1 (di sebelah kiri) sensor ke +5V
// CATATAN: Jika menggunakan papan dengan logika 3.3V seperti Arduino
// Due connect pin 1
// ke 3.3V, bukan 5V!
// Hubungkan pin 2 sensor ke apa pun DHTPIN Anda
// Hubungkan pin 4 (di sebelah kanan) sensor ke GROUND
// Hubungkan resistor 10K dari pin 2 (data) ke pin 1 (daya) sensor
// Inisialisasi sensor DHT.
// Perhatikan bahwa versi library yang lebih lama mengambil parameter
// ketiga opsional untuk
// Sesuaikan pengaturan waktu untuk prosesor yang lebih cepat. Parameter
// ini tidak lagi diperlukan
// karena algoritma pembacaan DHT saat ini menyesuaikan diri untuk
// bekerja pada proc yang lebih cepat.

DHT dht(DHTPIN, DHTTYPE);

void setup() {
    Serial.begin(9600);
    Serial.println("DHTxx test!");
    dht.begin();
}
```

```

}

void loop() {
    delay(2000); // Wait a few seconds between measurements
    float h = dht.readHumidity();
    // Membaca suhu atau kelembaban membutuhkan waktu sekitar 250
    milidetik!
    float t = dht.readTemperature();
    // Baca suhu sebagai Celsius (default)
    float f = dht.readTemperature(true);
    // Baca suhu sebagai Fahrenheit (isFahrenheit = true)
    // Periksa apakah ada bacaan yang gagal dan keluar lebih awal (untuk
    mencoba lagi).
    if (isnan(h) || isnan(t) || isnan(f)) {
        Serial.println("Failed to read from DHT sensor!");
        return;
    }

    // Hitung indeks panas dalam Fahrenheit (default)
    float hif = dht.computeHeatIndex(f, h);
    // Hitung indeks panas dalam Celsius (isFahreheit = false)
    float hic = dht.computeHeatIndex(t, h, false);
    Serial.print ("Humidity: ");
    Serial.print (h);
    Serial.print (" %\t");
    Serial.print ("Temperature: ");
    Serial.print (t);
    Serial.print (" *C ");
    Serial.print (f);
    Serial.print (" *F\t");
    Serial.print ("Heat index: ");
    Serial.print (hic);
    Serial.print (" *C ");
    Serial.print (hif);
    Serial.println (" *F");
}

```

Sensor DHT22 memiliki empat terminal (V_{cc} , DATA, NC, GND), yang dihubungkan ke papan sebagai berikut:

- Pin DATA ke pin Arduino nomor 2

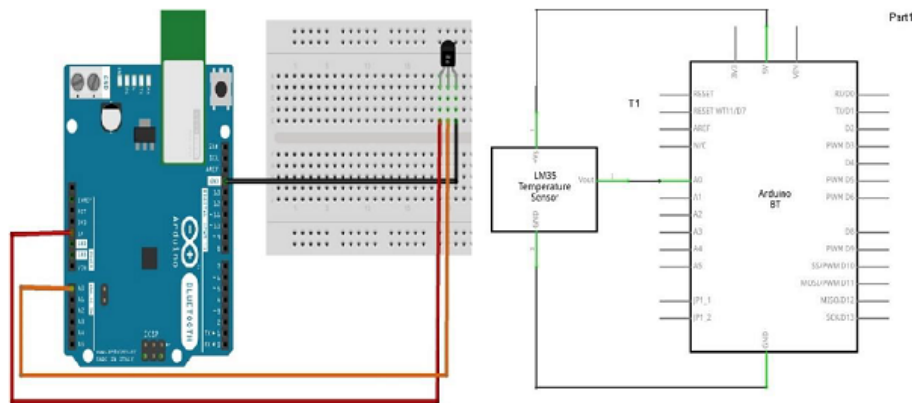
- 0,5°C memastikan akurasi (pada 25°C)
- Dinilai untuk rentang penuh -55°C hingga 150°C
- Cocok untuk aplikasi jarak jauh

Komponen yang Diperlukan:

- 1 × Breadboard
- 1 × Arduino Uno R3
- 1 × sensor LM35

Prosedur:

Ikuti diagram sirkuit dan sambungkan komponen pada *Breadboard* seperti yang ditunjukkan pada gambar di bawah.



Gambar 5. 5 Rangkaian proyek sensor suhu LM35

Sketch:

Buka perangkat lunak Arduino IDE di komputer Anda. Pengkodean dalam bahasa Arduino akan mengontrol sirkuit Anda. Buka File sketsa baru dengan mengklik Baru. Seperti terlihat pada Gambar 5.3.

Program Arduino:

```
float temp;
int tempPin = 0;
void setup() {
    Serial.begin(9600);
}
void loop() {
    temp = analogRead(tempPin);
    // Baca volt analog dari sensor dan simpan ke suhu variabel
    temp = temp * 0.48828125;
    // Ubah volt analog menjadi setara suhunya
    Serial.print("TEMPERATURE = ");
```

```

Serial.print(temp); // menampilkan nilai suhu
Serial.print("*C");
Serial.println();
delay(1000); // Perbarui pembacaan sensor setiap satu detik
}

```

Sensor LM35 memiliki tiga terminal - V_s , V_{out} dan GND. Kami akan menghubungkan sensor sebagai berikut:

- Hubungkan $+V_{ke} +5v$ pada papan Arduino Anda.
- Hubungkan V_{out} ke Analog0 atau A0 pada papan Arduino.
- Hubungkan GND dengan GND pada Arduino.

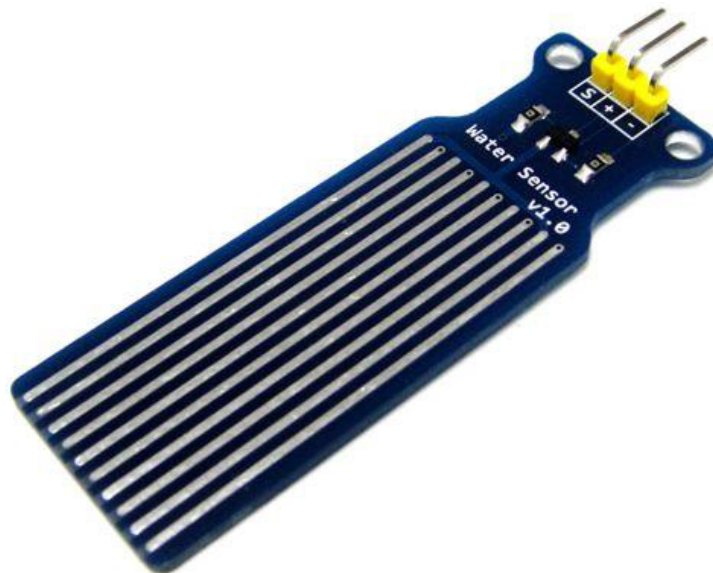
Konverter Analog ke Digital (ADC) mengubah nilai analog menjadi perkiraan digital berdasarkan rumus Nilai ADC = sampel * 1024 / tegangan referensi (+5v). Jadi dengan referensi +5 volt, perkiraan digital akan sama dengan tegangan input * 205.

Hasil:

Anda akan melihat tampilan suhu pada monitor port serial yang diperbarui setiap detik.

C. Detektor / Sensor Air

Bata sensor air dirancang untuk mendeteksi air, yang dapat digunakan secara luas dalam mendeteksi curah hujan, ketinggian air, dan bahkan kebocoran cairan.



Gambar 5. 6 Sensor air

Menghubungkan sensor air ke Arduino adalah cara terbaik untuk mendeteksi kebocoran, tumpahan, banjir, hujan, dll. Sensor ini dapat digunakan untuk mendeteksi keberadaan, ketinggian, volume dan/atau tidak adanya air. Meskipun ini dapat digunakan untuk mengingatkan Anda untuk menyiram tanaman, ada sensor Grove yang lebih baik untuk itu. Sensor memiliki serangkaian jejak terbuka, yang terbaca RENDAH ketika air terdeteksi.

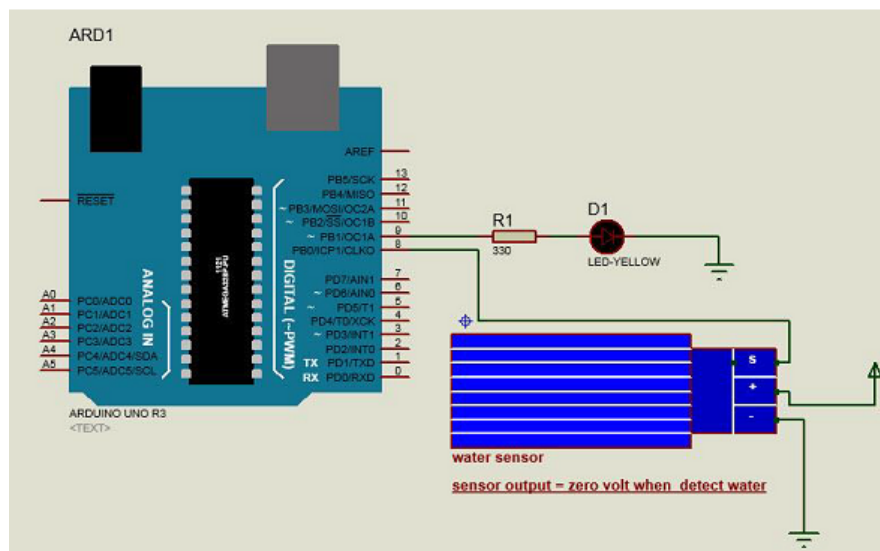
Dalam bab ini, kita akan menghubungkan sensor air ke Pin Digital 8 pada Arduino, dan akan menggunakan LED yang sangat berguna untuk membantu mengidentifikasi saat sensor air bersentuhan dengan sumber air.

Komponen yang Diperlukan:

- 1 × Breadboard
- 1×Arduino Uno R3
- 1 × Sensor Air
- 1 × dipimpin
- Resistor 1 × 330 ohm

Prosedur:

Ikuti diagram rangkaian dan hubungkan komponen-komponen pada Breadboard seperti yang ditunjukkan pada gambar di bawah ini.



Gambar 5. 7 Rangkaian proyek sensor suhu

Sketch:

Buka perangkat lunak Arduino IDE di komputer Anda. Pengodean dalam bahasa Arduino akan mengendalikan sirkuit Anda. Buka File sketsa baru dengan mengeklik Baru. Seperti pada Gambar 5.3.

Program Arduino:

```
#define Grove_Water_Sensor 8 // Pasang sensor Air ke Arduino Digital Pin 8
#define LED 9 // Pasang LED ke Digital Pin 9 (atau gunakan LED onboard)
void setup() {
    pinMode(Grove_Water_Sensor, INPUT); // Sensor Air adalah Input
    pinMode(LED, OUTPUT); // LED adalah Output
}
void loop() {
    /* Sensor air akan beralih RENDAH ketika air terdeteksi.
    Dapatkan Arduino untuk menerangi LED dan aktifkan bel
    ketika air terdeteksi, dan matikan keduanya ketika tidak ada air */
    if( digitalRead(Grove_Water_Sensor) == LOW) {
        digitalWrite(LED,HIGH);
    }else {
        digitalWrite(LED,LOW);
    }
}
```

Sensor air memiliki tiga terminal - S, V_{out} (+), dan GND (-). Hubungkan sensor sebagai berikut:

- Hubungkan $+V_{ke} +5v$ pada papan Arduino Anda.
- Hubungkan S ke pin digital nomor 8 pada papan Arduino.
- Hubungkan GND dengan GND pada Arduino.
- Hubungkan LED ke pin digital nomor 9 di papan Arduino.

Ketika sensor mendeteksi air, pin 8 pada Arduino menjadi LOW dan kemudian LED pada Arduino menyala.

Hasil:

Anda akan melihat lampu LED indikasi menyala ketika sensor mendeteksi air.

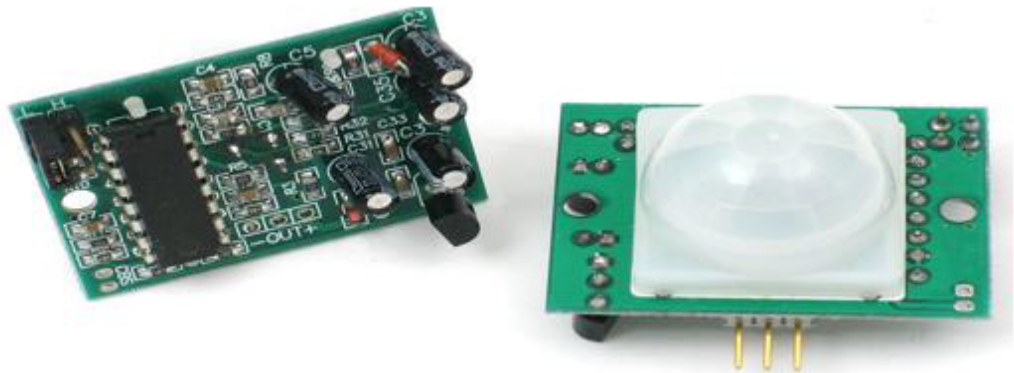
D. Sensor PIR

Sensor PIR memungkinkan Anda merasakan gerakan. Mereka digunakan untuk mendeteksi apakah manusia telah masuk atau keluar dari jangkauan sensor. Mereka umumnya ditemukan pada peralatan dan gadget yang digunakan di rumah atau untuk bisnis. Mereka sering disebut sebagai sensor PIR, "Inframerah Pasif", "Pyroelectric", atau "gerakan IR".

Berikut ini adalah keuntungan dari Sensor PIR:

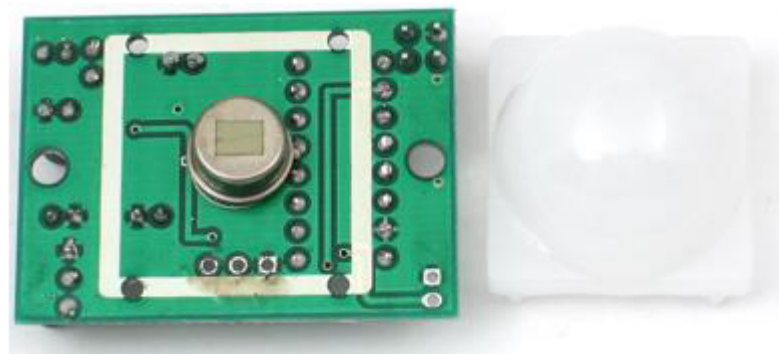
- Berukuran kecil

- Jangkauan lensa lebar
- Mudah untuk berinteraksi
- Murah
- Daya rendah
- Mudah digunakan
- Jangan sampai aus



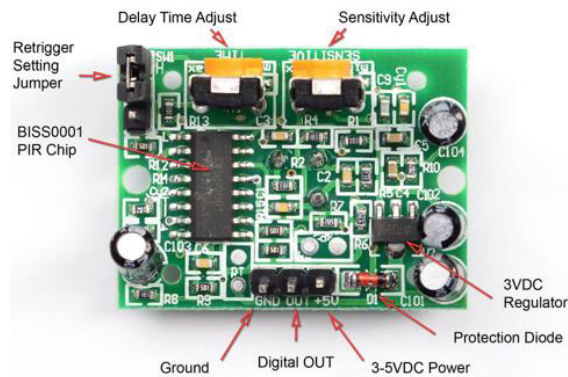
Gambar 5. 8 Sensor PIR

PIR terbuat dari sensor piroelektrik, kaleng logam bundar dengan kristal persegi panjang di bagian tengah, yang dapat mendeteksi tingkat radiasi inframerah. Segala sesuatu memancarkan radiasi tingkat rendah, dan semakin panas sesuatu, semakin banyak radiasi yang dipancarkan. Sensor dalam detektor gerakan dibagi menjadi dua bagian. Ini untuk mendeteksi gerakan (perubahan) dan bukan tingkat IR rata-rata. Kedua bagian tersebut terhubung sehingga saling meniadakan. Jika satu bagian melihat lebih banyak atau lebih sedikit radiasi IR daripada yang lain, output akan berayun tinggi atau rendah.



Gambar 5. 9 Bagian dalam sensor PIR

PIR memiliki pengaturan yang dapat disesuaikan dan memiliki header yang dipasang di ground/out/power pad 3-pin.



Gambar 5. 10 Pengaturan pada sensor PIR

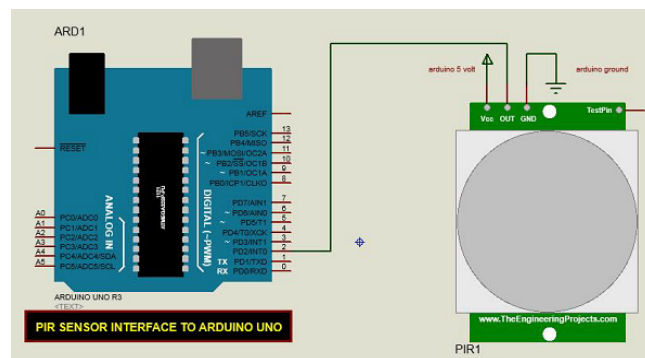
Untuk banyak proyek atau produk dasar yang perlu mendeteksi ketika seseorang telah meninggalkan atau memasuki area tersebut, sensor PIR sangat bagus. Perhatikan bahwa PIR tidak memberi tahu Anda jumlah orang di sekitar atau kedekatan mereka dengan sensor. Lensa sering kali dipasang pada sapuan tertentu pada jarak tertentu dan terkadang dipasang oleh hewan peliharaan di rumah.

Komponen yang Diperlukan:

- 1 × Breadboard
- 1 × Arduino Uno R3
- 1 × Sensor PIR (MQ3)

Prosedur:

Ikuti diagram rangkaian dan buat sambungan seperti ditunjukkan pada gambar di bawah.



Gambar 5. 11 Rangkaian proyek sensor PIR

Sketch:

Buka perangkat lunak Arduino IDE di komputer Anda. Pengkodean dalam bahasa Arduino akan mengontrol sirkuit Anda. Buka File sketsa baru dengan mengklik Baru. Seperti pada Gambar 5.3.

Program Arduino:

```
#define pirPin 2
int calibrationTime = 30;
long unsigned int lowIn;
long unsigned int pause = 5000;
boolean lockLow = true;
boolean takeLowTime;
int PIRValue = 0;

void setup() {
    Serial.begin(9600);
    pinMode(pirPin, INPUT);
}

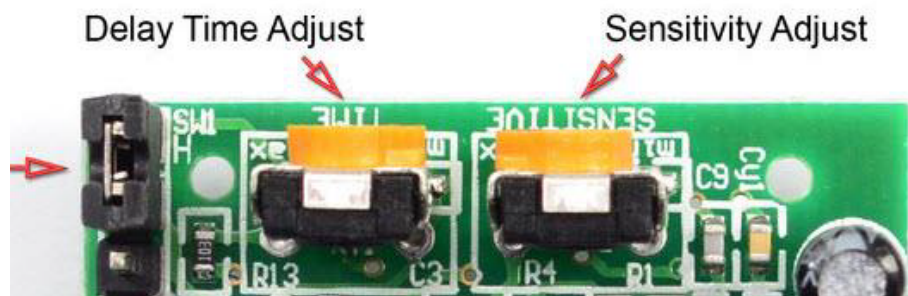
void loop() {
    PIRSensor();
}

void PIRSensor() {
    if(digitalRead(pirPin) == HIGH) {
        if(lockLow) {
            PIRValue = 1;
            lockLow = false;
            Serial.println("Motion detected.");
            delay(50);
        }
        takeLowTime = true;
    }
    if(digitalRead(pirPin) == LOW) {
        if(takeLowTime){
            lowIn = millis();takeLowTime = false;
        }
        if(!lockLow && millis() - lowIn > pause) {
            PIRValue = 0;
            lockLow = true;
            Serial.println("Motion ended.");
            delay(50);
        }
    }
}
```

Sensor PIR memiliki tiga terminal - V_{cc} , OUT dan GND. Hubungkan sensor sebagai berikut:

- Hubungkan $+V_{cc}$ ke +5v di papan Arduino.
- Hubungkan OUT ke pin digital 2 pada papan Arduino.
- Hubungkan GND dengan GND di Arduino.

Anda dapat menyesuaikan sensitivitas sensor dan waktu tunda melalui dua resistor variabel yang terletak di bagian bawah papan sensor.



Gambar 5. 12 Pengaturan waktu tunda dan sensitifitas sensor PIR

Setelah sensor mendeteksi adanya gerakan, Arduino akan mengirim pesan melalui port serial yang menyatakan bahwa gerakan terdeteksi. Sensor gerak PIR akan menunda beberapa saat untuk memeriksa apakah ada gerakan baru. Jika tidak ada gerakan yang terdeteksi, Arduino akan mengirim pesan baru yang menyatakan bahwa gerakan telah berakhir.

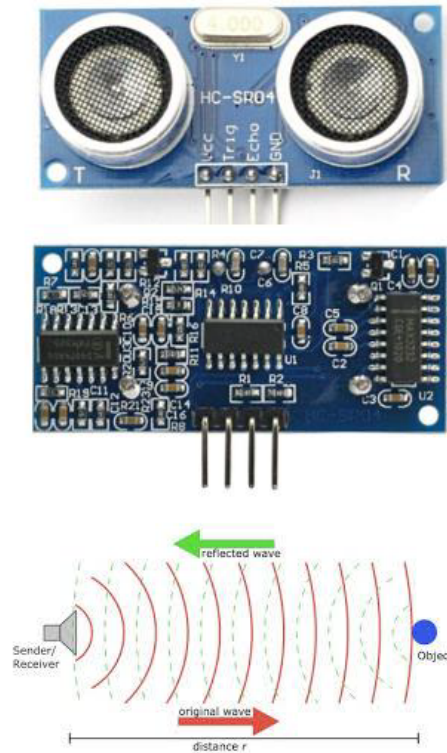
Hasil:

Anda akan melihat pesan di port serial Anda jika gerakan terdeteksi dan pesan lain saat gerakan berhenti.

E. Sensor Ultrasonik

Sensor ultrasonik HC-SR04 menggunakan SONAR untuk menentukan jarak suatu benda seperti yang dilakukan kelelawar. Ia menawarkan deteksi jangkauan non-kontak yang sangat baik dengan akurasi tinggi dan pembacaan stabil dalam paket yang mudah digunakan dari 2 cm hingga 400 cm atau 1" hingga 13 kaki.

Pengoperasiannya tidak terpengaruh oleh sinar matahari atau material hitam, meskipun secara akustik, material lembut seperti kain sulit dideteksi. Dilengkapi dengan modul pemancar dan penerima ultrasonik.



Gambar 5. 13 Sensor ultrasonik dan prinsip kerja

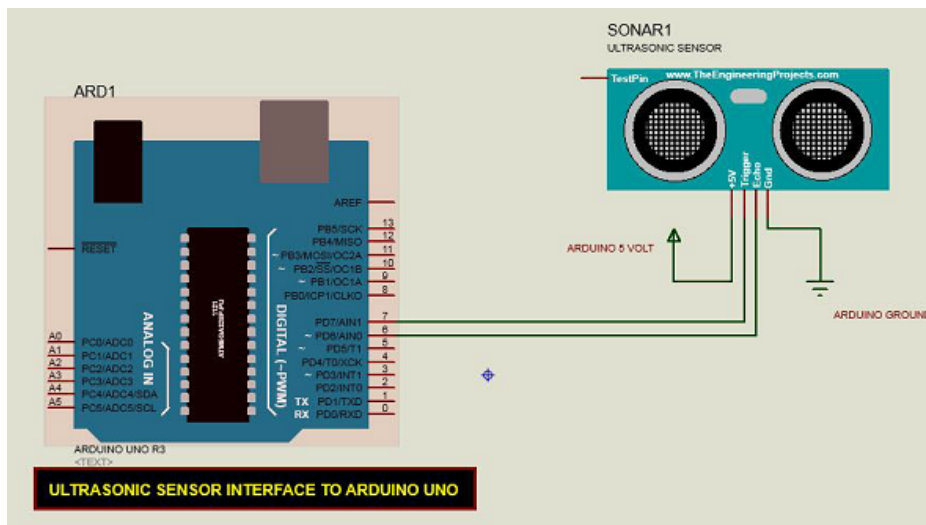
Spesifikasi teknis:

- Catu Daya - +5V DC
- Arus Diam - <2mA
- Arus Kerja – 15mA
- Sudut Efektif – <15°
- Jarak Jangkauan – 2cm – 400 cm/1" – 13ft
- Resolusi – 0,3 cm
- Mengukur Sudut – 30 derajat

Komponen yang Diperlukan:

- 1 × *Breadboard*
- 1 × Arduino Uno R3
- 1 × Sensor ULTRASONIK (HC-SR04)

Prosedur:



Gambar 5. 14 Rangkaian proyek sensor ultrasonik

Sketch:

Buka perangkat lunak Arduino IDE di komputer Anda. Pengkodean dalam bahasa Arduino akan mengontrol sirkuit Anda. Buka File sketsa baru dengan mengklik Baru. Seperti terlihat pada Gambar 5.3.

Program Arduino:

```
const int pingPin = 7; // Pin trigger Sensor Ultrasonik
const int echoPin = 6; // Pin echo Sensor Ultrasonik
void setup() {
    Serial.begin(9600); // Memulai Terminal Serial
}
void loop() {
    long duration, inches, cm;
    pinMode(pingPin, OUTPUT);
    digitalWrite(pingPin, LOW);
    delayMicroseconds(2);
    digitalWrite(pingPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(pingPin, LOW);
    pinMode(echoPin, INPUT);
    duration = pulseIn(echoPin, HIGH);
    inches = microsecondsToInches(duration);
    cm = microsecondsToCentimeters(duration);
    Serial.print(inches);
    Serial.print("in, ");
```

```

    Serial.print(cm);
    Serial.print("cm");
    Serial.println();
    delay(100);
}
long microsecondsToInches(long microseconds) {
    return microseconds / 74 / 2;
}
long microsecondsToCentimeters(long microseconds) {
    return microseconds / 29 / 2;
}

```

Sensor Ultrasonik memiliki empat terminal - +5V, Pemicu, Gema, dan GND yang terhubung sebagai berikut:

- Hubungkan pin +5V ke +5v pada papan Arduino Anda.
- Hubungkan Pemicu ke pin digital 7 pada papan Arduino Anda.
- Hubungkan Echo ke pin digital 6 di papan Arduino Anda.
- Hubungkan GND dengan GND di Arduino.

Dalam program kami, kami telah menampilkan jarak yang diukur oleh sensor dalam inci dan cm melalui port serial.

Hasil:

Anda akan melihat jarak yang diukur oleh sensor dalam inci dan cm pada monitor serial Arduino.

F. Saklar Penghubung

Tombol tekan atau sakelar menghubungkan dua terminal terbuka dalam suatu sirkuit. Contoh ini menyalakan LED pada pin 2 ketika Anda menekan sakelar tombol yang terhubung ke pin 8.



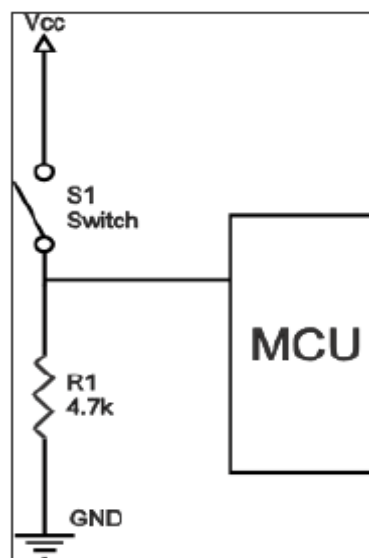
Gambar 5. 15 Saklar tombol

Resistor Pull-down

Resistor pull-down digunakan dalam rangkaian logika elektronik untuk memastikan bahwa input ke Arduino berada pada tingkat logika yang diharapkan jika perangkat eksternal terputus atau berada pada impedansi tinggi. Karena tidak ada yang terhubung ke pin input, itu tidak berarti logikanya nol. Resistor tarik ke bawah dihubungkan antara ground dan pin yang sesuai pada perangkat.

Contoh resistor pull-down dalam rangkaian digital ditunjukkan pada gambar berikut. Sakelar tombol tekan dihubungkan antara tegangan suplai dan pin mikrokontroler. Dalam rangkaian seperti itu, saat sakelar ditutup, masukan mikrokontroler berada pada nilai logika tinggi, tetapi saat sakelar terbuka, resistor pull-down menarik tegangan masukan ke bawah ke ground (nilai logika nol), mencegah keadaan yang tidak terdefinisi pada masukan.

Resistor pull-down harus memiliki resistansi yang lebih besar daripada impedansi rangkaian logika, jika tidak maka resistor tersebut akan menurunkan tegangan terlalu banyak dan tegangan masukan pada pin akan tetap pada nilai logis rendah yang konstan, terlepas dari posisi sakelar.



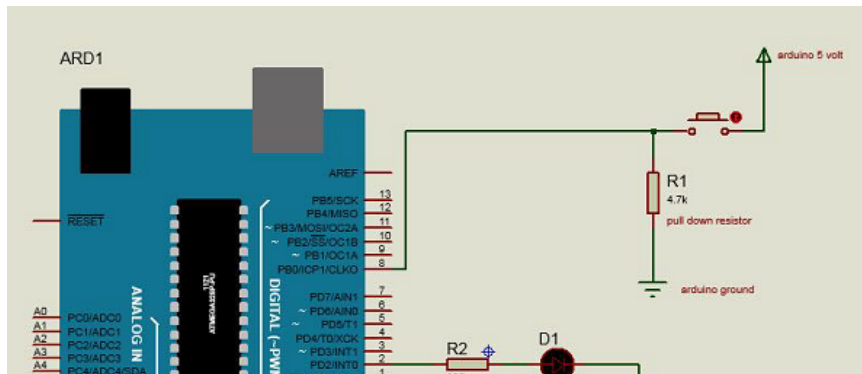
Gambar 5. 16 Rangkaian digital resistor pull-down

Komponen yang Diperlukan:

- 1 × Papan Arduino UNO
- 1 × 330 ohm resistor
- 1 × resistor 4,7K ohm (tarik ke bawah)
- 1×LED

Prosedur:

Ikuti diagram rangkaian dan buat sambungan seperti ditunjukkan pada gambar di bawah ini.



Gambar 5. 17 Rangkaian proyek saklar penghubung

Sketch:

Buka perangkat lunak Arduino IDE di komputer Anda. Pengkodean dalam bahasa Arduino akan mengontrol sirkuit Anda. Buka File sketsa baru dengan mengklik Baru. Seperti pada Gambar 5.3.

Program Arduino:

```
// Konstanta tidak akan berubah. Mereka digunakan di sini untuk
// Mengatur nomor PIN:
const int buttonPin = 8; // pin tombol tekan
const int ledPin = 2; // pin LED
// variables will change:
int buttonState = 0; // variabel untuk membaca status tombol tekan
void setup() {
    // menginisialisasi pin LED sebagai output:
    pinMode(ledPin, OUTPUT);
    // Menginisialisasi pin tombol tekan sebagai input:
    pinMode(buttonPin, INPUT);
}
void loop() {
    // Membaca status nilai tombol tekan:
    buttonState = digitalRead(buttonPin);
    // Periksa apakah tombol tekan ditekan.
    // jika ya, buttonState adalah HIGH:
    if (buttonState == HIGH) {
        // nyalakan LED:
        digitalWrite(ledPin, HIGH);
    }
}
```

```
    } else {  
        // matikan LED:  
        digitalWrite(ledPin, LOW);  
    }  
}
```

Ketika saklar terbuka, (tombol tidak ditekan), tidak ada hubungan antara kedua terminal tombol, sehingga pin dihubungkan ke ground (melalui resistor pull-down) dan kita membaca RENDAH. Ketika saklar ditutup (tombol ditekan), maka dibuat sambungan antara kedua terminalnya, menghubungkan pin ke 5 volt, sehingga kita membaca HIGH.

Hasil:

LED menyala saat tombol ditekan dan mati saat dilepas.

BAB 6

MOTOR DC

Dalam bab ini, kami akan menghubungkan berbagai jenis motor dengan papan Arduino (UNO) dan menunjukkan kepada Anda cara menghubungkan motor dan menggerakkannya dari papan Anda.

Ada tiga jenis motor yang berbeda:

- Motor DC
- Motor servo
- Motor stepper

A. Motor DC

Motor DC (Direct Current motor) merupakan jenis motor yang paling umum. Motor DC biasanya hanya memiliki dua kabel, satu positif dan satu negatif. Jika Anda menghubungkan kedua kabel ini langsung ke baterai, motor akan berputar. Jika Anda mengganti kabelnya, motor akan berputar ke arah yang berlawanan.



Gambar 6. 1 Motor DC 5v

Peringatan – Jangan menggerakkan motor langsung dari pin papan Arduino. Hal ini dapat merusak papan. Gunakan rangkaian driver atau IC.

Kami akan membagi bab ini menjadi tiga bagian

- Putar saja motormu
- Kontrol kecepatan motor
- Mengontrol arah putaran motor DC

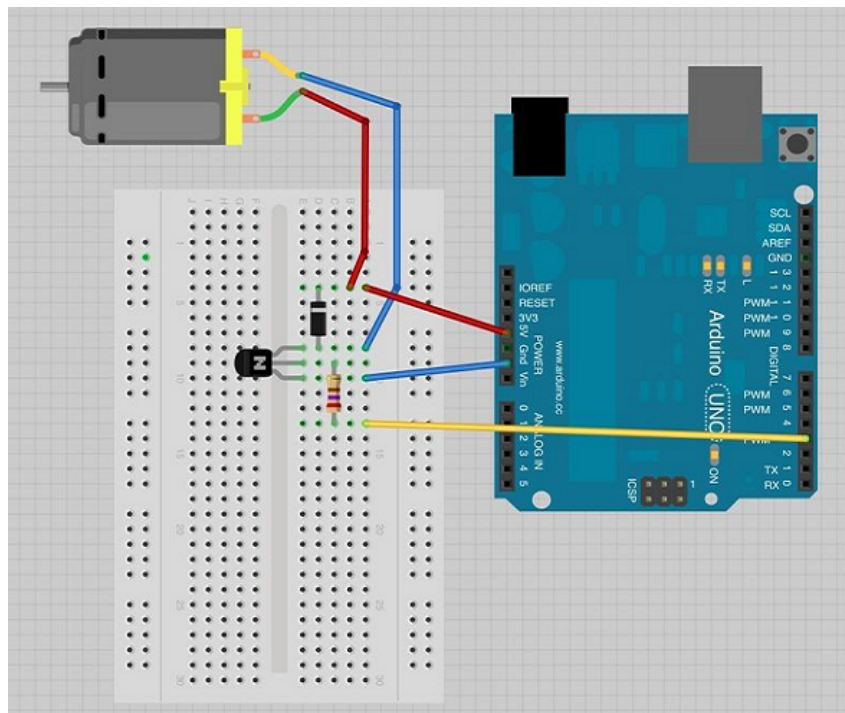
Komponen yang Diperlukan:

- 1x papan Arduino UNO

- 1 buah transistor PN2222
- 1x Motor DC 6V Kecil
- 1x dioda 1N4001
- 1x270Ω Resistor

Prosedur:

Ikuti diagram sirkuit dan buat koneksi seperti yang ditunjukkan pada gambar di bawah ini.



Gambar 6. 2 Rangkaian proyek Motor DC

Lakukan tindakan pencegahan berikut saat membuat sambungan:

- Pertama, pastikan transistor terhubung dengan benar. Sisi datar transistor harus menghadap papan Arduino seperti yang ditunjukkan pada gambar.
- Kedua, ujung dioda yang bergaris harus mengarah ke saluran listrik +5V sesuai dengan susunan yang ditunjukkan pada gambar.

Program Arduino:

```
int motorPin = 3;
void setup() {
}
void loop() {
    digitalWrite(motorPin, HIGH);
}
```

}

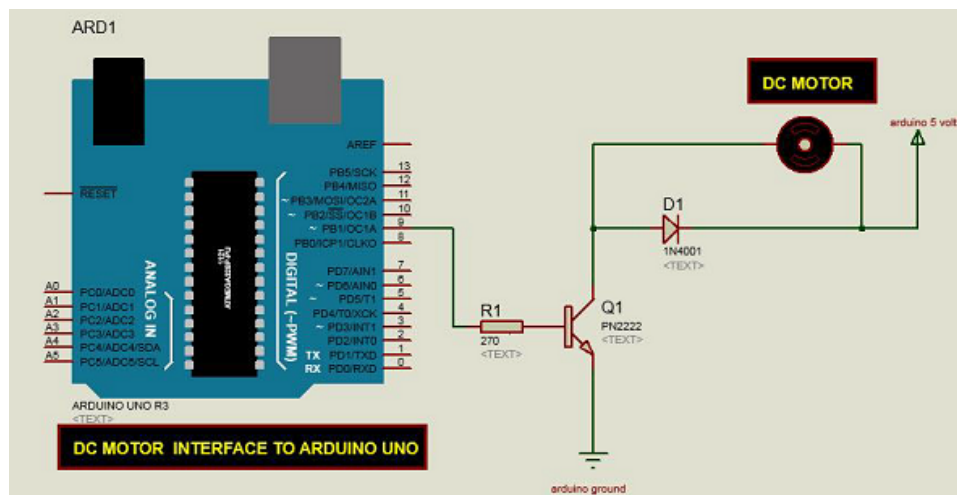
Transistor bertindak seperti saklar, mengendalikan daya ke motor. Arduino pin 3 digunakan untuk menghidupkan dan mematikan transistor dan diberi nama 'motorPin' pada sketsanya.

Hasil:

Motor akan berputar dengan kecepatan penuh ketika pin Arduino nomor 3 menjadi high.

Kontrol kecepatan motor

Berikut ini adalah diagram skema motor DC yang terhubung ke papan Arduino.



Gambar 6. 3 Rangkaian proyek untuk kecepatan motor DC

Program Arduino:

```
int motorPin = 9;

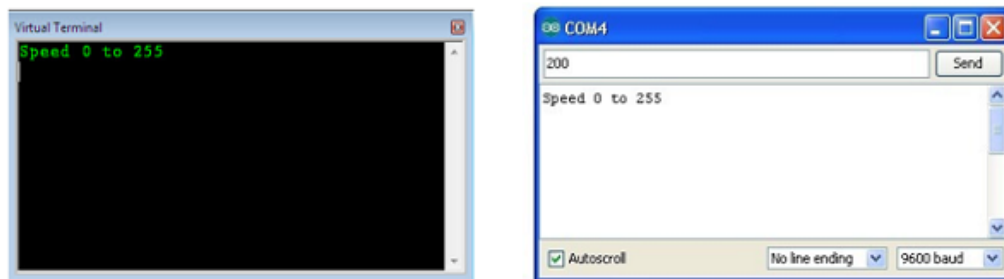
void setup() {
  pinMode(motorPin, OUTPUT);
  Serial.begin(9600);
  while (! Serial);
  Serial.println("Speed 0 to 255");
}

void loop() {
  if (Serial.available()) {
    int speed = Serial.parseInt();
    if (speed >= 0 && speed <= 255) {
      analogWrite(motorPin, speed);
    }
  }
}
```

```
}
```

Transistor bertindak seperti saklar, mengendalikan daya motor. Arduino pin 3 digunakan untuk menghidupkan dan mematikan transistor dan diberi nama 'motorPin' pada sketsanya.

Saat program dimulai, Anda akan diminta untuk memberikan nilai untuk mengontrol kecepatan motor. Anda harus memasukkan nilai antara 0 dan 255 di Serial Monitor.



Gambar 6. 4 Tampilan kecepatan pada serial monitor

Dalam fungsi 'loop', perintah 'Serial.parseInt' digunakan untuk membaca angka yang dimasukkan sebagai teks di Serial Monitor dan mengubahnya menjadi 'int'. Anda dapat mengetik angka apa pun di sini. Pernyataan 'if' di baris berikutnya hanya melakukan penulisan analog dengan angka ini, jika angka tersebut berada di antara 0 dan 255.

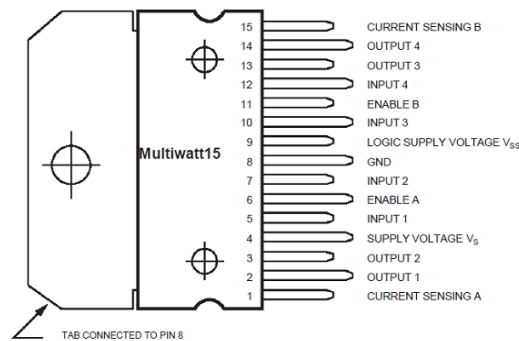
Hasil:

Motor DC akan berputar dengan kecepatan berbeda sesuai dengan nilai (0 hingga 250) yang diterima melalui port serial.

Kontrol Arah Putaran

Untuk mengendalikan arah putaran motor DC, tanpa harus menukar kabelnya, Anda dapat menggunakan rangkaian yang disebut **H-Bridge**. H-bridge adalah rangkaian elektronik yang dapat menggerakkan motor di kedua arah. H-bridge digunakan dalam berbagai aplikasi. Salah satu aplikasi yang paling umum adalah untuk mengendalikan motor pada robot. Disebut H-bridge karena menggunakan empat transistor yang dihubungkan sedemikian rupa sehingga diagram skematiknya tampak seperti huruf "H".

Kami akan menggunakan IC H-Bridge L298 di sini. L298 dapat mengontrol kecepatan dan arah motor DC dan motor stepper, serta dapat mengontrol dua motor secara bersamaan. Peringkatnya saat ini adalah 2A untuk setiap motor. Namun pada arus ini, Anda perlu menggunakan heat sink.



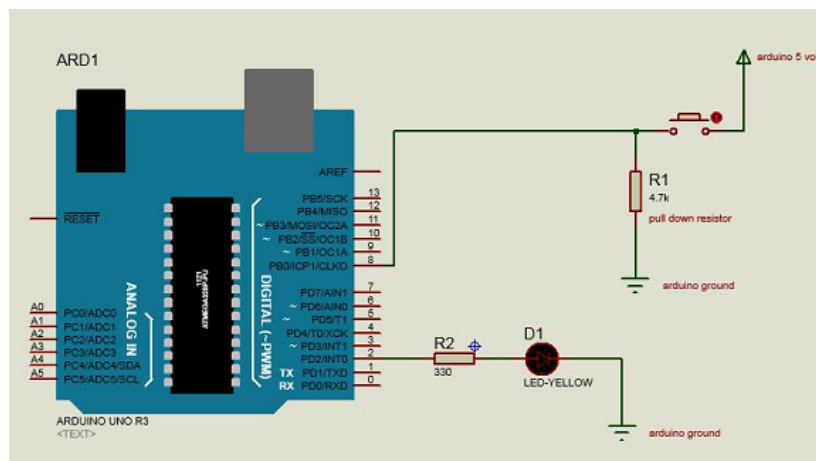
Gambar 6. 5 IC H-Bridge L298

Komponen yang Diperlukan:

- 1 × IC jembatan L298
- 1 × motor DC
- 1 × Arduino UNO
- 1 × *Breadboard*
- 10 × kabel jumper

Prosedur:

Berikut ini adalah diagram skematik antarmuka motor DC ke papan Arduino Uno.



Gambar 6. 6 Rangkaian antarmuka motor DC ke Arduino

Diagram di atas menunjukkan cara menghubungkan IC L298 untuk mengendalikan dua motor. Ada tiga pin input untuk setiap motor, Input1 (IN1), Input2 (IN2), dan Enable1 (EN1) untuk Motor1 dan Input3, Input4, dan Enable2 untuk Motor2.

Karena kita hanya akan mengendalikan satu motor dalam contoh ini, kita akan menghubungkan Arduino ke IN1 (pin 5), IN2 (pin 7), dan Enable1 (pin 6) dari IC

L298. Pin 5 dan 7 bersifat digital, yaitu input ON atau OFF, sedangkan pin 6 memerlukan sinyal termodulasi lebar pulsa (PWM) untuk mengontrol kecepatan motor.

Tabel berikut menunjukkan arah putaran motor berdasarkan nilai digital IN1 dan IN2.

Tabel 6. 1 Arah putaran motor berdasarkan nilai IN1 dan IN2

IN 1	IN2	Perilaku Motorik
0	0	REM
1	0	MAJU
0	1	KE BELAKANG
1	1	REM

Pin IN1 IC L298 dihubungkan ke pin 8 Arduino sedangkan IN2 dihubungkan ke pin 9. Kedua pin digital Arduino ini mengontrol arah motor. Pin EN A IC terhubung ke pin PWM 2 Arduino. Ini akan mengontrol kecepatan motor.

Untuk mengatur nilai pin Arduino 8 dan 9, kita menggunakan fungsi digitalWrite(), dan untuk mengatur nilai pin 2, kita harus menggunakan fungsi analogWrite().

Langkah-langkah Koneksi:

- Hubungkan 5V dan ground IC ke 5V dan ground Arduino secara berurutan.
- Hubungkan motor ke pin 2 dan 3 IC.
- Hubungkan IN1 IC ke pin 8 Arduino.
- Hubungkan IN2 IC ke pin 9 Arduino.
- Hubungkan EN1 IC ke pin 2 Arduino.
- Hubungkan SENS Sebuah pin IC ke ground.
- Hubungkan Arduino menggunakan kabel USB Arduino dan upload program ke Arduino menggunakan software Arduino IDE.
- Berikan daya ke papan Arduino menggunakan catu daya, baterai, atau kabel USB.

Program Arduino:

```
const int pwm = 2 ; //Menginisialisasi Pin 2 sebagai PWM
const int in_1 = 8 ;
const int in_2 = 9 ;
//Untuk memberikan logika ke IC L298 untuk memilih arah motor DC
void setup() {
    pinMode(pwm,OUTPUT) ; //kita harus mengatur pin PWM sebagai output
```

```

    pinMode(in_1,OUTPUT) ; //Pin logika juga ditetapkan sebagai output
    pinMode(in_2,OUTPUT) ;
}
void loop() {
    //For Clock wise motion , in_1 = High , in_2 = Low
    digitalWrite(in_1,HIGH) ;
    digitalWrite(in_2,LOW) ;
    analogWrite(pwm,255) ;
    /* Mengatur PWM motor ke 255 kita dapat mengubah kecepatan rotasi
    Dengan mengubah input PWM tetapi kami hanya menggunakan Arduino
    sehingga kami menggunakan tertinggi
    nilai untuk pengemudi motor */
    //Searah jarum jam selama 3 detik
    delay(3000) ;
    //Untuk rem
    digitalWrite(in_1,HIGH) ;
    digitalWrite(in_2,HIGH) ;
    delay(1000) ;
    //Untuk gerakan Anti Clock-wise - IN_1 = LOW , IN_2 = HIGH
    digitalWrite(in_1,LOW) ;
    digitalWrite(in_2,HIGH) ;
    delay(3000) ;
    //Untuk rem
    digitalWrite(in_1,HIGH) ;
    digitalWrite(in_2,HIGH) ;
    delay(1000) ;
}

```

Hasil:

Motor akan berjalan mula-mula searah jarum jam (CW) selama 3 detik dan kemudian berlawanan arah jarum jam (CCW) selama 3 detik.

B. Motor Servo

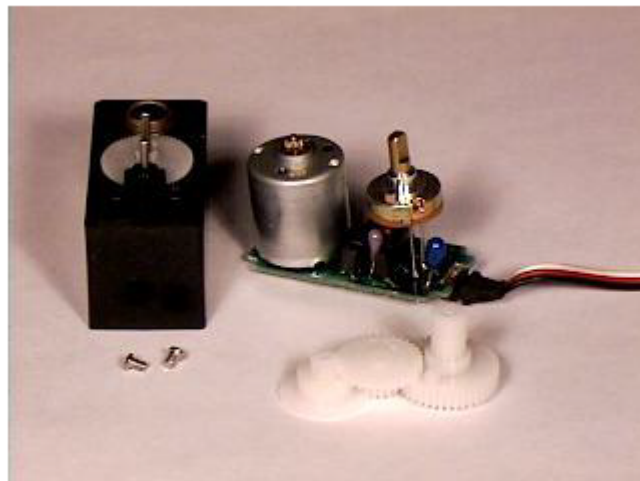
Motor Servo adalah perangkat kecil yang memiliki poros keluaran. Poros ini dapat diposisikan pada posisi sudut tertentu dengan mengirimkan sinyal berkode ke servo. Selama sinyal berkode tersebut ada pada jalur masukan, servo akan mempertahankan posisi sudut poros. Jika sinyal berkode berubah, posisi sudut poros pun berubah. Dalam praktiknya, servo digunakan dalam pesawat yang dikendalikan radio untuk memposisikan permukaan kendali seperti elevator dan kemudi. Servo juga digunakan dalam mobil yang dikendalikan radio, boneka, dan tentu saja, robot.



Gambar 6. 7 Servo

Servo sangat berguna dalam robotika. Motornya kecil, memiliki sirkuit kontrol internal, dan sangat bertenaga untuk ukurannya. Servo standar seperti Futaba S-148 memiliki torsi 42 oz/inci, yang cukup kuat untuk ukurannya. Ia juga menarik daya yang sebanding dengan beban mekanis. Oleh karena itu, servo dengan beban ringan tidak mengonsumsi banyak energi.

Bagian dalam motor servo ditunjukkan pada gambar berikut. Anda dapat melihat rangkaian kontrol, motor, satu set roda gigi, dan casing. Anda juga dapat melihat 3 kabel yang terhubung ke dunia luar. Satu untuk daya (+5 volt), ground, dan kabel putih adalah kabel kontrol.



Gambar 6. 8 Bagian dalam servo

Cara Kerja Motor Servo

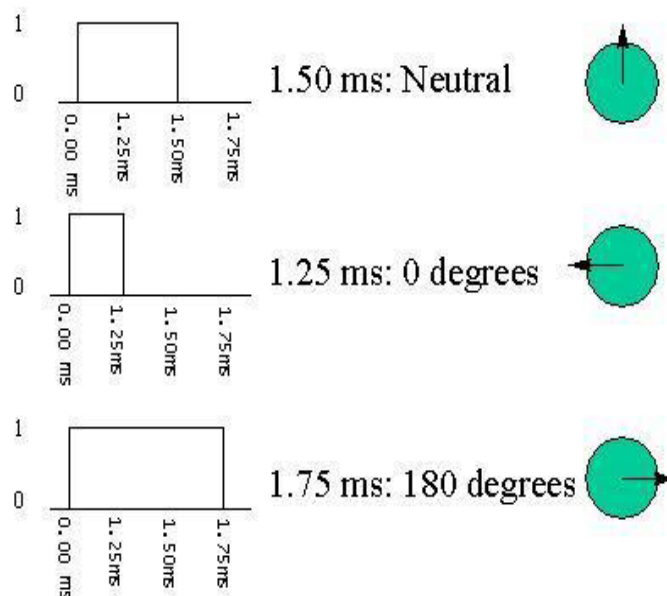
Motor servo memiliki beberapa sirkuit kontrol dan potensiometer (resistor variabel, alias pot) yang terhubung ke poros output. Pada gambar di atas, pot dapat dilihat di sisi kanan papan sirkuit. Pot ini memungkinkan sirkuit kontrol untuk memantau sudut arus motor servo.

Jika poros berada pada sudut yang benar, maka motor akan mati. Jika rangkaian menemukan bahwa sudutnya tidak tepat, maka motor akan berputar hingga mencapai sudut yang diinginkan. Poros keluaran servo mampu bergerak sekitar 180 derajat. Biasanya suhunya berada pada kisaran 210 derajat, namun bervariasi tergantung pabrikannya. Servo normal digunakan untuk mengontrol gerakan sudut 0 hingga 180 derajat. Secara mekanis ia tidak mampu berputar lebih jauh karena penghentian mekanis yang terpasang pada gigi keluaran utama.

Daya yang diberikan pada motor sebanding dengan jarak yang harus ditempuhnya. Jadi, jika poros perlu berputar dalam jarak yang jauh, motor akan berjalan dengan kecepatan penuh. Jika hanya perlu berputar dalam jumlah kecil, motor akan berjalan dengan kecepatan yang lebih lambat. Ini disebut **kontrol proporsional**.

Bagaimana Anda Mengomunikasikan Sudut Putaran Servo?

Kabel kontrol digunakan untuk mengomunikasikan sudut. Sudut ditentukan oleh durasi pulsa yang diterapkan pada kabel kontrol. Ini disebut **Modulasi Kode Pulsa**. Servo mengharapkan untuk melihat pulsa setiap 20 milidetik (0,02 detik). Panjang pulsa akan menentukan seberapa jauh motor berputar. Pulsa 1,5 milidetik, misalnya, akan membuat motor berputar ke posisi 90 derajat (sering disebut sebagai posisi netral). Jika pulsa lebih pendek dari 1,5 milidetik, maka motor akan memutar poros lebih dekat ke 0 derajat. Jika pulsa lebih panjang dari 1,5 milidetik, poros berputar lebih dekat ke 180 derajat.



Gambar 6. 9 Penggunaan sudut putaran pada servo

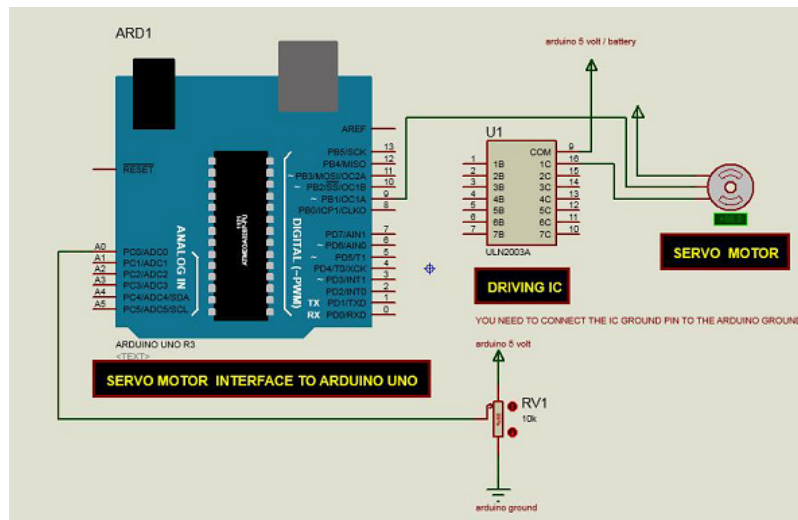
Komponen yang Diperlukan:

- 1 × Papan Arduino UNO

- 1 × Motor Servo
- 1 × IC penggerak ULN2003
- Resistor 1 × 10 K Ω

Prosedur:

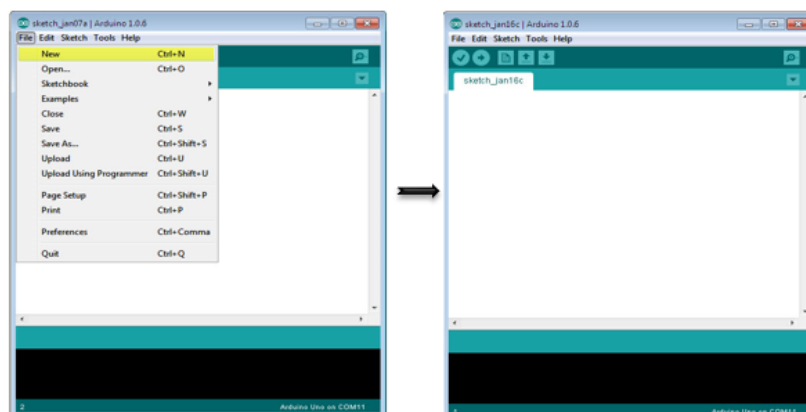
Ikuti diagram sirkuit dan buat koneksi seperti yang ditunjukkan pada gambar di bawah ini.



Gambar 6. 10 Rangkaian pengkabelan proyek servo

Sketch:

Buka perangkat lunak Arduino IDE di komputer Anda. Pengkodean dalam bahasa Arduino akan mengontrol sirkuit Anda. Buka File sketsa baru dengan mengklik Baru. Seperti pada Gambar 6.11.



Gambar 6. 11 Membuat sketch baru di Arduino IDE

Program Arduino:

```
// Mengontrol posisi servo menggunakan potensiometer (resistor variabel)
```

```
#include <Servo.h>

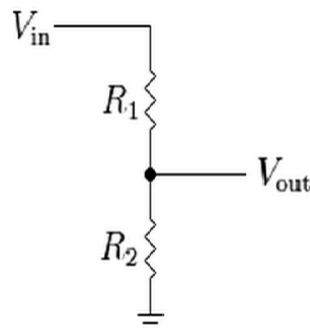
Servo myservo; // Membuat objek servo untuk mengontrol servo
int potpin = 0; // Pin analog yang digunakan untuk menghubungkan
potensiometer

int val; // variabel untuk membaca nilai dari pin analog
void setup() {
  myservo.attach(9); // Memasang servo pada pin 9 ke objek servo
}
void loop() {
  val = analogRead(potpin);
  // membaca nilai potensiometer (nilai antara 0 dan 1023)
  val = map(val, 0, 1023, 0, 180);
  // skala untuk menggunakannya dengan servo (nilai antara 0 dan 180)
  myservo.write(val); // sets the servo position according to the scaled
value
  delay(15);
}
```

Motor servo memiliki tiga terminal - daya, ground, dan sinyal. Kabel listrik biasanya berwarna merah, dan harus dihubungkan ke pin 5V di Arduino. Kabel ground biasanya berwarna hitam atau coklat dan harus dihubungkan ke satu terminal IC ULN2003 (10 -16). Untuk melindungi papan Arduino Anda dari kerusakan, Anda memerlukan beberapa IC driver untuk melakukan itu. Di sini kita telah menggunakan IC ULN2003 untuk menggerakkan motor servo. Pin sinyal biasanya berwarna kuning atau oranye dan harus dihubungkan ke pin Arduino nomor 9.

Menghubungkan Potensiometer

Pembagi tegangan/pembagi potensial adalah resistor dalam rangkaian seri yang menskalakan tegangan keluaran dengan rasio tertentu dari tegangan masukan yang diterapkan. Berikut adalah diagram rangkaian seperti pada Gambar 6.12.



Gambar 6. 12 Rangkaian seri resistor

$$V_{\text{keluar}} = (V_{\text{masuk}} \times R_2) / (R_1 + R_2)$$

V_{out} adalah potensi keluaran, yang bergantung pada tegangan masukan yang diberikan (V_{in}) dan resistor (R_1 dan R_2) dalam rangkaian. Ini berarti bahwa arus yang mengalir melalui R_1 juga akan mengalir melalui R_2 tanpa terbagi. Dalam persamaan di atas, saat nilai R_2 berubah, V_{out} berskala sesuai dengan tegangan masukan, V_{in} .

Biasanya, potensiometer adalah pembagi potensial, yang dapat menskalakan tegangan keluaran rangkaian berdasarkan nilai resistor variabel, yang diskalakan menggunakan kenop. Ini memiliki tiga pin: GND, Signal, dan +5V seperti yang ditunjukkan pada Gambar 6.13.



Gambar 6. 13 Pin pada potensimeter

Hasil:

Dengan mengubah posisi NOP pot, motor servo akan mengubah sudutnya.

C. Motor Stepper

Motor Stepper atau motor stepper adalah motor sinkron tanpa sikat yang membagi putaran penuh menjadi beberapa langkah. Tidak seperti motor DC tanpa sikat yang berputar terus-menerus saat tegangan DC tetap diberikan padanya, motor stepper berputar dalam sudut langkah yang terpisah.

Motor Stepper diproduksi dengan langkah per putaran sebesar 12, 24, 72, 144, 180, dan 200, sehingga menghasilkan sudut langkah sebesar 30, 15, 5, 2,5, 2, dan 1,8 derajat per langkah. Motor stepper dapat dikontrol dengan atau tanpa umpan balik.

Bayangkan sebuah motor pada pesawat RC. Motor berputar sangat cepat ke satu arah atau arah lainnya. Anda dapat mengubah kecepatannya dengan jumlah daya yang diberikan ke motor, tetapi Anda tidak dapat memerintahkan baling-baling untuk berhenti pada posisi tertentu.

Sekarang bayangkan sebuah printer. Ada banyak bagian yang bergerak di dalam printer, termasuk motor. Salah satu motor tersebut bertindak sebagai pengumpan kertas, memutar rol yang menggerakkan selebar kertas saat tinta sedang dicetak di atasnya. Motor ini harus mampu menggerakkan kertas dengan jarak yang tepat agar dapat mencetak baris teks berikutnya atau baris gambar berikutnya.

Ada motor lain yang dipasang pada batang berulir yang menggerakkan kepala cetak maju mundur. Sekali lagi, batang berulir itu perlu dipindahkan dengan jumlah yang tepat untuk mencetak huruf demi huruf. Di sinilah motor stepper berguna.



Gambar 6. 14 Motor stepper

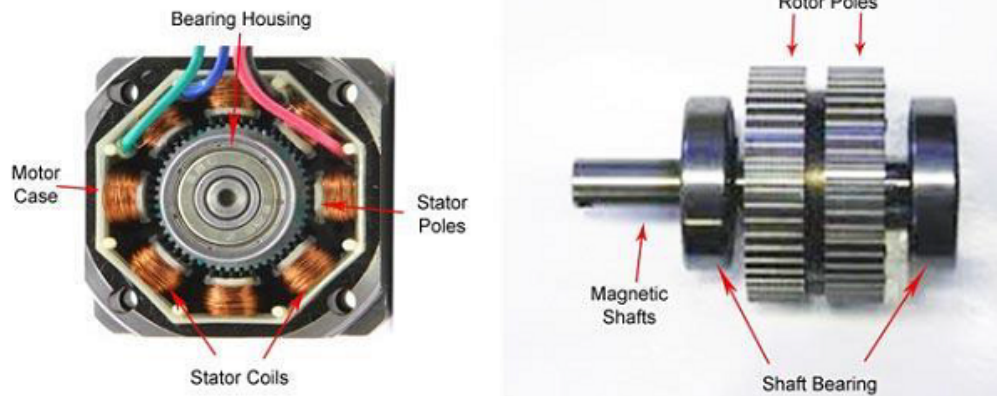
Bagaimana Motor Stepper Bekerja?

Motor DC biasa berputar hanya dalam satu arah sedangkan motor Stepper dapat berputar secara bertahap.

Motor stepper dapat berputar dalam jumlah derajat (atau langkah) yang tepat sesuai keinginan. Ini memberi Anda kendali penuh atas motor, yang memungkinkan Anda untuk memindahkannya ke lokasi yang tepat dan mempertahankan posisi tersebut. Motor ini melakukannya dengan memberi daya pada kumparan di dalam motor untuk periode waktu yang sangat singkat. Kerugiannya adalah Anda harus memberi daya pada motor sepanjang waktu untuk mempertahankannya pada posisi yang Anda inginkan.

Yang perlu Anda ketahui saat ini adalah, untuk menggerakkan motor stepper, Anda memerintahkannya untuk menggerakkan sejumlah langkah tertentu ke satu arah atau yang lain, dan memberi tahu kecepatan langkah yang harus diambil ke arah tersebut. Ada banyak jenis motor stepper. Metode yang dijelaskan di sini dapat digunakan untuk menyimpulkan cara menggunakan motor dan driver lain yang tidak disebutkan dalam tutorial ini. Namun, Anda selalu disarankan untuk

membaca lembar data dan panduan motor dan driver khusus untuk model yang Anda miliki.



Gambar 6. 15 Bagian dalam pada motor stepper

Komponen yang Diperlukan:

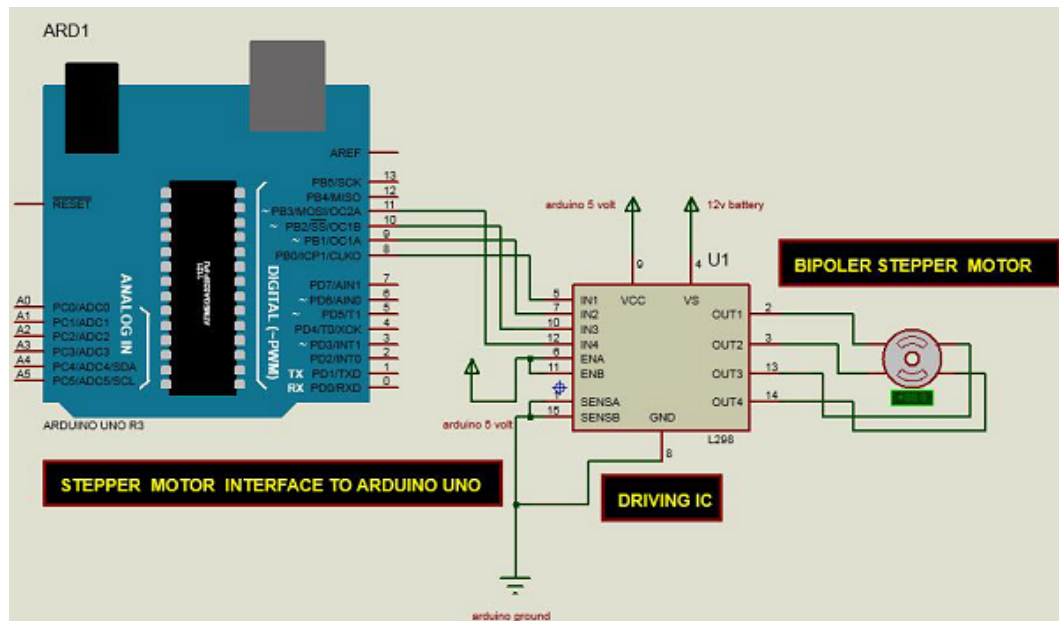
- 1 × papan Arduino UNO
- 1 × Motor stepper bipolar kecil seperti yang ditunjukkan pada gambar di bawah ini
- 1 × IC penggerak LM298



Gambar 6. 16 Motor stepper bipolar kecil

Prosedur:

Ikuti diagram rangkaian dan buat sambungan seperti ditunjukkan pada gambar di bawah ini.



Gambar 6. 17 Rangkaian proyek motor stepper

Sketch:

Buka perangkat lunak Arduino IDE di komputer Anda. Pengkodean dalam bahasa Arduino akan mengontrol sirkuit Anda. Buka File sketsa baru dengan mengklik Baru. Seperti terlihat pada Gambar 6.11.

Program Arduino:

```
/* Stepper Motor Control */
#include <Stepper.h>
const int stepsPerRevolution = 90;
// Ubah ini agar sesuai dengan jumlah langkah per revolusi
// untuk motor Anda
// Menginisialisasi pustaka stepper pada pin 8 hingga 11:
Stepper myStepper(stepsPerRevolution, 8, 9, 10, 11);
void setup() {
  // Atur kecepatan pada 60 rpm:
  myStepper.setSpeed(5);
  // Menginisialisasi port serial:
  Serial.begin(9600);
}
void loop() {
  // Langkah Satu Revolusi dalam Satu Arah:
  Serial.println("clockwise");
  myStepper.step(stepsPerRevolution);
  delay(500);
}
```

```
// Langkah Satu Revolusi ke Arah Lain:  
Serial.println("counterclockwise");  
myStepper.step(-stepsPerRevolution);  
delay(500);  
}
```

Program ini menggerakkan motor stepper unipolar atau bipolar. Motor tersebut terpasang pada pin digital 8 - 11 Arduino.

Hasil:

Motor akan berputar satu kali ke satu arah, lalu satu kali putaran ke arah yang lain.

BAB 7

PROYEK LANJUTAN ARDUINO

A. Perpustakaan Nada

Dalam bab ini, kita akan menggunakan Arduino Tone Library. Ini tidak lain adalah Perpustakaan Arduino, yang menghasilkan gelombang persegi dengan frekuensi tertentu (dan siklus kerja 50%) pada pin Arduino mana pun. Durasi opsional dapat ditentukan, jika tidak, gelombang akan berlanjut hingga fungsi `stop()` dipanggil. Pin dapat dihubungkan ke piezo buzzer atau speaker untuk memutar nada.

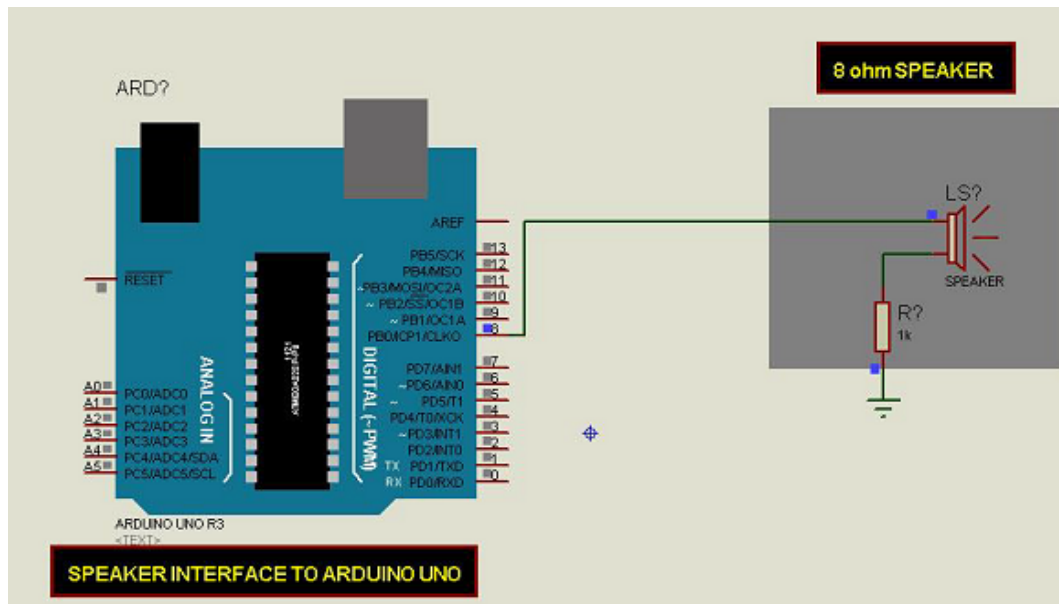
Peringatan – Jangan hubungkan pin langsung ke input audio apa pun. Tegangannya jauh lebih tinggi daripada tegangan level saluran standar, dan dapat merusak input kartu suara, dsb. Anda dapat menggunakan pembagi tegangan untuk menurunkan tegangan.

Komponen yang Diperlukan:

- Speaker 1×8 ohm
- Resistor $1 \times 1k$
- $1 \times$ Papan Arduino UNO

Prosedur:

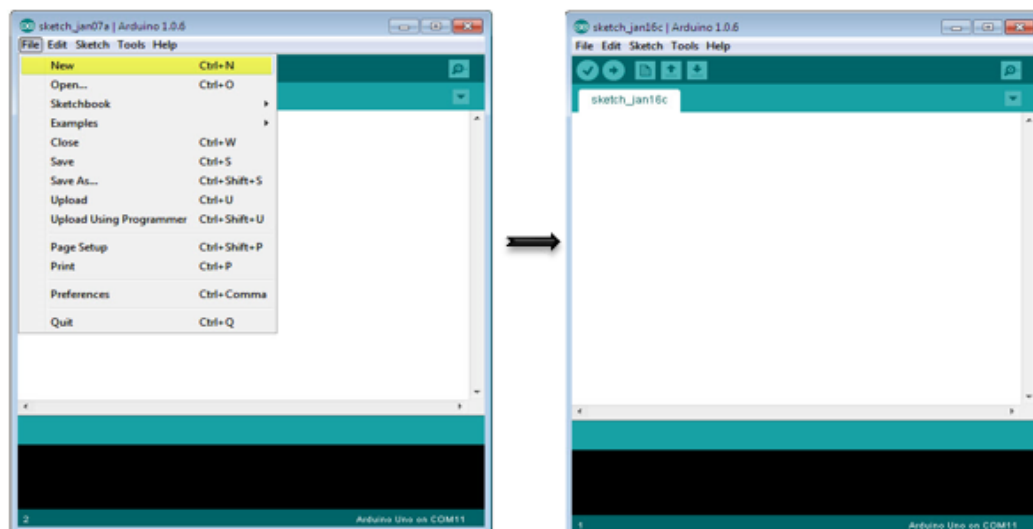
Ikuti diagram sirkuit dan buat koneksi seperti yang ditunjukkan pada gambar di bawah ini.



Gambar 7. 1 Rangkaian proyek nada di arduino

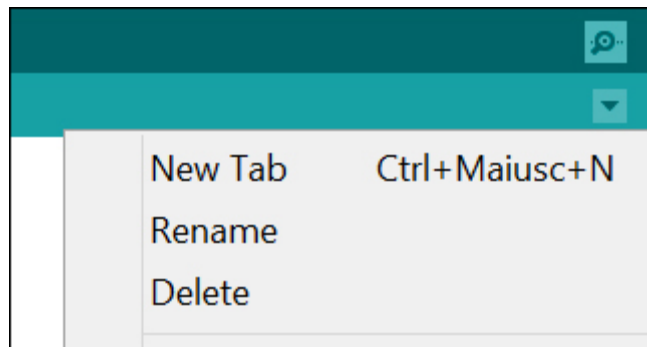
Sketch:

Buka perangkat lunak Arduino IDE di komputer Anda. Pengodean dalam bahasa Arduino akan mengendalikan sirkuit Anda. Buka File sketsa baru dengan mengeklik Baru. Seperti pada Gambar 7.2.



Gambar 7. 2 Membuat sketch baru di Arduino IDE

Untuk membuat file pitches.h, klik tombol tepat di bawah ikon monitor serial dan pilih "Tab Baru", atau gunakan Ctrl+Shift+N.



Gambar 7. 3 Menambahkan tab baru di Arduino IDE

Kemudian tempel kode berikut

```

/*****
 * Public Constants
 *****/

#define NOTE_B0 31
#define NOTE_C1 33
#define NOTE_CS1 35
#define NOTE_D1 37
#define NOTE_DS1 39
#define NOTE_E1 41
#define NOTE_F1 44
#define NOTE_FS1 46
#define NOTE_G1 49
#define NOTE_GS1 52
#define NOTE_A1 55
#define NOTE_AS1 58
#define NOTE_B1 62
#define NOTE_C2 65
#define NOTE_CS2 69
#define NOTE_D2 73
#define NOTE_DS2 78
#define NOTE_E2 82
#define NOTE_F2 87
#define NOTE_FS2 93
#define NOTE_G2 98
#define NOTE_GS2 104
#define NOTE_A2 110
#define NOTE_AS2 117

```

```
#define NOTE_B2 123
#define NOTE_C3 131
#define NOTE_CS3 139
#define NOTE_D3 147
#define NOTE_DS3 156
#define NOTE_E3 165
#define NOTE_F3 175
#define NOTE_FS3 185
#define NOTE_G3 196
#define NOTE_GS3 208
#define NOTE_A3 220
#define NOTE_AS3 233
#define NOTE_B3 247
#define NOTE_C4 262
#define NOTE_CS4 277
#define NOTE_D4 294
#define NOTE_DS4 311
#define NOTE_E4 330
#define NOTE_F4 349
#define NOTE_FS4 370
#define NOTE_G4 392
#define NOTE_GS4 415
#define NOTE_A4 440
#define NOTE_AS4 466
#define NOTE_B4 494
#define NOTE_C5 523
#define NOTE_CS5 554
#define NOTE_D5 587
#define NOTE_DS5 622
#define NOTE_E5 659
#define NOTE_F5 698
#define NOTE_FS5 740
#define NOTE_G5 784
#define NOTE_GS5 831
#define NOTE_A5 880
#define NOTE_AS5 932
#define NOTE_B5 988
#define NOTE_C6 1047
#define NOTE_CS6 1109
#define NOTE_D6 1175
```

```

#define NOTE_DS6 1245
#define NOTE_E6 1319
#define NOTE_F6 1397
#define NOTE_FS6 1480
#define NOTE_G6 1568
#define NOTE_GS6 1661
#define NOTE_A6 1760
#define NOTE_AS6 1865
#define NOTE_B6 1976
#define NOTE_C7 2093
#define NOTE_CS7 2217
#define NOTE_D7 2349
#define NOTE_DS7 2489
#define NOTE_E7 2637
#define NOTE_F7 2794
#define NOTE_FS7 2960
#define NOTE_G7 3136
#define NOTE_GS7 3322
#define NOTE_A7 3520
#define NOTE_AS7 3729
#define NOTE_B7 3951
#define NOTE_C8 4186
#define NOTE_CS8 4435
#define NOTE_D8 4699
#define NOTE_DS8 4978

```

Simpan kode yang diberikan di atas sebagai **pitches.h**

Program Arduino:

```

#include "pitches.h"
// Catatan dalam melodi:
int melody[] = {
NOTE_C4, NOTE_G3,NOTE_G3, NOTE_GS3, NOTE_G3,0, NOTE_B3, NOTE_C4};
// Durasi nada: 4 = nada seperempat, 8 = nada kedelapan, dll.:
int noteDurations[] = {
    4, 8, 8, 4,4,4,4,4
};

void setup() {
    // Iterasi atas nada melodi:
    for (int thisNote = 0; thisNote < 8; thisNote++) {

```

```

        // Untuk menghitung durasi not, luangkan satu detik
        // dibagi dengan jenis catatan.
        //e.g. quarter note = 1000 / 4, eighth note = 1000/8, etc.
        int noteDuration = 1000/noteDurations[thisNote];
        tone(8, melody[thisNote],noteDuration);
        //Jeda untuk durasi catatan ditambah 30 milidetik:
        delay(noteDuration +30);
    }
}

void loop() {
    // Tidak perlu mengulang melodi.
}

```

Kode tersebut menggunakan file tambahan, `pitches.h`. File ini berisi semua nilai pitch untuk not-not umum. Misalnya, `NOTE_C4` adalah C tengah. `NOTE_FS4` adalah F tajam, dan seterusnya. Tabel not ini awalnya ditulis oleh Brett Hagman, yang karyanya menjadi dasar perintah `tone()`. Anda mungkin merasa ini berguna setiap kali ingin membuat not musik.

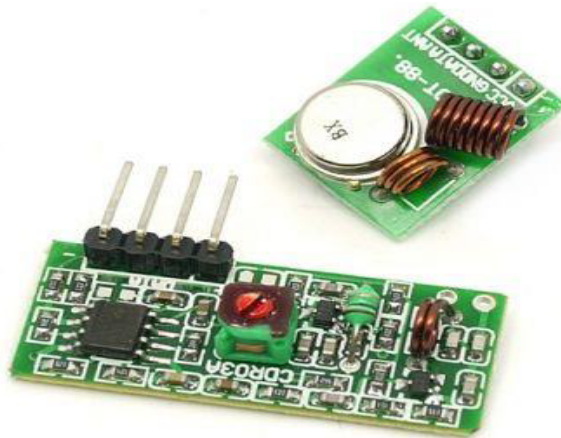
Hasil:

Anda akan mendengar not musik yang disimpan di `pitches.h`. mengajukan.

B. Komunikasi Nirkabel

Modul pemancar dan penerima nirkabel bekerja pada 315 Mhz. Mereka dapat dengan mudah masuk ke dalam *Breadboard* dan bekerja dengan baik dengan mikrokontroler untuk membuat tautan data nirkabel yang sangat sederhana. Dengan sepasang pemancar dan penerima, modul hanya akan berfungsi mengkomunikasikan data satu arah, namun Anda memerlukan dua pasang (frekuensi berbeda) untuk bertindak sebagai pasangan pemancar/penerima.

Catatan – Modul-modul ini tidak pandang bulu dan menerima cukup banyak gangguan. Baik pemancar maupun penerima bekerja pada frekuensi umum dan tidak memiliki ID.



Gambar 7. 4 Modul penerima dan pengirim

Spesifikasi Modul Penerima

- Model Produk - MX-05V
- Tegangan operasi – DC5V
- Arus Diam – 4mA
- Frekuensi penerimaan – 315Mhz
- Sensitivitas penerima – -105DB
- Ukuran – 30*14*7mm

Spesifikasi Modul Pemancar

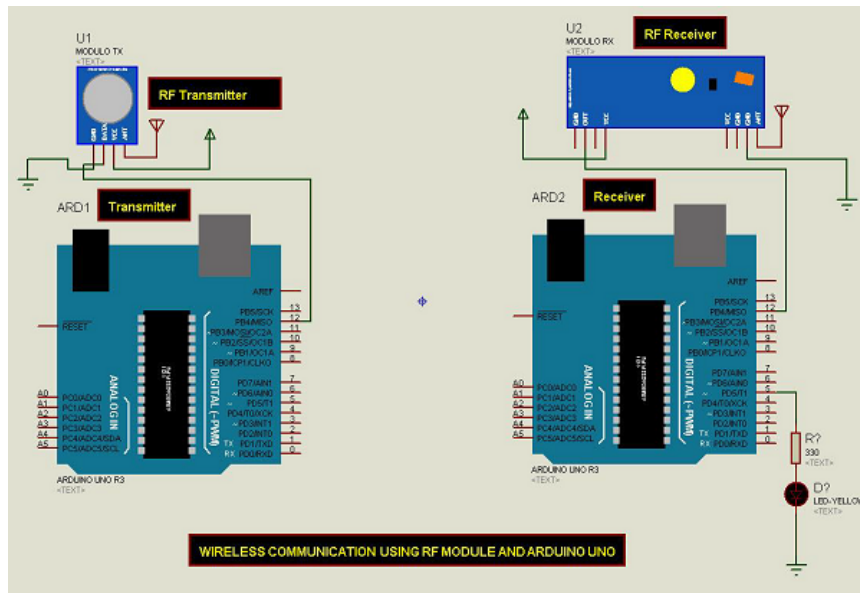
- Model Produk - MX-FS-03V
- Jarak peluncuran – 20-200 meter (voltase berbeda, hasil berbeda)
- Tegangan pengoperasian – 3,5-12V
- Dimensi – 19 * 19mm
- Mode pengoperasian - SAYA
- Kecepatan transfer – 4KB / S
- Transmisi daya – 10mW
- Frekuensi transmisi – 315Mhz
- Antena eksternal – saluran multi-inti atau inti tunggal biasa sepanjang 25cm
- Pinout dari kiri → kanan – (DATA; V_{CC} ; GND)

Komponen yang Diperlukan:

- 2 × papan Arduino UNO
- 1 × Pemancar tautan Rf
- 1 × Penerima tautan Rf

Prosedur:

Ikuti diagram rangkaian dan buat sambungan seperti ditunjukkan pada gambar di bawah ini.

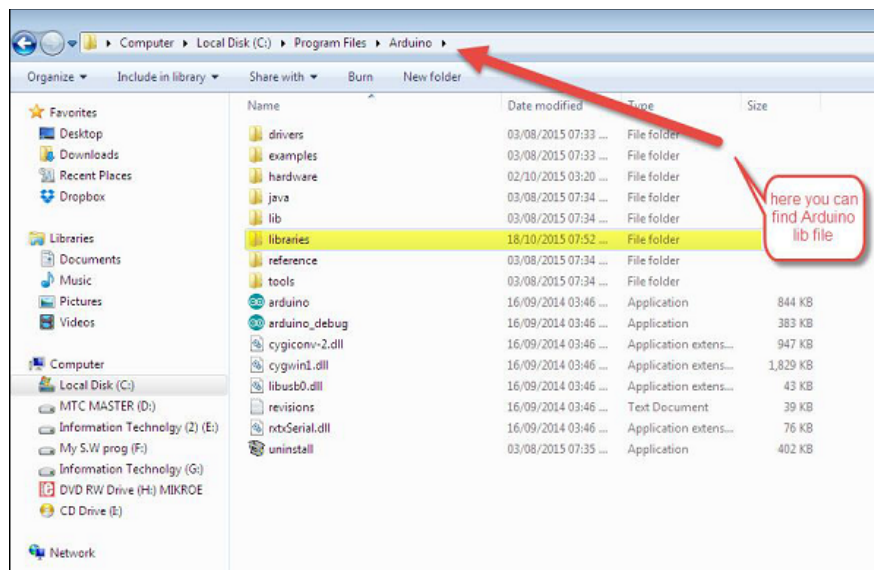


Gambar 7. 5 Rangkaian untuk komunikasi nirkabel

Sketch:

Buka perangkat lunak Arduino IDE di komputer Anda. Pengodean dalam bahasa Arduino akan mengendalikan sirkuit Anda. Buka File sketsa baru dengan mengeklik Baru. Seperti pada Gambar 7.2.

Catatan- Anda harus menyertakan perpustakaan keypad di file perpustakaan Arduino Anda. Salin dan tempel file VirtualWire.lib di folder perpustakaan seperti yang disorot pada gambar di bawah.



Gambar 7. 6 Lokasi folder *library* pada Arduino

Program Arduino untuk pemancar:

```
//Tx sederhana pada pin D12
#include <VirtualWire.h>
char *controller;

void setup() {
    pinMode(13,OUTPUT);
    vw_set_ptt_inverted(true);
    vw_set_tx_pin(12);
    vw_setup(4000); // kecepatan transfer data Kbps
}

void loop() {
    controller="1" ;
    vw_send((uint8_t *)controller, strlen(controller));
    vw_wait_tx(); // Tunggu hingga seluruh pesan hilang
    digitalWrite(13,1);
    delay(2000);
    controller="0" ;
    vw_send((uint8_t *)controller, strlen(controller));
    vw_wait_tx(); // Tunggu hingga seluruh pesan hilang
    digitalWrite(13,0);
    delay(2000);
}
```

Ini adalah kode sederhana. Pertama, ia akan mengirim karakter '1' dan setelah dua detik ia akan mengirim karakter '0' dan seterusnya.

Program Arduino untuk penerima:

```
//simple Rx on pin D12
#include <VirtualWire.h>

void setup() {
    vw_set_ptt_inverted(true); // Diperlukan untuk DR3100
    vw_set_rx_pin(12);
    vw_setup(4000); // Bit per detik
    pinMode(5, OUTPUT);
    vw_rx_start(); // Mulai PLL receiver berjalan
}

void loop() {
```

```

uint8_t buf[VW_MAX_MESSAGE_LEN];
uint8_t buflen = VW_MAX_MESSAGE_LEN;
if (vw_get_message(buf, &buflen)) // Non-blocking {
    if(buf[0]=='1') {
        digitalWrite(5,1);
    }
    if(buf[0]=='0') {
        digitalWrite(5,0);
    }
}
}

```

LED yang terhubung ke pin nomor 5 pada papan Arduino menyala saat karakter '1' diterima dan mati saat karakter '0' diterima.

C. Komunikasi Jaringan

Modul WiFi CC3000 dari Texas Instruments adalah paket perak kecil, yang akhirnya menghadirkan fungsionalitas WiFi yang mudah digunakan dan terjangkau untuk proyek Arduino Anda.

Ia menggunakan SPI untuk komunikasi (bukan UART!) sehingga Anda dapat mengirimkan data secepat yang Anda inginkan atau selambat yang Anda inginkan. Ia memiliki sistem interupsi yang tepat dengan pin IRQ sehingga Anda dapat memiliki koneksi asinkron. Mendukung keamanan 802.11b/g, terbuka/WEP/WPA/WPA2, TKIP & AES. Tumpukan TCP/IP bawaan dengan antarmuka "socket BSD" mendukung TCP dan UDP dalam mode klien dan server.



Gambar 7. 7 Modul adafruit cc3000

Komponen yang Diperlukan:

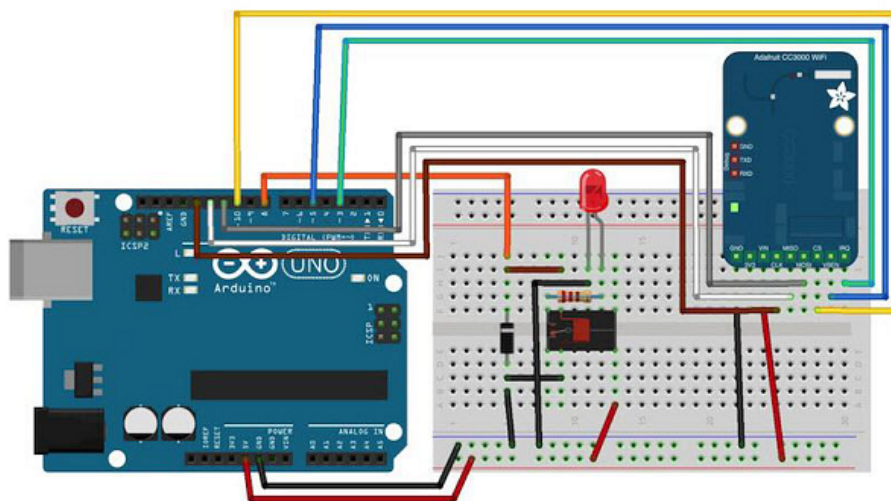
- 1 × Arduino Uno
- 1 × papan breakout Adafruit CC3000
- 1 × relai 5V

- 1 × Dioda penyearah
- 1×LED
- 1 × 220 Ohm resistor
- 1 × *Breadboard* dan beberapa kabel jumper

Untuk proyek ini, Anda hanya memerlukan Arduino IDE biasa, pustaka CC3000 milik Adafruit, dan pustaka CC3000 MDNS. Kita juga akan menggunakan pustaka aREST untuk mengirim perintah ke relai melalui WiFi.

Prosedur:

Ikuti diagram rangkaian dan buat sambungan seperti ditunjukkan pada gambar di bawah ini.



Gambar 7. 8 Rangkaian proyek komunikasi jaringan

Konfigurasi perangkat keras untuk proyek ini sangat mudah.

- Hubungkan pin IRQ papan CC3000 ke pin nomor 3 papan Arduino.
- VBAT ke pin 5, dan CS ke pin 10.
- Hubungkan pin SPI ke papan Arduino: MOSI, MISO, dan CLK ke pin 11, 12, dan 13.
- V terhubung ke Arduino 5V, dan GND ke GND.

Sekarang mari kita hubungkan relai.

Setelah menempatkan relai pada *Breadboard*, Anda dapat mulai mengidentifikasi dua bagian penting pada relai Anda: bagian kumparan yang memerintahkan relai, dan bagian sakelar tempat kita akan memasang LED.

- Pertama, hubungkan pin nomor 8 papan Arduino ke salah satu pin kumparan.

- Hubungkan pin lainnya ke ground papan Arduino.

Anda juga harus menempatkan dioda penyearah (anoda yang terhubung ke pin ground) di atas pin kumparan untuk melindungi sirkuit saat relai sedang menyala.

- Hubungkan +5V papan Arduino ke pin umum sakelar relai.
- Terakhir, hubungkan salah satu pin sakelar lainnya (biasanya, pin yang tidak terhubung saat relai mati) ke LED secara seri dengan resistor 220 Ohm, dan hubungkan sisi lain LED ke ground papan Arduino.

Menguji Komponen Individu

```
const int relay_pin = 8; // Relay pin

void setup() {
  Serial.begin(9600);
  pinMode(relay_pin, OUTPUT);
}

void loop() {
  // Aktifkan relai
  digitalWrite(relay_pin, HIGH);
  // Tunggu 1 detik
  delay(1000);
  // Menonaktifkan relai
  digitalWrite(relay_pin, LOW);
  // Tunggu 1 detik
  delay(1000);
}
```

Kodenya cukup jelas. Anda tinggal mengunggahnya ke papan dan relai akan berganti status setiap detik, dan LED akan menyala dan mati sesuai dengan keadaan.

Menambahkan Konektivitas WiFi

Sekarang mari kita mengontrol relai secara nirkabel menggunakan chip WiFi CC3000. Perangkat lunak untuk proyek ini didasarkan pada protokol TCP. Namun, untuk proyek ini, papan Arduino akan menjalankan server web kecil, sehingga kita dapat “mendengarkan” perintah yang datang dari komputer. Pertama-tama kita akan mengurus sketsa Arduino, dan kemudian kita akan melihat cara menulis kode sisi server dan membuat antarmuka yang bagus.

Pertama, sketsa Arduino. Tujuannya di sini adalah untuk terhubung ke jaringan WiFi Anda, membuat server web, memeriksa apakah ada koneksi TCP yang masuk, lalu mengubah status relai sesuai dengan kebutuhan.

Bagian Penting dari Kode

```
#include <Adafruit_CC3000.h>
#include <SPI.h>
#include <CC3000_MDNS.h>
#include <Ethernet.h>
#include <aREST.h>
```

Anda perlu menentukan di dalam kode apa yang spesifik untuk konfigurasi Anda, yaitu nama dan kata sandi Wi-Fi, dan port untuk komunikasi TCP (kami telah menggunakan 80 di sini).

```
// WiFi network (change with your settings!)
#define WLAN_SSID "yourNetwork" // Tidak boleh lebih dari 32 karakter!
#define WLAN_PASS "yourPassword"
#define WLAN_SECURITY WLAN_SEC_WPA2 // Ini bisa WLAN_SEC_UNSEC,
WLAN_SEC_WEP,
// WLAN_SEC_WPA atau WLAN_SEC_WPA2

// Port untuk mendengarkan koneksi TCP masuk
#define LISTEN_PORT 80
```

Kami kemudian dapat membuat instance CC3000, server dan instance aREST

```
// Contoh server
Adafruit_CC3000_Server restServer(LISTEN_PORT); // DNS responder
instance

MDNSResponder mdns; // Membuat instans aREST
aREST rest = aREST();
```

Di bagian setup() dari sketsa, kita sekarang dapat menghubungkan chip CC3000 ke jaringan

```
cc3000.connectToAP(WLAN_SSID, WLAN_PASS, WLAN_SECURITY);
```

Bagaimana komputer mengetahui ke mana harus mengirim data? Salah satu caranya adalah dengan menjalankan sketsa satu kali, kemudian mendapatkan alamat IP board CC3000, dan memodifikasi kode server lagi. Namun, kami dapat melakukan yang lebih baik, dan di situlah perpustakaan CC3000 MDNS berperan. Kami akan menetapkan nama tetap untuk papan CC3000 kami dengan perpustakaan ini, sehingga kami dapat menuliskan nama ini langsung ke dalam kode server.

Ini dilakukan dengan potongan kode berikut

```
if (!mdns.begin("arduino", cc3000)) {
  while(1);
}
```

Kita juga perlu mendengarkan koneksi yang masuk.

```
restServer.begin();
```

Selanjutnya, kita akan mengkodekan fungsi loop() dari sketsa yang akan terus dieksekusi. Pertama-tama kita harus memperbarui server mDNS.

```
mdns.update();
```

Server yang berjalan pada papan Arduino akan menunggu koneksi masuk dan menangani permintaan.

```
Adafruit_CC3000_ClientRef client = restServer.available();  
rest.handle(client);
```

Kini cukup mudah untuk menguji proyek melalui WiFi. Pastikan Anda memperbarui sketsa dengan nama dan kata sandi WiFi Anda sendiri, dan unggah sketsa ke papan Arduino Anda. Buka monitor serial Arduino IDE Anda, dan cari alamat IP papan Anda.

Mari kita asumsikan untuk sisanya di sini bahwa alamatnya adalah seperti 192.168.1.103.

Kemudian, cukup buka browser web favorit Anda, dan ketik –

192.168.1.103/digital/8/1

Anda akan melihat relai Anda otomatis menyala.

Membangun Antarmuka Relai

Kami sekarang akan mengkodekan antarmuka proyek. Akan ada dua bagian di sini: file HTML yang berisi antarmuka, dan file Javascript sisi klien untuk menangani klik pada antarmuka. Antarmuka di sini didasarkan pada proyek **aREST.js**, yang dibuat untuk mengontrol perangkat WiFi dengan mudah dari komputer Anda.

Mari kita lihat dulu file HTMLnya, bernama interface.html. Bagian pertama terdiri dari mengimpor semua perpustakaan yang diperlukan untuk antarmuka

```
<head>  
  <meta charset = utf-8 />  
  <title> Relay Control </title>  
  <link rel = "stylesheet" type = "text/css"  
    href  
    =  
    "https://maxcdn.bootstrapcdn.com/bootstrap/3.3.4/css/bootstrap.min.css"  
  >  
  <link rel="stylesheet" type = "text/css" href = "style.css">  
  <script type = "text/javascript"  
    src = "https://code.jquery.com/jquery-2.1.4.min.js"></script>  
  <script type = "text/javascript"  
    src  
    =  
    "https://cdn.rawgit.com/Foliotek/AjaxQ/master/ajaxq.js"></script>  
  <script type = "text/javascript"
```

```

    src
    "https://cdn.rawgit.com/marcoschwartz/aREST.js/master/aREST.js"></scrip
t>
    <script type = "text/javascript"
        src = "script.js"></script>
</head>

```

Kemudian, kita tentukan dua tombol di dalam antarmuka, satu untuk menyalakan relai, dan yang lainnya untuk mematikannya kembali.

```

<div class = 'container'>
  <h1>Relay Control</h1>
  <div class = 'row'>
    <div class = "col-md-1">Relay</div>
    <div class = "col-md-2">
      <button id = 'on' class = 'btn btn-block btn-
success'>On</button>
    </div>
    <div class = "col-md-2">
      <button id = 'off' class = 'btn btn-block btn-
danger'>On</button>
    </div>
  </div>
</div>

```

Sekarang, kita juga memerlukan berkas Javascript sisi klien untuk menangani klik pada tombol. Kita juga akan membuat perangkat yang akan kita tautkan ke nama mDNS perangkat Arduino kita. Jika Anda mengubah ini dalam kode Arduino, Anda perlu mengubahnya di sini juga.

```

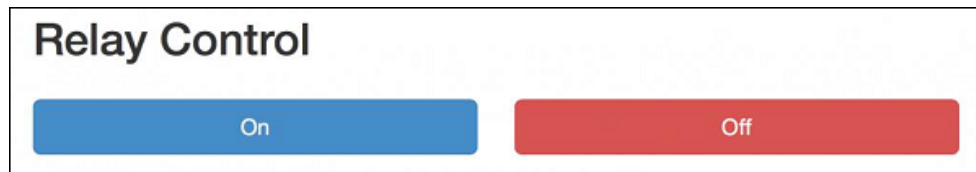
// Buat perangkat
var device = new Device("arduino.local");
// Button

$('#on').click(function() {
  device.digitalWrite(8, 1);
});

$('#off').click(function() {
  device.digitalWrite(8, 0);
});

```

Kode lengkap untuk proyek ini dapat ditemukan di repositori **GitHub** . Masuk ke folder interface, dan cukup buka file HTML dengan browser favorit Anda. Anda akan melihat sesuatu yang serupa di dalam browser Anda



Gambar 7. 9 Tampilan hasil dari program inteface

Coba klik tombol pada antarmuka web; itu akan mengubah status relai hampir seketika.

Jika Anda berhasil membuatnya berfungsi, bravo! Anda baru saja membuat saklar lampu yang dikontrol Wi-Fi. Tentu saja, Anda dapat mengontrol lebih dari sekadar lampu dengan proyek ini. Pastikan saja relai Anda mendukung daya yang diperlukan untuk perangkat yang ingin Anda kendalikan, dan Anda siap menggunakannya.

PENUTUP

Dengan berakhirnya buku ini, saya berharap Anda telah memperoleh pemahaman yang baik tentang dasar-dasar Arduino dan menikmati setiap proyek yang Anda selesaikan. Perjalanan Anda dalam dunia Arduino baru saja dimulai, dan masih banyak lagi yang dapat Anda eksplorasi dan ciptakan. Setiap proyek yang Anda buat, sekecil apa pun, adalah langkah maju dalam mengembangkan keterampilan dan kreativitas Anda.

Arduino adalah platform yang terus berkembang dengan komunitas yang selalu berbagi ide dan inovasi. Saya mendorong Anda untuk terus belajar dan berpartisipasi dalam komunitas ini. Bergabunglah dengan forum-forum, hadir workshop, dan berbagi proyek Anda dengan orang lain. Semangat berbagi dan kolaborasi adalah kunci dalam mengembangkan keterampilan dan pengetahuan Anda.

Kesalahan adalah bagian dari proses belajar. Jangan takut untuk mencoba hal-hal baru dan berani berinovasi. Dari kesalahan tersebut, Anda akan menemukan solusi yang lebih baik dan kreatif. Ingatlah bahwa setiap tantangan adalah kesempatan untuk belajar dan tumbuh. Dengan tekad dan ketekunan, Anda akan mampu mengatasi berbagai rintangan dan mencapai hasil yang memuaskan.

Saya mengucapkan terima kasih kepada Anda, para pembaca, yang telah meluangkan waktu untuk membaca dan mempraktikkan isi buku ini. Saya berharap buku ini menjadi batu loncatan yang baik bagi Anda dalam mengeksplorasi dunia elektronik dan pemrograman dengan Arduino. Semoga Anda terus bersemangat dalam belajar dan menciptakan proyek-proyek yang inovatif.

Selamat berkreasi, dan semoga sukses selalu menyertai setiap langkah Anda dalam perjalanan ini. Saya berharap dapat melihat karya-karya Anda di masa depan dan mendengar tentang keberhasilan Anda dalam dunia Arduino. Teruslah belajar, berbagi, dan menginspirasi orang lain dengan kreativitas dan inovasi Anda.

Salam hangat,
Yogyakarta, 18 Juni 2025

Penulis

Daftar Pustaka

1. Kadir, A. (2017). Pemrograman arduino dan processing. Elex Media Komputindo.
2. Mutaqin, R., Kom, S., & Yoziarde, L. A. *Pemrograman Arduino*. PT KIMHSAFI ALUNG CIPTA.
3. Arduino, U. N. O. (2015). Arduino uno. Datasheet.(Downloaded: 13 Jun 2020). <https://datasheet.octopart.com/A000066-Arduino-datasheet-38879526.pdf>.
4. Badamasi, Y. A. (2014, September). The working principle of an Arduino. In 2014 11th international conference on electronics, computer and computation (ICECCO) (pp. 1-4). IEEE.
5. Cameron, N., Cameron, N., & Pao. (2019). *Arduino Applied* (pp. 237-259). New York, NY, USA: Apress.
6. Nayyar, A., & Puri, V. (2016, March). A review of Arduino board's, Lilypad's & Arduino shields. In 2016 3rd international conference on computing for sustainable global development (INDIACom) (pp. 1485-1492). IEEE.
7. Kurniawan, A. (2015). *Arduino Uno: A Hands-On Guide for Beginner*. PE press.
8. Handoko, P. (2017). Sistem Kendali Perangkat Elektronika Monolitik Berbasis Arduino Uno R3. Prosiding Semnastek.
9. Pan, T., Zhu, Y., Pan, T., & Zhu, Y. (2018). Getting started with Arduino. *Designing Embedded Systems with Arduino: A Fundamental Technology for Makers*, 3-16.
10. Kadir, A. (2016). *Simulasi Arduino*. Elex Media Komputindo.

Pemrograman Arduino untuk Pemula: Proyek Sederhana dan Edukatif

Ir. Alfian Ma'arif, ST, M.Eng.,
Shandy Auliya Ma'arief,
Ir. Purwono, S.Kom., M.Kom



Penerbit Yayasan Peneliti Teknologi Teknik Indonesia