****

**zXC**

**KATA PENGANTAR**

Pada era globalisasi seperti ini dibutuhkan pengetahuan dan pengalaman yang cukup memadai. Dengan begitu seorang pelajar Indonesia diharapkan dapat bersaing dan memiliki kompetensi yang tinggi dalam usaha – usaha pengembangan di sektor ilmu pengetahuan maupun teknologi.

Begitu pula dalam pengembangan dunia mikrokontroller, mikrokontroller merupakan devais yang sangat penting dalam kemajuan teknologi di berbagai bidang. Baik dalam sektor otomotif, industri, robotika, dan sektor penting lainnya. Mikrokontroller memiliki peran yang sangat vital yaitu sebagai pusat pengolahan data yang berfungsi menyerupai otak manusia. Arduino merupakan salah satu pengembangan teknologi mikrokontroller yang saat ini menjadi sangat populer di kalangan mahasiswa maupun kalangan insinyur.

Dikarenakan hal tersebut di atas, maka disusunlah Modul Pelatihan Arduino. Modul ini disusun untuk memudahkan dan sebagai pemandu bagi para peserta “*Arduino Training*”. Modul ini juga menguraikan prinsip dasar pemrograman Arduino hingga pengaplikasian fungsi – fungsi yang terdapat pada Arduino.

Dengan terselesaikannya Modul Pelatihan Arduino ini, tim penyusun mengucapkan terima kasih kepada :

1. Tuhan Yang Maha Esa, karena atas rahmat dan hidayah-Ny, *Arduino Training* 2015 dapat terlaksana.
2. Rekan – rekan Pengurus Laboratorium Pengukuran Fisis Teknik Fisika – ITS, karena atas loyalitas dan dedikasiny, *Arduino Matlab Training* 2015 dapat terealisasi.
3. Semua pihak yang turut mendukung dalam berjalannya kegiatan *Arduino Training* 2015 dan penerbitan modul ini secara langsung maupun tidak langsung.

Akhir kata, tidak ada gading yang tak retak, penulis menyadari bahwa modul pelatihan ini masih jauh dari sempurna, karena kesempurnaan hanyalah milik Tuhan Yang Maha Esa. Oleh karena itu kritik dan saran yang membangun sangatlah diharapkan guna perbaikan modul ini pada edisi berikutnya. Semoga bermanfaat.

Surabaya, 19 April 2015

Tim Penyususn

**MODUL I**

**ARDUINO & MATLAB**

1. **ARDUINO**

Arduino adalah sebuah platform dari *physical computing* yang bersifat *open source.* Arduino tidak hanya sekedar sebuah alat pengembangan, tetapi ia adalah kombinasi dari hardware, bahasa pemrograman dan Integrated Development Environment (IDE) yang canggih. IDE adalah sebuah software yang sangat berperan untuk menulis program, meng-compile menjadi kode biner dan meng-upload ke dalam memory microcontroller. Ada banyak projek dan alat-alat dikembangkan oleh akademisi dan profesional dengan menggunakan Arduino, selain itu juga ada banyak modul-modul pendukung (sensor, tampilan, penggerak dan sebagainya) yang dibuat oleh pihak lain untuk bisa disambungkan dengan Arduino.

Arduino dikembangkan oleh sebuah tim yang beranggotakan orang-orang dari berbagai belahan dunia. Anggota inti dari tim ini adalah:

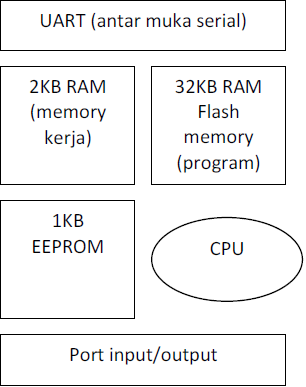
* Massimo Banzi, Milano, Italy
* David Cuartielles, Malmoe, Sweden
* Tom Igoe, New york, USA
* Gianluca Martina, Torino, Italy
* David A. Melis, Boston, USAMateri 2 : instalasi arduino

Saat ini ada bermacam-macam bentuk papan Arduino yang disesuaikan dengan peruntukannya seperti diperlihatkan berikut ini:

Menggunakan USB sebagai antar muka pemrograman atau komunikasi komputer. Contoh:

* Arduino Uno
* Arduino Duemilanove
* Arduino Diecimila
* Arduino NG Rev. C
* Arduino NG (Nuova Generazione)
* Arduino Extreme dan Arduino Extreme v2
* Arduino USB dan Arduino USB v2.0

Komponen utama di dalam papan Arduino adalah sebuah microcontroller 8 bit dengan merk **ATmega** yang dibuat oleh perusahaan **Atmel Corporation**. Berbagai papan Arduino menggunakan tipe ATmega yang berbeda-beda tergantung dari spesifikasinya, sebagai contoh Arduino Uno menggunakan ATmega328 sedangkan Arduino Mega 2560 yang lebih canggih menggunakan ATmega2560. Untuk memberikan gambaran mengenai apa saja yang terdapat di dalam sebuah microcontroller, pada gambar berikut ini diperlihatkan contoh diagram blok sederhana dari microcontroller ATmega328 (**dipakai pada Arduino Uno**).



Blok-blok di atas dijelaskan sebagai berikut:

* ***Universal Asynchronous Receiver/Transmitter (UART)***adalah antar muka yang digunakan untuk komunikasi serial seperti pada RS-232, RS-422 dan RS-485.
* **2KB RAM** pada memory kerja bersifat *volatile* (hilang saat daya dimatikan), digunakan oleh variable-variabel di dalam program.
* **32KB RAM** flash memory bersifat *non-volatile*, digunakan untuk menyimpan program yang dimuat dari komputer. Selain program, flash memory juga menyimpan *bootloader*. **Bootloader** adalah program inisiasi yang ukurannya kecil, dijalankan oleh CPU saat daya dihidupkan. Setelah bootloader selesai dijalankan, berikutnya program di dalam RAM akan dieksekusi.
* **1KB EEPROM** bersifat non-volatile, digunakan untuk menyimpan data yang tidak bolehhilang saat daya dimatikan. Tidak digunakan pada papan Arduino.
* ***Central Processing Unit (CPU)***, bagian dari microcontroller untuk menjalankan setiap instruksi dari program.
* **Port input/output**, pin-pin untuk menerima data (input) digital atau analog, dan mengeluarkan data (output) digital atau analog.
  1. **Bagian-Bagian Papan Arduino**

Dengan mengambil contoh sebuah papan Arduino tipe USB, bagian-bagiannya dapat dijelaskan sebagai berikut.



* **14 pin input/output digital (0-13)** Berfungsi sebagai input atau output, dapat diatur oleh program. Khusus untuk 6 buah pin 3, 5, 6, 9, 10 dan 11, dapat juga berfungsi sebagai pin analog output dimana tegangan output-nya dapat diatur. Nilai sebuah pin output analog dapat diprogram antara 0 – 255, dimana hal itu mewakili nilai tegangan 0 – 5V.
* **USB** Berfungsi untuk:

1. Memuat program dari komputer ke dalam papan
2. Komunikasi serial antara papan dan komputer
3. Memberi daya listrik kepada papan

* **Sambungan SV1** Sambungan atau *jumper* untuk memilih sumber daya papan, apakah dari sumber eksternal atau menggunakan USB. Sambungan ini tidak diperlukan lagi pada papan Arduino versi terakhir karena pemilihan sumber daya eksternal atau USB dilakukan secara otomatis.
* **Q1 – Kristal (*quartz crystal oscillator*)** Jika microcontroller dianggap sebagai sebuah otak, maka kristal adalah jantung-nya karena komponen ini menghasilkan detak-detak yang dikirim kepada microcontroller agar melakukan sebuah operasi untuk setiap detak-nya. Kristal ini dipilih yang berdetak 16 juta kali per detik (16MHz).
* **Tombol Reset S1** Untuk me-reset papan sehingga program akan mulai lagi dari awal. Perhatikan bahwa tombol reset ini bukan untuk menghapus program atau mengosongkan microcontroller.
* ***In-Circuit Serial Programming (ICSP)*** Port ICSP memungkinkan pengguna untuk memprogram microcontroller secara langsung, tanpa melalui bootloader. Umumnya pengguna Arduino tidak melakukan ini sehingga ICSP tidak terlalu dipakai walaupun disediakan.
* **IC 1 – Microcontroller Atmega** Komponen utama dari papan Arduino, di dalamnya terdapat CPU, ROM dan RAM.
* **X1 – Sumber Daya Eksternal** Jika hendak disuplai dengan sumber daya eksternal, papan Arduino dapat diberikan tegangan DC antara 9-12V.
* **6 pin input analog (0-5)** Pin ini sangat berguna untuk membaca tegangan yang dihasilkan oleh sensor analog, seperti sensor suhu. Program dapat membaca nilai sebuah pin input antara 0 – 1023, dimana hal itu mewakili nilai tegangan 0 – 5V.

Secara umum Arduino terdiri dari dua bagian, yaitu:

1. Hardware: Papan input/output (I/O)
2. Software: Software Arduino meliputi IDE untuk menulis program, *driver* untuk koneksi dengan komputer, contoh program dan *library* untuk pengembangan program.
   1. **Bahasa Pemrograman Arduino**

Bahasa pemrograman Arduino adalah bahasa C. Tetapi bahasa ini sudah dipermudah menggunakan fungsi-fungsi yang sederhana sehingga pemula pun bisa mempelajarinya dengan cukup mudah. Untuk membuat program Arduino dan mengupload ke dalam board Arduino, anda membutuhkan software Arduino IDE (Integrated Development Enviroment) yang bisa di download gratis di [http://arduino.cc/en/Main/Software](http://arduino.cc/en/Main/Software" \t "_blank" \o "Software Arduino)Sebelum kita mulai menggunakan Arduino, terlebuih dahulu kita harus mengetahui struktur pemrograman Arduino. Berikut ini adalah macam-macam struktur pemrograman pada Arduino:

**Strukture**

Berikut ini adalah struktur yang terdapat pada bahasa pemrograman arduino:

1. Struktur

Struktur dasar dari bahasa pemrograman Arduino adalah sederhana, yang hanya terdiri dari dua bagian.

void setup() {

// put your setup code here, to run once:

}

void loop() {

// put your main code here, to run repeatedly:

}

Dimana setup() merupakan bagian untuk inisialisasi yang hanya dijalankan sekali di awal program, sedangkan loop() untuk mengeksekusi bagian program yang akan dijalankan berulang-ulang.

1. {} Curly Braces

Curly brace mendefinisikan awal dan akhir dari sebuah blok fungsi. Apabila programer lupa memberi curly brace, maka ketika di-compile akan mengalami eror.

1. Semicolon

Semicolon harus diberikan pada setiap statement program, karena berfungsi sebagai pembatas setiap statement program yang dibuat.

1. /\*...\*/ Block Comment

Semua statement yang ditulis dalam block comment tidak akan dieksekusi dan tidak akan di-compile sehingga tidak mempengaruhi program yang dibuat.

1. //Line Comment

Sama halnya dengan block comment, line comment pun sama hanya saja yang dijadikan komentar adalah perbaris.

**Variabel**

Variabel adalah sebuah penyimpan nilai yang dapat digunakan dalam program. Variabel dapat diubah sesuai dengan instruksi yang dibuat. Ketika mendeklarasikan variabel harus diikutsertakan type variabe; serta nilai awal variabel.

Type variabelName = 0;

Contoh :

Int inputVariable = 0;

Mendefinisikan sebuah variabel bernamam inputVariable dengan nilai awal 0.

Kemudian skop variabel/variable scope. Variabel ini dapat dideklarasikan pada awal void setup() secara lokal di dalam sebuah fungsi dan terkadang di dalam sebuah blok statement pengulangan. Yang terakhir adalah variabel global, variabel ini dideklarasikan di setiap blok fungsi dan dideklarasikan pada awal program sebelum void setup().

**Tipe Data**

Seperti dipemrograman bahasa C, di Arduino terdapat beberapa tipe data yang harus dipahami dan diketahui agar dapat menggunakan suatu tipe data dengan benar. Berikut adalah tipe data yang terdapat dalam pemrograman Arduino:

1. Byte

Tipe byte dapat menyimpan 8-bit nilai angka bilangan asli tanpa koma. Byte memiliki range 0 - 255.

Contoh :

Byte biteVariabel=180;

Mendeklarasikan ‘biteVariable’ sebagai type byte.

1. Integer

Integer adalah tipe data yang utama untuk menyimpan nilai bilangan bulat tanpa koma. Penyimpanan integer sebesar 16-bit dengan range 32767 – 32768.

Contoh :

Int integerVariable = 1600;

mendeklarasikan ‘integerVariable’ sebagai type integer

1. Long

Perluasan ukuran untuk long integer, penyimpanan long integer sebesar 32-bit dengan range 2147483647 sampai -2147483647.

Contoh :

Long longVariable = 100000;

mendeklarasikan ‘longVariable’ sebagai type long.

1. Float

Float adalah tipe data yang dapat menampung nilai desimal, float merupakan penyimpan yang lebih besar dan dapat meyimpan sebesar 32-bit dengan range 34028235E+38 sampai -34028235E+38.

Contoh :

Int integerVariable = 3,14;

mendeklarasikan ‘integerVariable’ sebagai type float.

1. Array

Array adalah kumpulan nilai yang dapat diakses dengan index number, nilai yang terdapat dalam array dapat dipanggil dengan cara menuliskan nama array dan index number. Array dengan index 0 merupakan nilai dari array. Array perlu dideklarasikan dan kalau perlu diberi nilai sebelum digunakan.

Int arraysName[] = {nilai0, nilai1, nilai2...}

Contoh :

Int arrayAku[] = {2,3,4,8,9};

x = arrayAku[5];

maka x sekarang adalah 9.

**Aritmatik**

Operator aritmayik terdiri dari penjumlahan, pengurangan, pengkalian, dan pembagian (+, -, \*, /).

**Compound Assignments**

Compound assignments merupakan kombinasi dari aritmatika dengan sebuah variabel. Ini biasanya dipakai pada pengulangan.

x ++; // sama seperti x = x + 1 atau menaikkan nilai x sebesar 1

x --; // sama seperti x = x – 1 atau mengurangi nilai x sebesar 1

x +=; // sama seperti x = x + y

x-=y; // sama seperti x = x – y

x\*=y; // sama seperti x = x\*y

x/=y; // sama seperti x = x/y

**Comparison**

Statement ini membandingkan dua variabel dan apabila terpenuhi akan bernilai 1 atau true. Statement ini banyak digunakan dalam operator bersyarat.

x ==y; // x sama dengan y

x!=y; // x tidak sama dengan y

x<y; // x lebih kecil dari y

x>y; // x lebih besar dari y

x<=y; // x lebih kecil dari sama dengan y

x>=y; // x lebih besar dari sama dengan y

**Logic Operator**

Operator logikal digunakan untuk membandingkan 2 expresi dan mengembalikan nilai balik benar atau salah tergantung dari operator yang digunakan. Terdapat 3 operator logik AND, OR, dan NOT, yang biasanya digunakn pada IF statement.

Logika AND

If( x>0&&x<5); // bernilai benar apabila kedua operator pembanding terpenuhi

Logika OR

If( x>0||y>0); // bernilai benar apabila salah satu operator pembanding terpenuhi

Logika NOT

If( !x>0); // bernilai benar apabila ekspresi operator bernilai salah

**Konstanta**

Arduino mempunyai beberapa variabel yang sudah dikenal yang kita sebut konstanta. Ini membuat lebih mudah untuk dibaca. Konstanta diklasifikasikan berdasarkan grup.

1. True/False

Merupakan konstanta Booleab yang mendefinisikan logic level. FALSE mendefinisikan 0 dan TRUE mendifinisaikan 1

1. High/Low

Konstata ini mendefinisikan aktifitas pin HIGH atau LOW dan digunakan ketika membaca dan menulis ke digital pin. HIGH didefinisikan sebagai 1 sedangkan LOW sebagai 0.

1. Input/Output

Konstanta ini digunakan dengan funsi pinMode() untuk mendefinisikan mode pin digital, sebagai input atau output.

**Flow Control**

Flow control merupakn sebuah operator yang memiliki pengaruh atau dapat pengkontrol suatu keadaan dalam pemrograman. Berikut ini flow control yang ada pada Arduino:

1. If

Operator if mengeset sebuah kondisi berdyarat, jika kondisi ini terpenuhi maka program akan dijalankan.

Contoh:

If(someVariable?? value)

{

//Do something;

}

1. If...else

Operator if...else mengeset sebuah kondisi bersyarat, apabila tidak sesuai dengan kondisi pertama maka akan mengeksekusi kondisi yang ada di else.

Contoh:

if(inputPin==HIGH)

{

//Do first plan;

else

//Do second plan;

}

1. For

Operator for digunakan dalam blok pengulangan tertutup.

Contoh:

For(initialization;condition;expression)

{

//Do something;

}

1. While

Operator while akan terus mengulang baris perintah yang ada dalam blok sampai ekspresi sebagai kondisi pengulangan bernilai salah.

Contoh:

While(someVariable??value)

{

//Do something

}

1. Do...while

Sama halnya dengan while() hanya saja pada operator Do...While tidak melakukan pengecekan pada awal tapi di akhir, sehingga otomatis akan melakukan satu kali baris perintah walaupun pada awalnya sudah terpenuhi.

Contoh:

Do

{

//Do something;

}

While(someVariable//value);

**Digital I/O**

Input/output digital pada breadboard Arduino Uno ada 14, pengalamatanya 0 – 13, ada saat tertentu I/O 0 dan 1 tidak bisa digunakan karena dipakai untuk komunakasi serial, sehingga harus hati-hati dalam pengalokasian I/O. berikut ini merupak istilah dalam bahasa pemrograman digital I/O Arduino:

1. Pin Mode (pin, Mode)

Digunakan dalam void setup() untuk mengkonfigurasi pin apakah sebagai input atau output. Pin-pin Arduino digital secara default dikonfigurasi sebagai input sehingga untuk merubahnya harus menggunakan operator pinMode (pin, mode).

Contoh:

pinMode(pin,OUTPUT); //mengeset pin sebagai output;

pinMode(pin,INPUT): // mengeset pin sebagai input.

1. Digital Read (pin)

Membaca nilai pin yang kita kehendaki dengan hasil HIGH atau LOW.

Contoh:

Value = digitalRead(pin); //membaca ‘value’ pin HIGH atau LOW

1. Digital Write (pin)

Digunakan untuk mengset pin digital. Arduino memilki 14 pin digital.

Contoh:

digitalWrite(pin, HIGH); //seet pin to HIGH

**Analog I/O**

Input/output analog pada breadbroad Arduino UNO ada 6 pengalamatanya 0 – 5. berikut ini merupak istilah dalam bahasa pemrograman analog I/O Arduino:

1. Analog Read (pin)

Membaca nilai pin analog yang memiliki resolusi 10-bit. Fungsi ini hanya dapat bekerja pada analog pin (0 - 5). Hasil dari pembacaa berupa nilai integer dengan range 1023.

Contoh:

Value = analogRead (pin); // membaca ‘value’ sama dengan nilai analog

1. Analog Write (pin, value)

Mengirimkan nilai analog pada pin analog Arduino.

Contoh:

analogWrite(pin, value) // menulis ke pin analog

**Time**

Ardunio memiliki istilah bahasa pemrograman dalam mendeklarasikan waktu. Berikut ini adalah time yang ada pada Arduino:

1. Delay (ms)

Menghentikan program untuk sesaat sesuai dengan yang di kehendaki, satuanya dalam millisecond.

Contoh:

delay(1000); // menggu selama 1 detik

1. Millis ()

Mengembalikan nilai dalam millisecond, dihitung sejak arduino board menyala. Penapungnya harus long integer.

Contoh:

Value = millis(); //set ‘value’ equal to millis()

**Math**

Berikut ini merupakan istilah math dalam bahasa pemroraman arduino:

1. Min (x,y)

Membandingkan 2 variabel dan akan mengembalikan nilai yang paling kecil.

Contoh:

Value = min(value, 100); // set ‘value’ sebagai yang paling kecill

1. Max (x,y)

Max merupakan kebalikan dari min.

Contoh:

Value = max(value, 1000); //set ‘value’ sebagai nilai yang paling besar dari kedua nilai

**Serial**

Statement ini digunlan untuk mengaktifkan komunikasi serial pada Arduino. Berikut ini adalah istilah komunikasi serial pada bahasa pemrgraman Arduino:

1. Serial.begin (rate)

Staement ini digunakan untuk mengaktifkan komunikasi serial dan mengeset baudrate.

Contoh:

void setup()

{

Serial.begin(9600); //mengaktifkan seeria; poert dan mengeset baudrate 9600 bps

}

1. Serial Printing (data)

Mengirim data ke serial port.

Conoh:

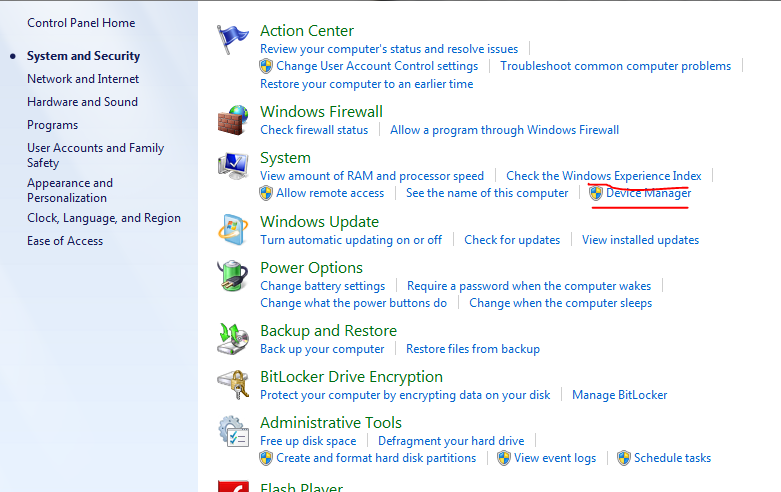
Serial.print(100); //mengirimkan nilai 100 setiap baris

Serial.println(100); //mengirimkan 100 setiap kolom

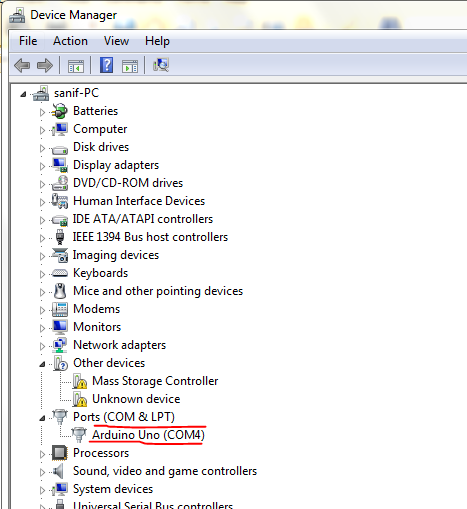
* 1. **Instalasi Arduino IDE**

Arduino IDE merupakan software Arduino yang digunakan sebagai media penghubung antara user dan hardware Arduino. Sofware Arduino IDE ini tersedia di <http://arduino.cc/en/Main/software>, disana juga terdapat petunjuk langkah-langkah pengisntalan software Arduino IDE ke laptop/Pc. Berikut adalah langkah-langkah menginstal software arduino pada windows:

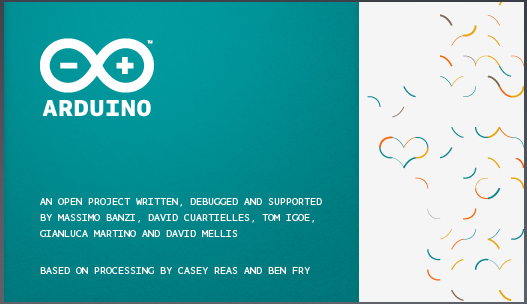
1. Tancapkan kabel USB Arduino uno ke laptop/PC.
2. Instal software ardunio yang telah anda miliki. Instal dengan mengikuti intruksi yang muncul.
3. Akan muncul perintah menginstal USB driver. Klik Yes.
4. Setelah proses instalasi selesai, klik pada Start Menu, dan buka Control Panel.
5. Didalam Control Panel, arahkan ke System and Security. Kemudian, klik pada System window. Ketika System windows muncul, buka Device Manager.



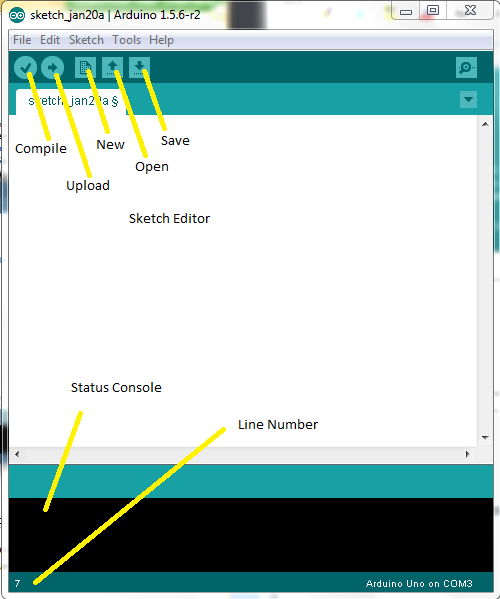
1. Pada Device Manager, arahkan ke Ports (COM & PLT) dan pastikan port Arduino telah terdeteksi. Sepeti gambr dibawah ini:



1. Dan ingat bahwa Arduino anda berada pada pada COMxx.



Berikut ini adalah penjelasan mengenai tampilan muka dari Arduino IDE:



**Compile:** proses sebelum program instruksi di kirim ke board, compile dibutuhkan untuk merubah instruksi agar dimengerti oleh board.

**Upload**: meng-Compile dan kemudian mengirim program instruksi ke board.

**Skecth Editor**: tempat menulis program instruksi.

**Open:** membuka skecth yang telah ada dalam laptop/PC.

**Save:** menyimpan program instruksi.

**New:** membuka skecth edit or baru.

**Status Console**: menampilkan running IDE dan menampilkan eror yang terjadi jika terdapat eror.

**Line Number:** menuntukkan baris dimana krusor berada.

1. **MATLAB**

**MATLAB** (**mat**rix **lab**oratory) adalah sebuah lingkungan [komputasi numerikal](http://id.wikipedia.org/wiki/Analisis_numerik" \o "Analisis numerik) dan [bahasa pemrograman](http://id.wikipedia.org/wiki/Bahasa_pemrograman) komputer generasi keempat. Dikembangkan oleh [The MathWorks](http://id.wikipedia.org/w/index.php?title=The_MathWorks&action=edit&redlink=1" \o "The MathWorks (halaman belum tersedia)), MATLAB memungkinkan manipulasi [matriks](http://id.wikipedia.org/wiki/Matriks_%28matematika%29" \o "Matriks (matematika)), pem-plot-an fungsi dan data, implementasi [algoritma](http://id.wikipedia.org/wiki/Algoritma" \o "Algoritma), pembuatan [antarmuka](http://id.wikipedia.org/wiki/Antarmuka" \o "Antarmuka) pengguna, dan peng-antarmuka-an dengan program dalam bahasa lainnya. Meskipun hanya bernuansa numerik, sebuah kotak kakas (*toolbox*) yang menggunakan mesin simbolik [MuPAD](http://id.wikipedia.org/w/index.php?title=MuPAD&action=edit&redlink=1), memungkinkan akses terhadap kemampuan aljabar komputer. Sebuah paket tambahan, [Simulink](http://id.wikipedia.org/wiki/Simulink" \o "Simulink), menambahkan simulasi grafis multiranah dan Desain Berdasar-Model untuk sistem terlekat dan dinamik. Pada tahun 2004, MathWorks mengklaim bahwa MATLAB telah dimanfaatkan oleh lebih dari satu juta pengguna di dunia pendidikan dan industri

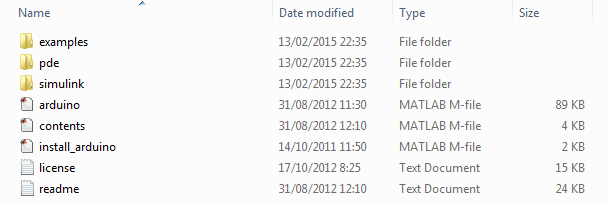
MATLAB (yang berarti "*matrix laboratory*") diciptakan pada akhir tahun 1970-an oleh [Cleve Moler](http://id.wikipedia.org/w/index.php?title=Cleve_Moler&action=edit&redlink=1" \o "Cleve Moler (halaman belum tersedia)), yang kemudian menjadi Ketua Departemen [Ilmu Komputer](http://id.wikipedia.org/wiki/Ilmu_Komputer" \o "Ilmu Komputer) di Universitas New Mexico. Ia merancangnya untuk memberikan akses bagi mahasiswa dalam memakai [LINPACK](http://id.wikipedia.org/w/index.php?title=LINPACK&action=edit&redlink=1" \o "LINPACK (halaman belum tersedia)) dan [EISPACK](http://id.wikipedia.org/w/index.php?title=EISPACK&action=edit&redlink=1" \o "EISPACK (halaman belum tersedia)) tanpa harus mempelajari [Fortran](http://id.wikipedia.org/wiki/Fortran" \o "Fortran). Karyanya itu segera menyebar ke universitas-universitas lain dan memperoleh sambutan hangat di kalangan komunitas [matematika terapan](http://id.wikipedia.org/wiki/Matematika_terapan" \o "Matematika terapan). [Jack Little](http://id.wikipedia.org/w/index.php?title=John_N._Little&action=edit&redlink=1" \o "John N. Little (halaman belum tersedia)), seorang insinyur, dipertemukan dengan karyanya tersebut selama kunjungan Moler ke Universitas Stanford pada tahun 1983. Menyadari potensi komersialnya, ia bergabung dengan Moler dan Steve Bangert. Mereka menulis ulang MATLAB dalam bahasa pemrograman C, kemudian mendirikan [The MathWorks](http://id.wikipedia.org/w/index.php?title=The_MathWorks&action=edit&redlink=1" \o "The MathWorks (halaman belum tersedia)) pada tahun 1984 untuk melanjutkan pengembangannya. Pustaka yang ditulis ulang tadi kini dikenal dengan nama JACKPAC.[*[butuh rujukan](http://id.wikipedia.org/wiki/Wikipedia:Kutip_sumber_tulisan" \o "Wikipedia:Kutip sumber tulisan)*] Pada tahun 2000, MATLAB ditulis ulang dengan pemakaian sekumpulan pustaka baru untuk manipulasi matriks, [LAPACK](http://id.wikipedia.org/w/index.php?title=LAPACK&action=edit&redlink=1" \o "LAPACK (halaman belum tersedia)).

MATLAB pertama kali diadopsi oleh insinyur rancangan kontrol (yang juga spesialisasi Little), tapi lalu menyebar secara cepat ke berbagai bidang lain. Kini juga digunakan di bidang pendidikan, khususnya dalam pengajaran [aljabar linear](http://id.wikipedia.org/wiki/Aljabar_linear" \o "Aljabar linear) dan [analisis numerik](http://id.wikipedia.org/wiki/Analisis_numerik), serta populer di kalangan ilmuwan yang menekuni bidang [pengolahan citra](http://id.wikipedia.org/wiki/Pengolahan_citra" \o "Pengolahan citra).

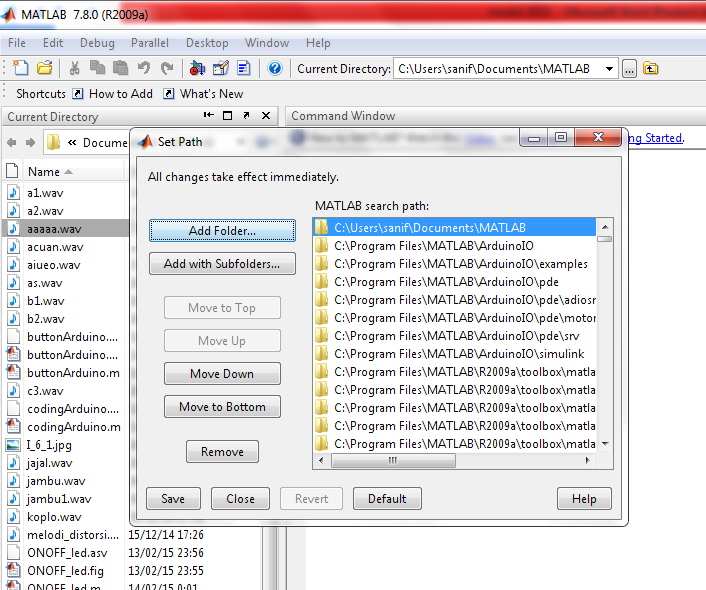
1. **MENGINTEGRASIKAN ARDUINO DAN MATLAB**

Pada pelatihan kali ini kita akan mengintegerasikan Arduino dengan software Matlab. Ketika kita telah dapat mengintegerasikan keduanya, maka kita dapat mengontrol Arduino menggunakan Matlab. Secara khusus, kita dapat menampilkan grafik, diagram, ataupun interface menggunakan GUI (sebuah interface interaktif bawaan Matlab). Langsung saja kita ke langkah pengintergrasian kedua sofware ini:

1. Pastikan kita telah mempunyai file : “ArduinoIO.exe”.
2. Ekstrak file tersebut, maka didalamnya akan muncul 6 file berikut:

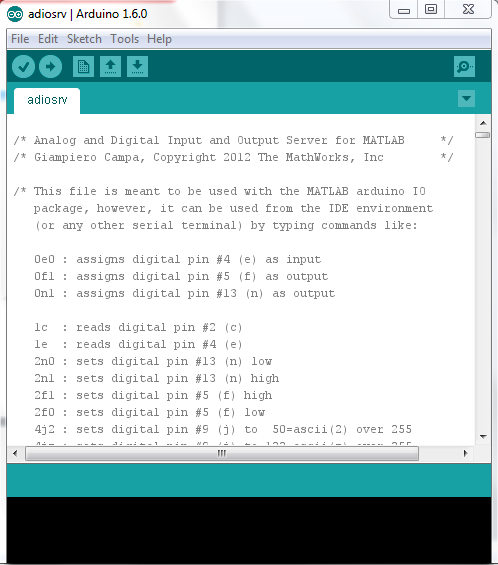


1. Copy filie “pde”, kemudian masuk ke Local Disk C (tempat instalasi arduino) - Program Files –Arduino –Libraries - paste “pde”. Atau dengan langkah: buka sofware Arduino - tools - add libraries - “pde”.
2. Buka software Matlab, klik file- Set Path. Akan muncul tampilan seperti dibawah ini:

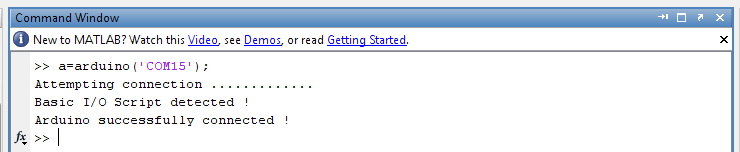


Kemudian pilih “Add folder” – arahkan ke file ArduinoIO – save.

1. Langkah selanjutnya adalah mengunggah fungsi Matlab ke Arduino. Buka software Arduino – file - example – ArduinoIO – adiosrv.



1. Langkah terakhir adalah menggunggah fungsi Arduino ke Matlab. Buka Matlab – ketikkan “a = arduino(‘COM15’) pada Command Window. Untuk port COMxx disesuaikan dengan port yang terinstal.



**MODUL II**

**INPUT DAN OUTPUT DIGITAL**

Sebagai pemula kali ini kita akan belajar mengenai input/output digital pada arduino dengan menggunakan proyek sederhana yang cukup untuk membuka pengertian akan akses pin pada arduino uno board.

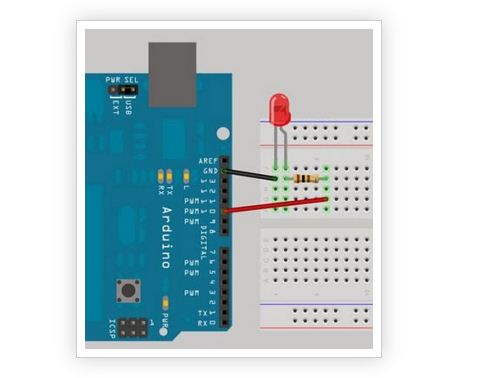
**Proyek pertama LED Berkedip**

Komponen yang dibutuhkan :

1. 1 LED
2. 1 Resistor 220 ohm
3. Kabel jumper

Langkah-langkah :

a. Buat rangkaian seperti gambar 1



Gambar 1 . Rangkain LED pada arduino uno

b. Coding pada arduino

// Project 1 - LED Blink

// Membuat LED nyala selama 1 detik kemudian mati dan

nyala lagi.

// Hubungkan Pin 8 Arduino dengan sebuah Led.

int ledPin = 8; // Deklarasi Awal

void setup() {

pinMode(ledPin, OUTPUT); // Pendefinisian pin 8 sebagai output

}

void loop() {

digitalWrite(ledPin, HIGH); // membuat LED nyala

delay(1000); // tunggu 1 detik

digitalWrite(ledPin, LOW); // membuat LED mati

delay(1000); // tunggu 1 detik

}

**Proyek kedua ( Nyala traffic lights)**

Komponen yang dibutuhkan :

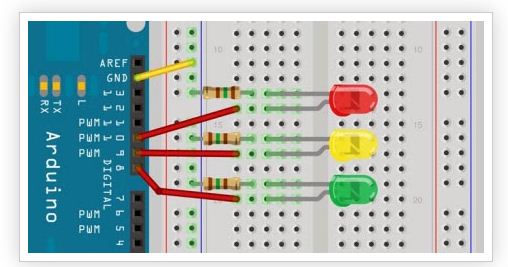
a. LED ( merah , kuning , hijau)

b. 3 resistor 150 ohm

c. kabel jumper

Langkah-langkah :

a. Buat rangkaian seperti gambar 2



Gambar 2. Rangkaian traffic light

b. Coding pada arduino

// Project 2 - traffic Lights

int ledDelay = 5000; // delay in between changes

int redPin = 10;

int yellowPin = 9;

int greenPin = 8;

void setup() {

pinMode(redPin, OUTPUT);

pinMode(yellowPin, OUTPUT);

pinMode(greenPin, OUTPUT);

}

void loop() {

digitalWrite(redPin, HIGH); // turn the red light on

delay(ledDelay); // wait 5 seconds

digitalWrite(yellowPin, HIGH); // turn on yellow

delay(2000); // wait 2 seconds

digitalWrite(greenPin, HIGH); // turn green on

digitalWrite(redPin, LOW); // turn red off

digitalWrite(yellowPin, LOW); // turn yellow off

delay(ledDelay); // wait ledDelay milliseconds

digitalWrite(yellowPin, HIGH); // turn yellow on

digitalWrite(greenPin, LOW); // turn green off

delay(2000); // wait 2 seconds

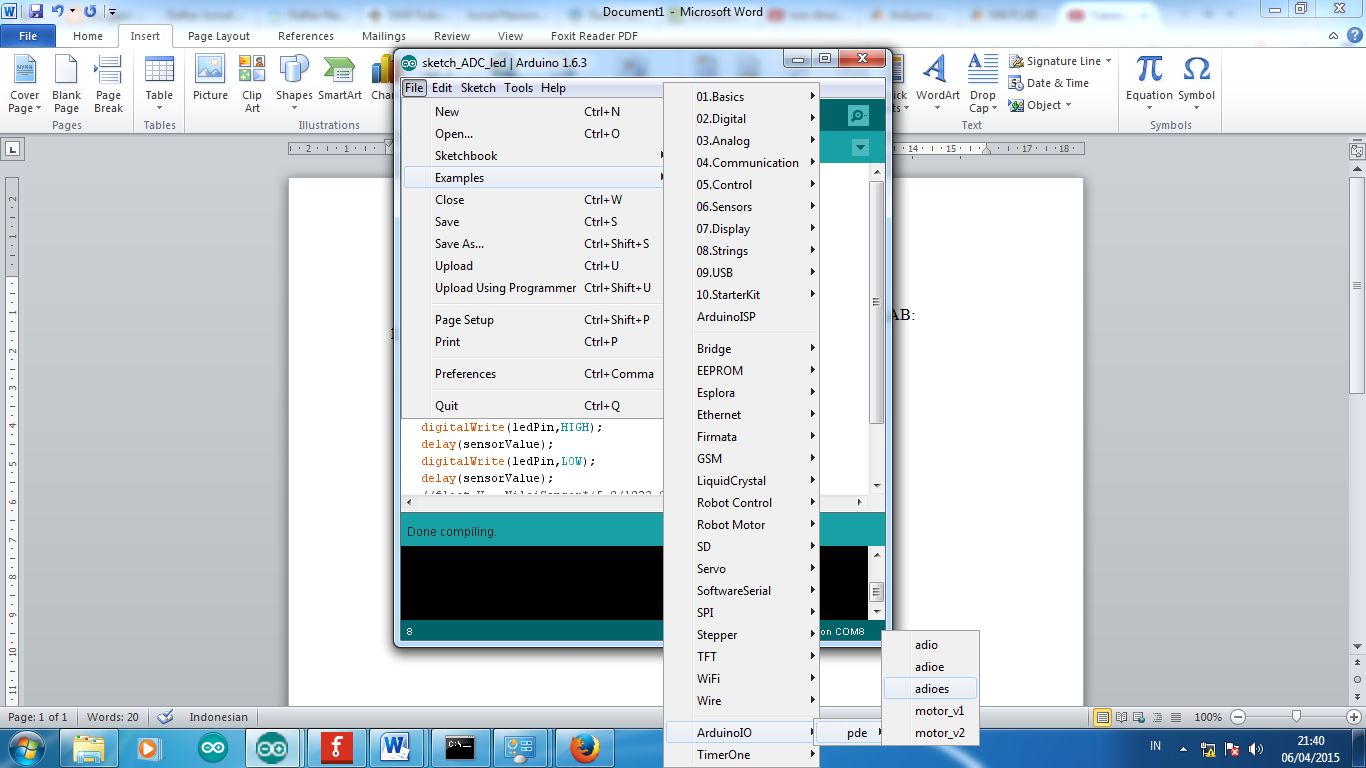
digitalWrite(yellowPin, LOW); // turn yellow off

}

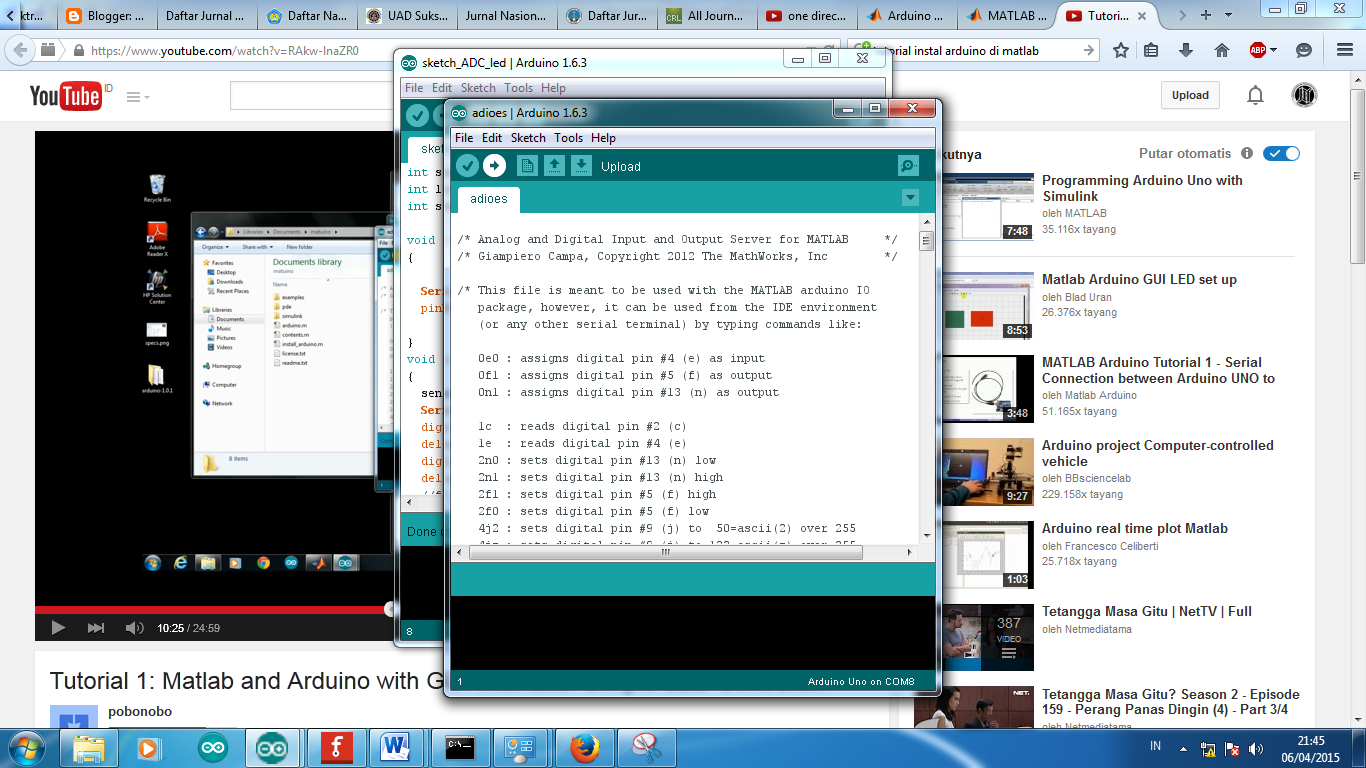
**INTERFACING ARDUINO TO MATLAB**

Berikut adalah langkah-langkah untuk interface dari arduino ke MATLAB untuk menyalakan dan mematikan LED:

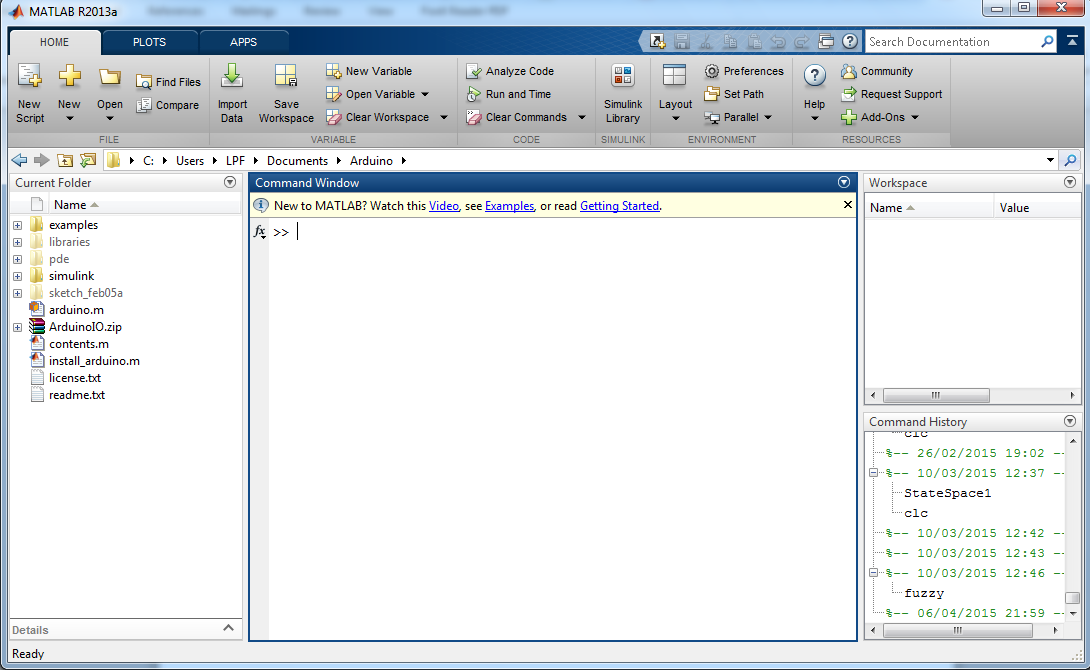
1. Buka arduino, lalu klik File → Examples → pde → adioes



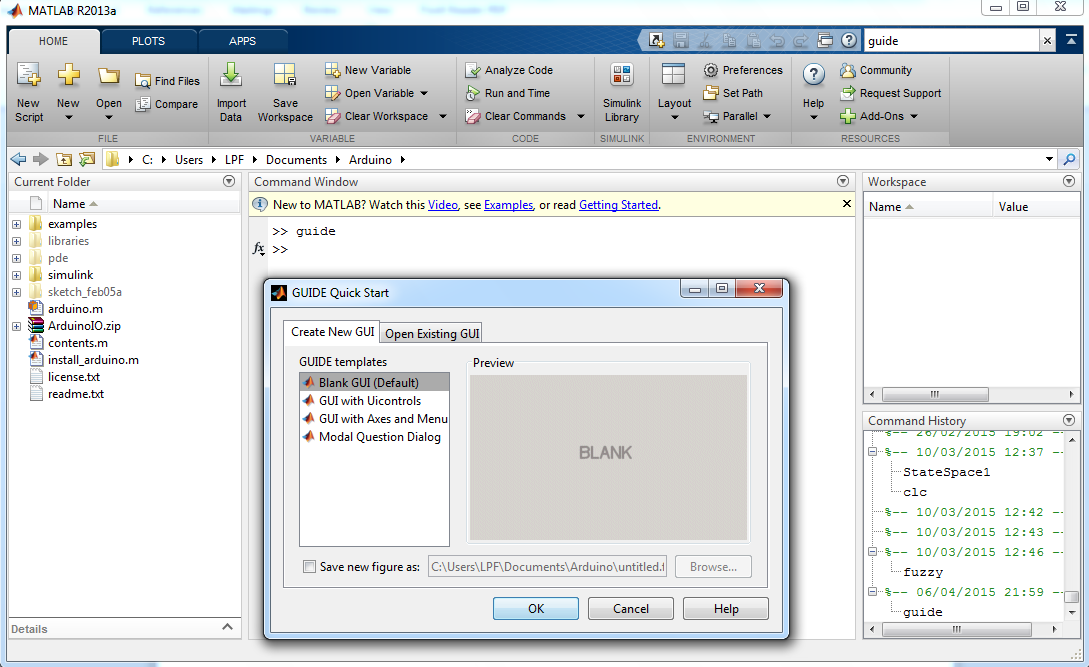
1. Setelah terbuka, klik tanda  untuk mengupload file arduino ke MATLAB



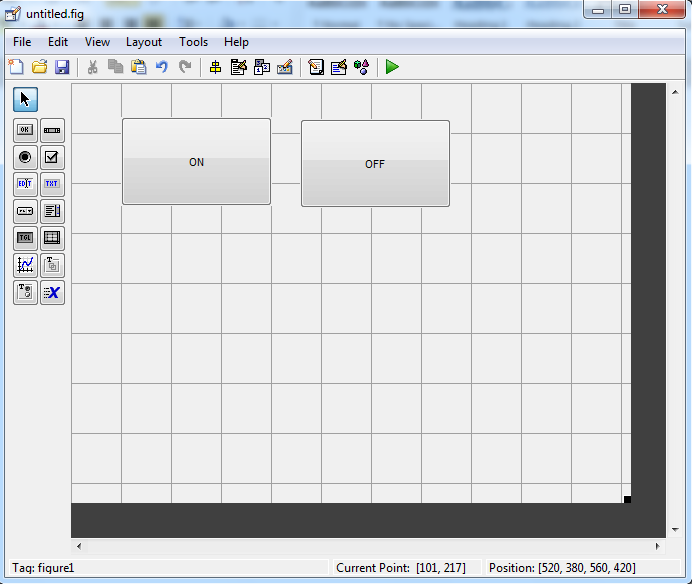
1. Setelah file arduino ter upload, buka software MATLAB. Lalu ganti folder dengan folder arduino yang berisi adioes tersebut.



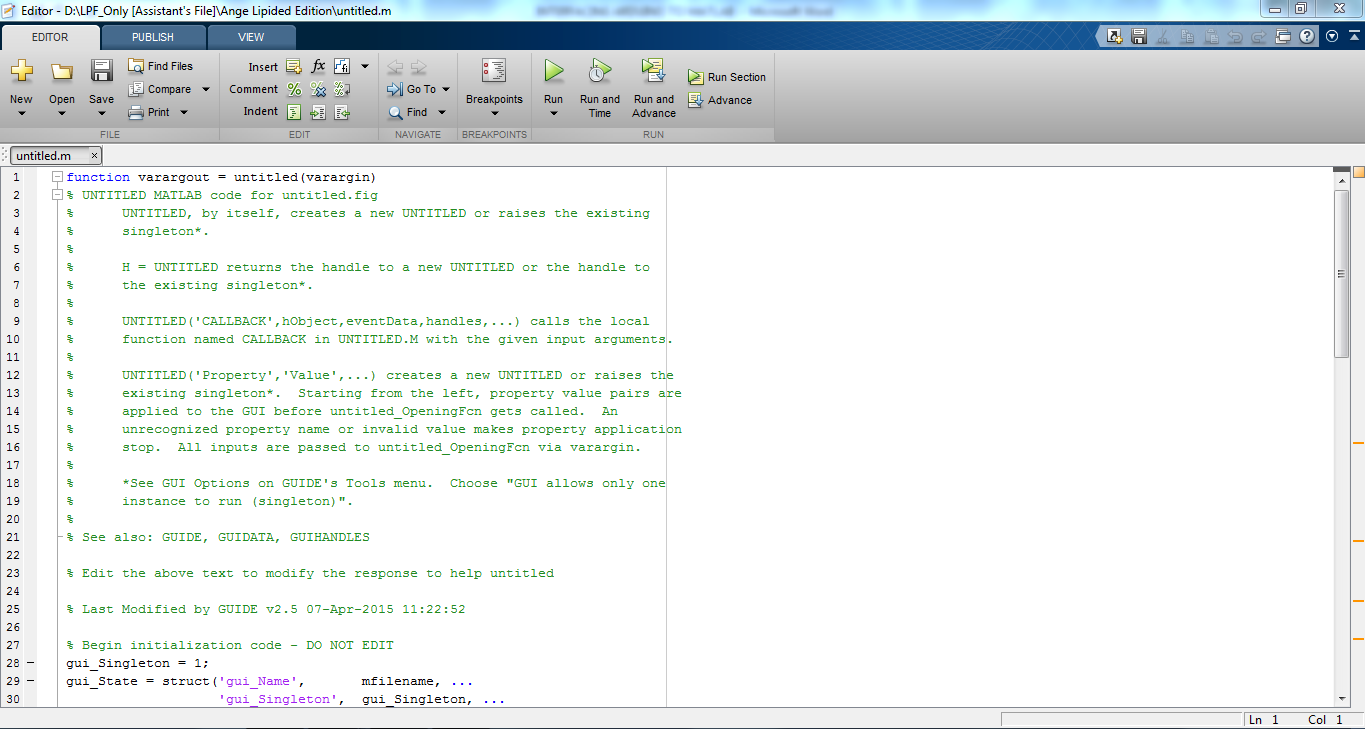
1. Ketik “guide” pada command window untuk membuat GUI editor lalu enter, lalu pilih “create new GUI” → Blank GUI (default) → OK



1. Setelah GUI editor terbuka, untuk menyalakan dan mematikan LED pada rangkaian, gunakan 2 push button. Untuk membuat push button, pilih icon push button () pada toolbar dan letakan pada worksheet. Untuk me-rename push button menjadi ON dan OFF sehingga memudahkan penggunaan, klik kiri 2x pada kotak push button lalu ganti nama push button dengan ON dan OFF pada kotak string.



1. Setelah itu, untuk meng kode push button sehingga berfungsi untuk menyalakan dan mematikan LED yaitu dengan masuk command window MATLAB yang telah berisi file arduino. Klik kanan pada push button → editor → save terlebih dahulu lalu masuk ke command window.

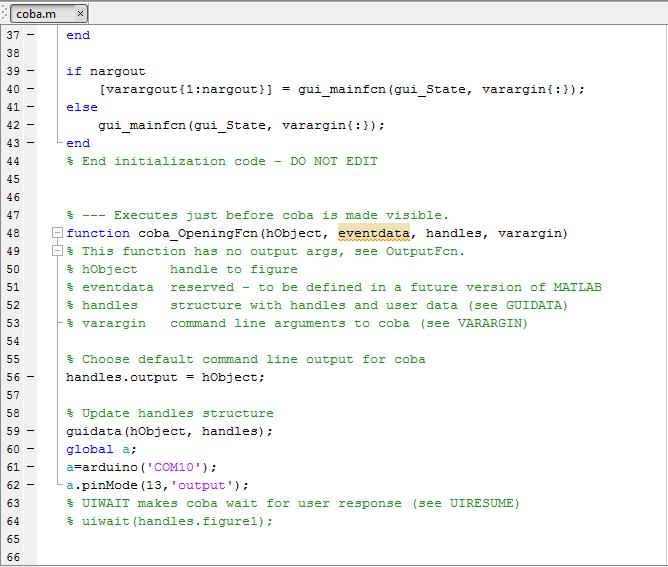


1. Untuk mendeklarasikan kodingan dari arduino, maka ketik kode dibawah guidata(hobject, handles) pada function\_namafolderFcn:

Global a;

a=arduino(‘COMxx’) → xx adalah port masukan dari arduino, dapat dilihat pada device manager

a.pinMode(y,’ouput’) → y adalah pin pada arduino yang disambungkan pada led, pin yang bisa dipakai adalah pin 8.



1. untuk menyalakan dan mematikan led pada push button, menggunakan koding dibawah function\_pushbutton1 dan function\_pushbutton2. Jika push button 1 bertujuan untuk menyalakan led, maka dibawah funtion\_pushbutton1:

global a;

a.digitalWrite(y,1); → 1 adalah logika high yang mana artinya akan menyalakan led

jika push button 2 bertujuan untuk mematikan led, maka dibawah funtion\_pushbutton2:

global a;

a.digitalWrite(y,0); → 0 adalah logika low yang mana artinya akan mematikan led

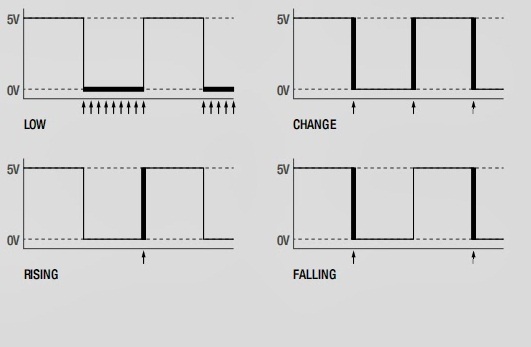
Perlu diperhatikan, sebelum menghubungkan sebuah led kedalam arduino kita harus memasang resistor agar arus yang memasuki led tidak terlampau besar sehingga tidak merusak led. Disini kita menggunakan resistor 330 Ohm.

MODUL III

**TIMER DAN INTERRUPT**

Menurut Kamus Besar Bahasa Indonesia Interrupt/Intrupsi adalah menyela atau memutus. Interrupt dalam pemograman arduino, merupakan sebuah *class* yang digunakan untuk mengolah program, dimana ketika sebuah program sedang berjalan , sebuah interrupt dapat dijalankan untuk menghentikan program secara internal interrupt dengan misalnya mengatur timer, dan external interrupt dengan memberikan interrupt melalui sebuah rangkaian hardware contohnya switch. Arduino uno memiliki 2 pin interrupt yaitu INT0 pada pin 2 digital dan INT1 pada pin 3 digital.

Arduino memiliki 4 keadaan yang dapat memicu interrupt. Yang pertama adalah mode LOW. Pada mode ini interrupt akan diaktifkan saat pin int memiliki logika low dan akan aktif selama masih berlogika low. Kemudian yang kedua adalah mode CHANGE. Interrupt akan diaktifkan saat terjadi perubahan logika baik dari low ke high ataupun high ke low. Namun pengaktifan hanya bersifat sementara dan beberapa saat kemudian program akan berjalan kembali seperti sedia kala. Ketiga adalah mode RISING. Pengaktifan interrupt akan terjadi jika pin mengalami perubahan logika dari low ke high. Sama seperti mode CHANGE, pengaktifan interrupt hanya sesaat saja dan kemudian arduino akan kembali menjalankan program yang berjalan sebelumnya. Terakhir mode FALLING. Perubahan logika pada pin int dari high ke low akan mengaktifkan fungsi interrupt. hanya sesaat saja dan kemudian arduino akan kembali menjalankan program yang berjalan sebelumnya.



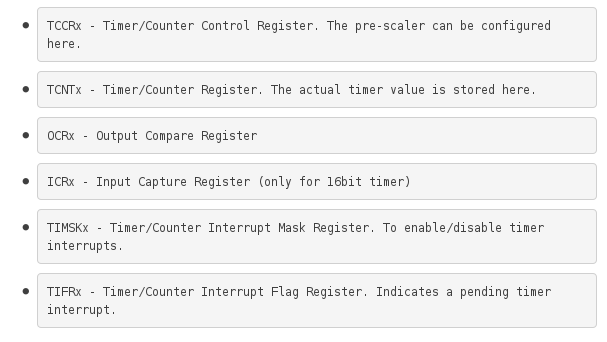
Interrupt di bedakan menjadi dua yaitu Interrupt Internal dan External. Interrupt Internal seperti USART, Timer Comparator, SPI, TWI, ADC. Sedangkan Interrupt External itu seperti RESET, INT0, dan INT1. Timer dan Counter adalah bagian dari Fitur yang dibangun di microcontroller Arduino. Hal ini seperti sebuah jam, dan dapat digunakan untuk mengukur waktu kejadian. Timer dapat diprogram oleh beberapa register khusus. Anda dapat mengkonfigurasi pra-scaler untuk timer, atau modus operasi dan banyak hal lainnya. Board Arduino yang dibangun dengan Atmel AVR ATmega168 atau ATmega328. Chip ini adalah pin yang kompatibel dan hanya berbeda dalam ukuran memori internal. Keduanya memiliki 3 timer, disebut Timer0, Timer1 dan Timer2. Timer0 dan Timer2 adalah waktu 8bit, di mana Timer1 adalah timer 16bit. Perbedaan yang paling penting antara 8bit dan 16bit timer resolusi timer. 8bits berarti 256 nilai (dua dengan kekuatan 8) di mana 16bit berarti nilai 65536 (dua pangkat 16) yang merupakan resolusi lebih tinggi. Seri Arduino Mega didasarkan pada Atmel AVR ATmega1280 atau ATmega2560. Mereka hampir sama dengan chip sebelumnya, tetapi hanya berbeda dalam ukuran memori. Chip ini memiliki 6 timer. 3 timer pertama (Timer 0, Timer1 dan Timer2) yang identik dengan ATmega168/328. 3 timer kedua Timer3, Timer4 dan Timer5 semua timer 16bit, mirip dengan Timer1.

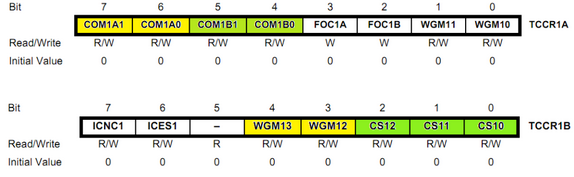
Kondisi yang memicu atau memaksa mikrokontroler untuk menghentikan program utama dan memaksa untuk menjalankan program interupsi. Interrupt di bedakan menjadi dua yaitu Interrupt Internal dan External. Interrupt Internal seperti USART, Timer Comparator, SPI, TWI, ADC. Sedangkan Interrupt External itu seperti RESET, INT0, dan INT1.

Timer dan Counter adalah bagian dari Fitur yang dibangun di microcontroller Arduino. Hal ini seperti sebuah jam, dan dapat digunakan untuk mengukur waktu kejadian. Timer dapat diprogram oleh beberapa register khusus. Anda dapat mengkonfigurasi pra-scaler untuk timer, atau modus operasi dan banyak hal lainnya.

Board Arduino yang dibangun dengan Atmel AVR ATmega168 atau ATmega328. Chip ini adalah pin yang kompatibel dan hanya berbeda dalam ukuran memori internal. Keduanya memiliki 3 timer, disebut Timer0, Timer1 dan Timer2. Timer0 dan Timer2 adalah waktu 8bit, di mana Timer1 adalah timer 16bit. Perbedaan yang paling penting antara 8bit dan 16bit timer resolusi timer. 8bits berarti 256 nilai (dua dengan kekuatan 8) di mana 16bit berarti nilai 65536 (dua pangkat 16) yang merupakan resolusi lebih tinggi.

Seri Arduino Mega didasarkan pada Atmel AVR ATmega1280 atau ATmega2560. Mereka hampir sama dengan chip sebelumnya, tetapi hanya berbeda dalam ukuran memori. Chip ini memiliki 6 timer. 3 timer pertama (Timer 0, Timer1 dan Timer2) yang identik dengan ATmega168/328. 3 timer kedua Timer3, Timer4 dan Timer5 semua timer 16bit, mirip dengan Timer1.



Sumber clock yang berbeda dapat dipilih untuk setiap waktu secara independen. Untuk menghitung frekuensi waktu (misalnya 2Hz menggunakan Timer1) Anda akan perlu:

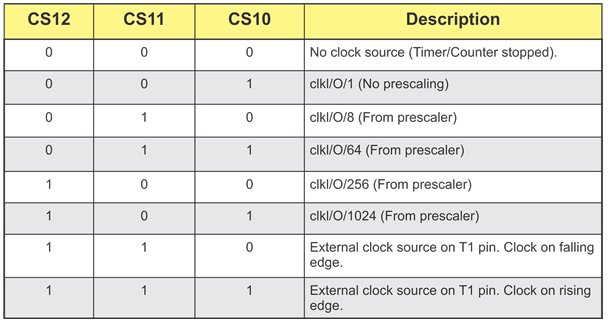
1. CPU frekuensi 16MHz untuk Arduino

2. Maksimum nilai timer counter (256 untuk 8bit, 65536 untuk 16bit timer)

3. Bagilah frekuensi CPU melalui dipilih pra-scaler (16000000/256 = 62500)

4. Bagilah hasil melalui frekuensi yang diinginkan (62500/31250 = 2Hz)

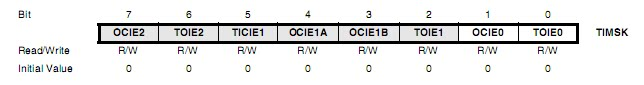
5. Verifikasi hasil melawan maksimum nilai timer counter (31250 <65536 sukses) jika gagal, pilih pra-scaler yang lebih besar.





Register TIMSK dan TIFR

Register TIMSK (Timer/Counter Interrupt Mask Register) dan TIFR (Timer Interrupts Flag Register) adalah dua buah register yang sering di atur pada saat, menggunakan interupsi timer, lebih-lebih para pengguna assembler pasti lebih familiar dengan kedua register ini.Register TIMSK (Timer/Counter Interrupt Mask Register) memiliki skema seperti ini:



berikut keterangan setiap bit-nya:

1. bit 0 \_\_TOEI0: Timer/Counter 0 Interrupt Enable

Jika bit tersebut diberi logika satu dan bit 1 register SREG juga set maka bisa dilakukan enable interupsi overflow timer/counter0.

2. bit1\_\_OCIE0: Timer/Counter0,Output Compare Match Interrupts Enable

Jika bit ini diberi logika satu dan bit 1 register register SREG, maka bisa dilakukan enable interupsi output compare match timer/counter0.

3. bit2\_\_TOEI1:Timer/Counter 1 Interrupt Enable

Jika bit tersebut diberi logika satu dan bit 1 register SREG juga set maka bisa dilakukan enable interupsi overflow timer/counter1.

4. bit3\_\_OCIE1B: Timer/Counter1,Output Compare B Match Interrupts Enable

Jika bit ini diberi logika satu dan bit 1 register SREG juga set, maka bisa dilakukan enable interupsi output compare B match timer/counter1.

5. bit4\_\_OCIE1A: Timer/Counter1,Output Compare A Match Interrupts Enable

Jika bit ini diberi logika satu dan bit 1 register SREG juga set, maka bisa dilakukan enable interupsi output compare A match timer/counter1.

6. bit5\_\_TICIE1: Timer/Counter 1,Input Capture Interrupt Enable

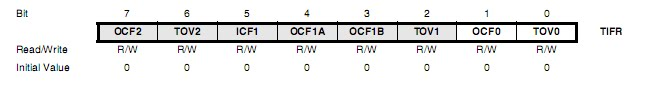
Jika bit ini diberi logika satu dan bit 1 register SREG juga set, maka bisa dilakukan enable interupsi Input Capture timer/counter1.

7. bit6\_\_TOEI2: Timer/Counter2, Overflow Interrupts Enable

Jika bit ini diberi logika satu dan bit 1 register SREG juga set, maka bisa dilakukan enable interupsi overflow timer/counter2.

8. bit7\_\_OCIE2: Timer/Counter2, OutputCompare Match Interrupts Enable

Jika bit ini diberi logika satu dan bit 1 register SREG juga set, maka bisa dilakukan enable interupsi OutputCompare Match timer/counter2.

Register TIFR (Timer Interrupts Flag Register) memiliki skema seperti ini:

1. bit0\_\_TOV0: Timer/Counter0 Overflow Flag

Bit akan bernilai satu jika timer/counter0 overflow. Bit dapat dinolkan lagi dengan memberikan logika 1 ke bit flag ini.

2. bit1\_\_OCF0: Output Compare Flag 0

Bit akan bernilai 1 jika nilai padaTimer/Counter 0 sama dengan nilai pada OCR0\_\_Output Compare Register 0.Bit dapat dinolkan lagi dengan memberi logika 1 pada bit flag ini.

3. bit2\_\_TOV1: Timer/Counter1 Overflow Flag

Bit akan bernilai satu jika timer/counter1 overflow. Bit dapat dinolkan lagi dengan memberikan logika 1 ke bit flag ini.

4. bit3\_\_OCF1B: Output Compare 1B Match Flag

Bit akan bernilai 1 jika nilai pada Timer/Counter1 sama dengan pada OCR1B\_\_Output Compare Register 1B.Bit dapat dinolkan lagi dengan memberikan logika 1 ke bit flag ini.

5. bit4\_\_OCF1A: Output Compare 1A Match Flag

Bit akan bernilai 1 jika nilai pada Timer/Counter1 sama dengan pada OCR1B\_\_Output Compare Register 1A.Bit dapat dinolkan lagi dengan memberikan logika 1 ke bit flag ini.

6. bit5\_\_ICIF1: Timer/Counter 1, Input Capture Flag

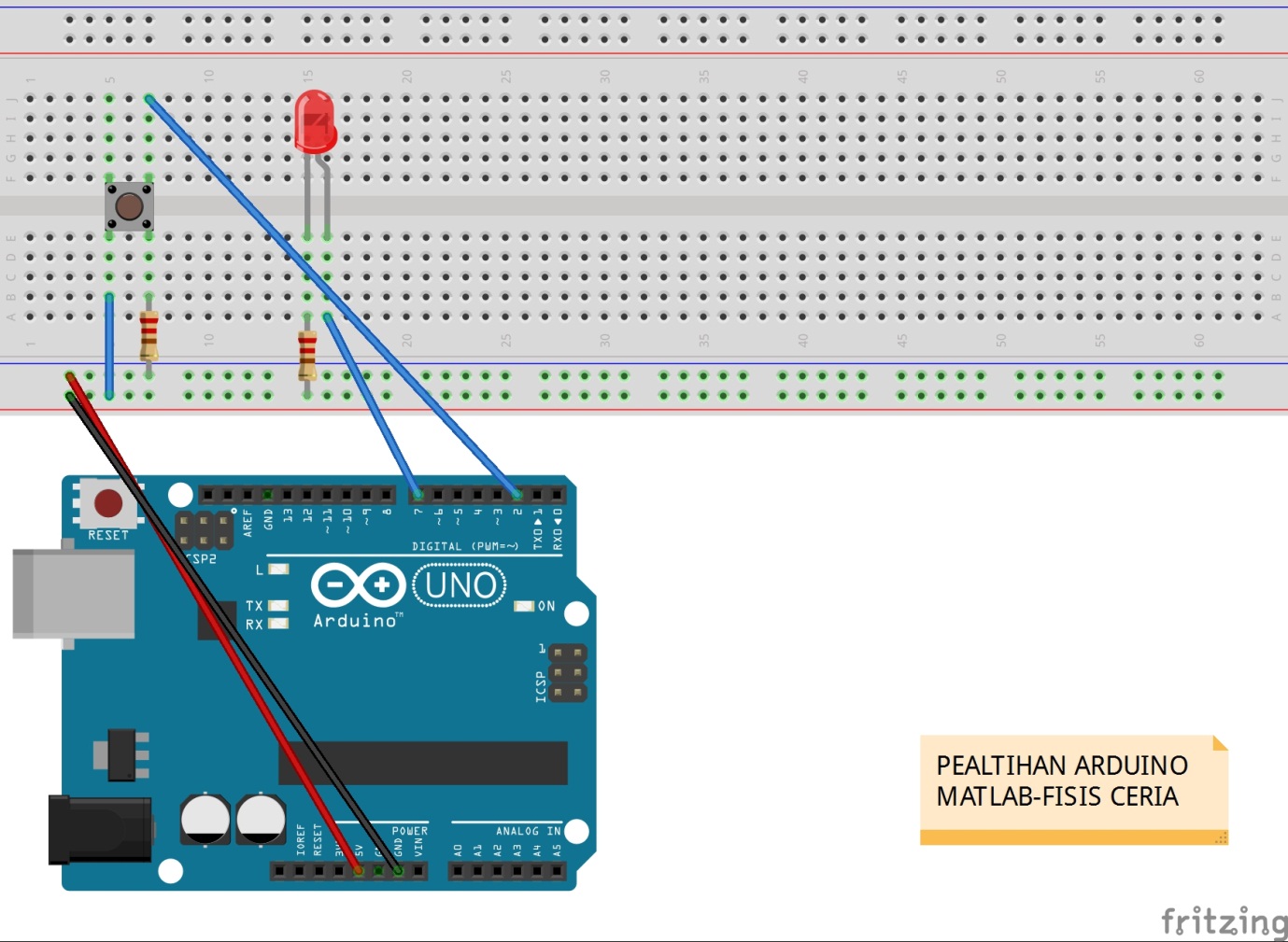
7. bit6\_\_TOV2: Timer/Counter 2 Overflow Flag

Bit akan bernilai satu jika timer/counter 2 overflow. Bit dapat dinolkan lagi dengan memberikan logika 1 ke bit flag ini.

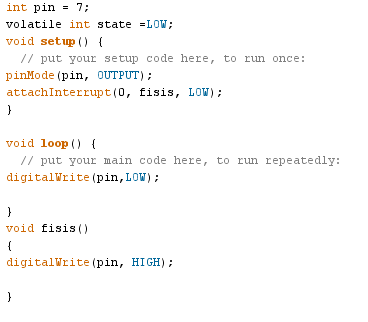
8. bit7\_\_OCF2: Output Compare Flag 2

Bit akan bernilai 1 jika nilai padaTimer/Counter 2 sama dengan nilai pada OCR0\_\_Output Compare Register 2. Bit dapat dinolkan lagi dengan memberi logika 1 pada bit flag ini.

**INTERRUPT EKSTERNAL :**

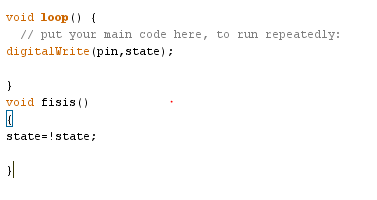


1. Hubungkan kolom paling kanan ke pin GNDArduino.
2. Hubungkan kolom selanjutnya ke pin 5V Arduino.
3. Letakkan kaki-kaki pushbutton di e5, e7, f5, f7(jika tidak sesuai, putar pushbutton 90˚) .
4. Hubungkan j7 ke pin 2 Arduino.
5. Hubungkan e7 dengan sebuah resistor 10kΩ ke 5V.
6. Tancapkan LED ke e16 (kaki panjang) dan e15.
7. Hubungkan a15 dengan sebuah resistor 330Ω ke GND.
8. Hubungkan pin 7 ke a16
9. Upload sketch dibawah ini:



**Latihan !**

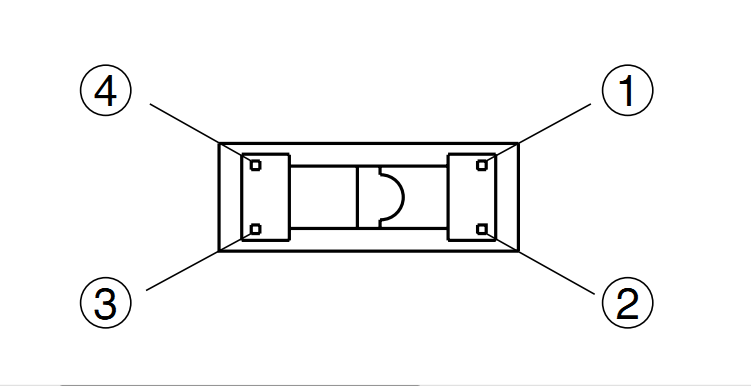
Cobalah mengganti keadaan interrupt dengan FALLING, RISING, CHANGE. Dengan merubah sketch seperti dibawah ini:



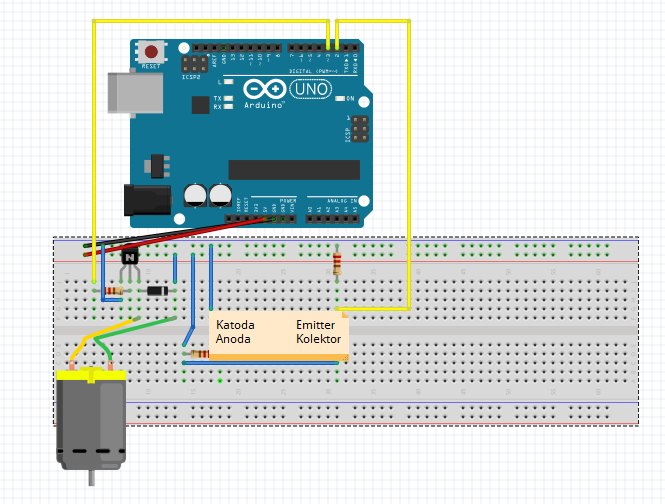
Cobalah memasang tiga LED untuk menampilkan.

Buatlah sebuah keadaan interupt yang menyatakan ketika pushbutton ditekan maka akan terjadi kedipan pada led. Gunakan delay 5000 ms, 10000 ms, dan variasikan.

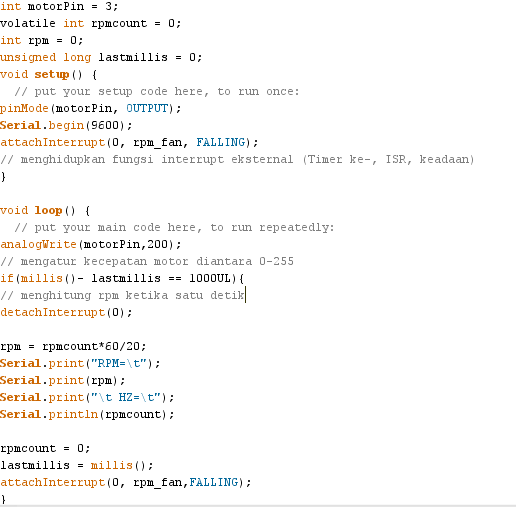
**MENGHITUNG RPM:**

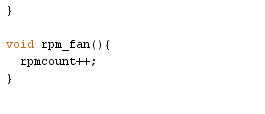


1. Katoda
2. Anoda
3. Kolektor
4. Emiter



1. Hubungkan pin GND ke kolom paling kanan.
2. Hubungkan pin 5V ke kolom berikutnya.
3. Hubungkan transistor ke j7, j8, j9.
4. Hubungkan resistor 100 kΩ diantara i4 dan i8.
5. Hubungkan diode diantara i9 dan i13.
6. Hubungkan j4 dengan pin 3.
7. Hubungkan i7 dengan GND.
8. Hubungkan j13 dengan 5V.
9. Hubugkan kaki motor DC ke f9 dan f13.
10. Hubungkan Optocupler ke breadboard (anoda ke e18, katoda ke f18, kolektor ke pin e22, dan emitter ke f22).
11. Hubungkan reisitor 220Ω diantara d18 dan d14, kemudian hubungkan e14 ke 5V.
12. Hubungkan g18 ke GND.
13. Hubungkan c14 dan c22.
14. Hubungkan resitor 4,7 kΩ diantara j22 dan GND.
15. Hubungkan pin i22 ke pin 2
16. Upload skecth dibawah ini:

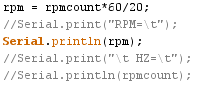




PLOT RPM

Setelah kita berhasil menghitung rpm sebuah motor DC, selanjutnya plot hasil perhitungan rpm anda dengan durasi 10 detik. Ikuti langkah-langkah dibawah ini:

1. Ubahlah sketch diatas sesuai petunjuk dibawah ini:



1. Selanjutnya buka Matlab - New M-File, tulislah *script* dibawah ini pada editor, kemudian *save* :



1. Sesuaikan port (COMxx) pada script poin 2.
2. Klik tombol “run”.

Latihan!

* Ubahlah kecepatn motor menjadi 50, 100, 150, 200, dan 250.
* Ubahlah durasi ploting menjadi 20 dan 30.

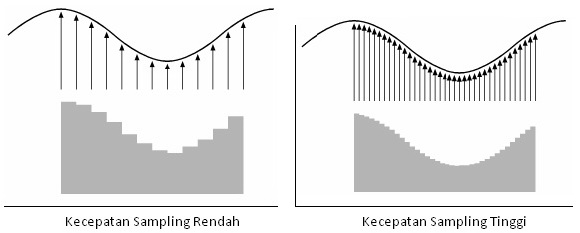
**MODUL 4**

**ADC (ANALOG TO DIGITAL CONVERSION)**

ADC adalah sebuah cara untuk mengolah sinyal, dimana kita mengkonversi sinyal analog yaitu berupa tegangan listrik menjadi sinyal digital yaitu berupa deretan angka biner. Mengapa kita perlu ADC? ADC diperlukan karena sistem yang berjalan pada sebuah mikrokontroller menggunakan sistem digital, jadi jelas tidak mungkin mengolah sinyal yang berupa data analog pada sebuah sistem digital. Maka dari itu diperlukan adanya ADC sebagai sistem pengkonversi sinyal. Karena pada kenyataannya semua proses fisis / besaran fisis bersifat analog.

ADC memiliki 2 karakter prinsip yaitu : Kecepatan Sampling (sampling rate) dan resolusi.

a. Kecepatan Sampling



b. Resolusi

Resolusi ADC menentukan ketelitian nilai hasil konversi ADC. Sebagai contoh: ADC 8 bit akan memiliki output 8 bit data digital, ini berarti sinyal input dapat dinyatakan dalam 255 (2n – 1) nilai diskrit. ADC 12 bit memiliki 12 bit output data digital, ini berarti sinyal input dapat dinyatakan dalam 4096 nilai diskrit. Dari contoh diatas ADC 12 bit akan memberikan ketelitian nilai hasil konversi yang jauh lebih baik daripada ADC 8 bit.

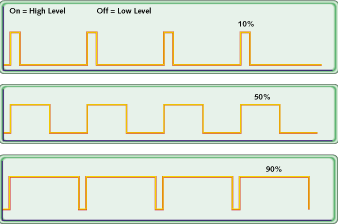
2.1 Fitur ADC pada Arduino

* ADC bekerja pada 6 Pin Analog Input A0-A5
* Resolusi ADC 10 bit 🡪 memiliki resolusi 4,9 mV/unit (untuk input tegangan 0-5 volt dikonversi ke data digital 0-1023 (hal ini dapat diubah dengan fungsi analogReference() )
* Kecepatan pembacaan data analog = 100 microseconds (0,0001 seconds) maka sampling rate = 10.000 kali/ detik
* Fungsi pembacaan data analog menggunakan fungsi analogRead ()

**MODUL 5**

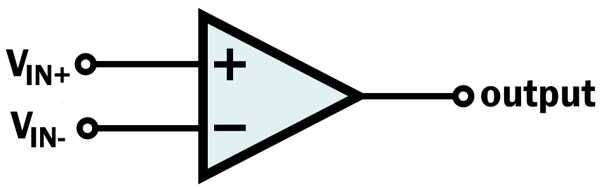
**PWM (PULSE WIDTH MODULATION)**

Pulse Width Modulation atau yang lebih sering kita sebut dengan PWM adalah sebuah cara memanipulasi lebar sinyal yang dinyatakan dengan pulsa dalam suatu perioda, untuk mendapatkan tegangan rata-rata yang berbeda. Aplikasi PWM biasanya digunakan untuk pemodulasian data di bidang telekomunikasi, pengontrolan daya atau tegangan yang masuk ke beban, regulator tegangan, audio effect dan penguatan, serta aplikasi-aplikasi lainnya. Aplikasi PWM berbasis mikrokontroler biasanya berupa pengendalian kecepatan motor DC, Pengendalian Motor Servo, Pengaturan nyala terang LED.

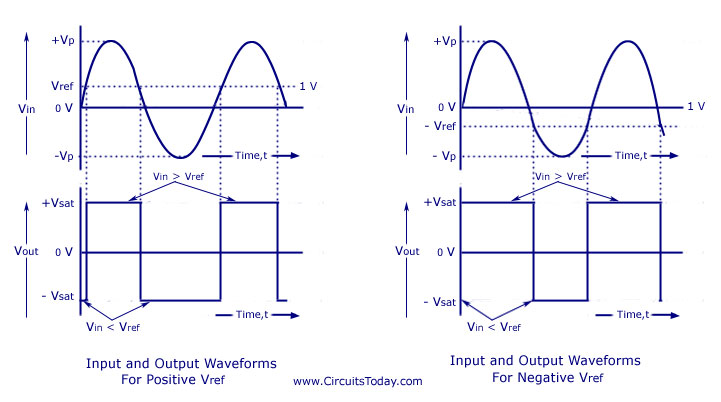


Gambar PWM

Nilai 10%, 50% dan 90% diatas merupakan nilai dari Duty Cycle. Duty Cycle adalah presentasi dari nilai sistem ketika aktif. Untuk mendapatkan suatu sinyal PWM dapat dilakukan dengan dua cara yaitu secara analog dan digital. Untuk secara analog dapat menggunakan dua buah sinyal untuk kemudian dibandingkan di comparator. Misalnya saja sinyal sinus atau sinyal gigih gergaji yang sebagai tegangan carrier dengan tegangan referensi.



Cara kerja dari komparator ini adalah seperti pada gambar dibawah ini.



Saat nilai tegangan referensi lebih besar dari tegangan carrier (sinus) maka output comparator akan bernilai high. Namun saat tegangan referensi bernilai lebih kecil dari tegangan carrier, maka output comparator akan bernilai low. Dengan memanfaatkan prinsip kerja dari komparator inilah, untuk mengubah duty cycle dari sinyal output cukup dengan mengubah-ubah besar tegangan referensi. Rumus dari duty cycle adalah sebagai berikut ini.

Duty Cycle = x 100%

Dengan menghitung duty cycle yang diberikan, akan didapat tegangan output yang dihasilkan.   


Average voltage merupakan tegangan output pada motor yang dikontrol oleh sinyal PWM. **a** adalah nilai  duty cycle saat kondisi sinyal  “on”. **b** adalah nilai  duty cycle saat kondisi sinyal “off”. V full adalah tegangan maximum pada motor. Dengan menggunakan rumus diatas, maka akan didapatkan tegangan output sesuai dengan sinyal kontrol PWM yang dibangkitkan.

Pada metode digital setiap perubahan PWM dipengaruhi oleh resolusi dari PWM itu sendiri. Misalkan PWM digital 8 bit berarti PWM tersebut memiliki resolusi 28= 256, maksudnya nilai keluaran PWM ini memiliki 256 variasi, variasinya mulai dari 0 – 255 yang mewakili duty cycle 0 – 100% dari keluaran PWM tersebut.