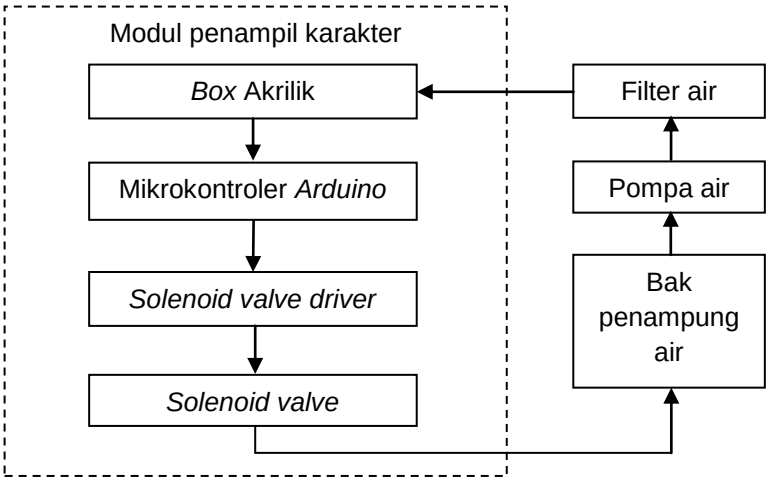
3. PERANCANGAN SISTEM

Pada bab ini akan dijelaskan mengenai perancangan sistem secara keseluruhan yang dibahas ke dalam empat sub bab yaitu desain sistem, perancangan perangkat keras/*hardware* pendukung yang akan digunakan pada sistem ini, perancangan maket sistem serta perancangan perangkat lunak/*software* berikut algoritma yang akan digunakan untuk mengolah tampilan teks dan grafik sederhana.

3.1. Desain Sistem

Sesuai dengan skema dasar sistem *waterfall sign display* yang sudah dijelaskan sebelumnya pada bab 2.1.2 maka struktur dasar ini digunakan sebagai panduan untuk membuat desain *waterfall sign display*. Sistem ini menggunakan jatuhan air yang dikendalikan melalui *solenoid valve* untuk menampilkan karakter berupa campuran teks dan grafik sederhana. Tinggi daerah tampilan sebesar 2 meter, tampilan karakternya 2 dimensi dan menggunakan 8 buah *solenoid valve* sehingga tampilan resolusi maksimumnya 8-*bit*.

Cara kerja utama dari sistem ini adalah mengatur *on-off* dari *solenoid valve* untuk tiap baris atau *frame* karakter secara terus menerus sehingga dapat air yang jatuh dapat membentuk karakter yang diinginkan.



Gambar 3.1. Blok diagram sistem *waterfall sign display*

Blok diagram ini terdiri dari 4 komponen utama yang memiliki fungsi masing-masing, yaitu modul penampil karakter, bak penampung air, pompa air dan filter air.

Modul penampil karakter adalah modul yang digunakan untuk membuat tampilan karakter-karakter dengan menggunakan air. Pada modul ini terdapat beberapa komponen antara lain modul mikrokontroler Arduino, *solenoid valve driver* dan *solenoid valve*. Modul mikrokontroler Arduino ini bertugas sebagai kontroler yang mengolah informasi yang berada di dalam program berupa *array* untuk ditampilkan melalui *solenoid valve*. Sehingga peran *solenoid valve driver* ini adalah sebagai perantara antara modul mikrokontroler dengan *solenoid valve*. Pembahasan mengenai perancangan *hardware* dapat dilihat pada bab 3.2 dan perancangan *box* dari modul penampil karakter dapat dilihat pada bab 3.3.2.1.

Bak penampungan air digunakan sebagai penampung air yang jatuh dari modul penampil karakter dan juga sebagai suplai air bagi sistem. Pembahasan mengenai perancangan bak penampung air dapat dilihat pada bab 3.3.2.2.

Pompa air digunakan untuk menyalurkan air dari bak penampung air menuju modul penampil karakter. Filter air digunakan untuk menyaring kotoran-kotoran yang terdapat pada air, *solenoid valve* ini adalah komponen yang termasuk sensitif terhadap kotoran maka peran filter sangat vital. Karena jika ada kotoran pada air ini masuk melalui *solenoid valve* akan menyumbat katup yang terdapat di dalam *solenoid valve*. Sehingga dapat membuat *solenoid valve* tidak dapat menutup dengan sempurna dan dapat mengganggu tampilan karakter.

3.2. Desain *Hardware*

Desain *hardware* yang digunakan di dalam sistem *waterfall sign display* ini terdiri dari rangkaian modul mikrokontroler Arduino, rangkaian *solenoid valve driver* dan koneksi modul mikrokontroler dengan *solenoid valve driver*.

3.2.1. *Solenoid Valve Driver*

Desain dari *solenoid valve driver* ini didasarkan pada spesifikasi *solenoid valve*. Tipe dari *solenoid valve* yang digunakan adalah VAI3430/00 buatan Invensys, spesifikasi lengkapnya adalah sebagai berikut:

- *Rated Voltage* : 220-240VAC, 110-120VAC, 24VAC & 12VAC. *Others upon request*.
- *Rated Frequency* : 50-60Hz
- *Power Draw* : 6 Watts (3 Watt DC *servo-assisted - optional*) *at* 230VAC
- *Coil Insulation* : *Class* F (140°C *Operating Temperature*)
- *Ambient Temperature* : 60 °C *maximum*.
- *Liquid Temperature* : 90°C *maximum*.
- *Operating Cycle Rate* : Tu 25°C, Tm 90°C & Tu 60°C, Tm 25°C - *Continuous* (100% ED), Tu 60°C, Tm 90°C – 3 *minutes* ON, 5 *minutes* OFF.
- *Operating Pressure Range*: 0.2 *to* 10 bar.
- *Flow Regulator* (*Optional*): 4 *to* 17 *Litres/minute* (3/4" BSP *Inlet version only*)
- *Insulation* : Class II *Fully double insulated. No grounding required.*
- *Terminals* : *Two* 6.35 x 0.8mm *male tab terminal*s. RAST 5 *and* RAST 2.5 *Optional*.
  (*VAI Solenoid Valve* 2)

Mengenai konsumsi tegangan dan daya yang digunakan pada tipe *solenoid valve* ini terdapat pada label yang tertempel pada *solenoid valve* yang dibeli, tercantum bahwa tegangan kerjanya sebesar 12VDC dan daya sebesar 3 Watt.

Gambar 3.2. Keterangan konsumsi daya dari *solenoid valve* yang digunakan

Ketika *solenoid valve* dalam keadaan menyala, arus yang mengalir pada *solenoid valve* adalah:

$$P = VI \tag{3.1.}$$

$$3Watt = (12Volt)(I) \tag{3.2.}$$

$$I = \frac{3}{12} = 0{,}25Ampere = 250mA \tag{3.3.}$$

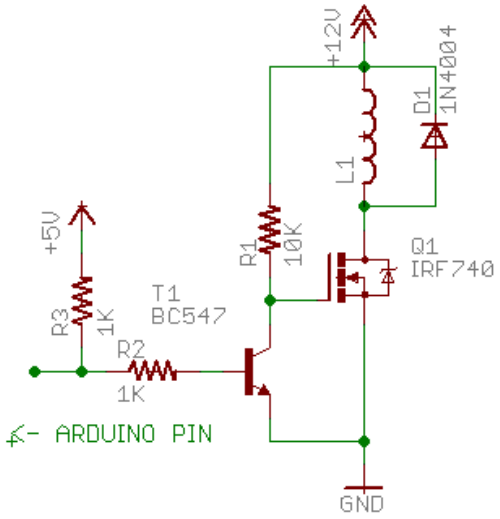
Hasil perhitungan yang didapatkan adalah 250mA, karena *solenoid valve* terpasang pada terminal *drain* dari MOSFET maka arus ini adalah arus *drain* yang disebut sebagai I_D sehingga I_D = 250mA. Total *solenoid valve* yang akan digunakan adalah 8 buah, dan rangkaian *driver* ini menggunakan satu rangkaian dasar MOSFET sebagai *switch* yang digandakan sebanyak 8 disusun secara paralel, sehingga arus *drain* total untuk *solenoid valve* yang dibutuhkan dari *power supply* adalah 2000 mA atau 2 Ampere.

Gambar 3.3. Keterangan *output power supply* yang digunakan

Pada gambar 3.3 terdapat keterangan bahwa *output* arus maksimum dari level tegangan +12V sebesar 8 Ampere secara kontinyu, sehingga *power supply* ini mampu untuk memenuhi kebutuhan suplai untuk *solenoid valve* maupun mikrokontroler Arduino (dengan level tegangan +5V). *Power supply* yang digunakan adalah *power supply* merk ASTEC yang diambil dari *box* CPU komputer, suplai tegangan yang digunakan yaitu +12V yang ditandai dengan kabel warna kuning, +5V yang ditandai dengan kabel warna merah dan *ground* yang ditandai dengan kabel warna hitam.

Sedangkan rangkaian *driver* ini menggunakan skematik dasar MOSFET sebagai *switch* seperti yang telah dijelaskan sebelumnya pada bab 2.3 dan memperbanyak rangkaian ini sebanyak 8 buah. MOSFET yang digunakan tipe IRF740 dan transistor dengan tipe BC547 sebagai rangkaian *switch* elektroniknya, dengan dioda *snubber* yang dipasang secara paralel dengan *solenoid valve* digunakan untuk mencegah arus induktif balik dengan cara membuat arus induktif balik mengalir di kumparan pada *solenoid valve* dan dioda sehingga *solenoid valve* tetap aktif selama beberapa waktu setelah dimatikan. Pengaruh dari diode *snubber* ini menambah sedikit *off delay* saat *solenoid valve* dimatikan.

MOSFET IRF740 dipilih karena murah, memiliki V_{DS} = 400V maks (*breakdown voltage* yang besar) dan I_{DS} yang cukup besar yaitu 10A. Sedangkan transistor BC547 dipilih karena harganya juga murah, memiliki tegangan saturasi V_{CE} = 45V maks, dan arus *collector* I_C = 100mA cukup untuk men-*drive gate* MOSFET. Komponen yang ditunjukkan sebagai simbol induktor pada gambar 3.4 adalah *solenoid valve*.



Gambar 3.4. Skematik dasar rangkaian *solenoid valve driver*

Gambar 3.4 menunjukkan rangkaian *solenoid valve driver* untuk satu *solenoid valve*. Transistor NPN tipe BC547 ditambahkan untuk membantu mikrokontroler Arduino mengendalikan MOSFET secara tidak langsung, karena MOSFET tipe IRF740 membutuhkan tegangan V_{GS} minimal 10V untuk mencapai kondisi $R_{DS(ON)}$ yang baik. Jika transistor berada dalam keadaan *on*, maka tegangan V_{GS} yang didapatkan adalah 12V. Pada terminal *base* dari transistor terdapat resistor *base* (R_B) yang berfungsi untuk membatasi arus pada terminal *base* dan resistor *pull-up* yang berfungsi menjaga kondisi transistor tidak akan berubah kondisi ketika *output* Arduino sedang dalam keadaan *floating* saat *reset*.

Menurut *datasheet* ATMEGA2560 arus maksimum yang dapat disuplai pada setiap I/O *pin* adalah 40mA, sehingga konsumsi arus *base* pada transistor harus diketahui agar tidak merusak *pin* karena arus berlebih (*ATMEGA2560 Datasheet* 367).

Berikut ini diberikan transistor berada dalam kondisi *saturation*. Tegangan *base-to-emitter* (V_{BE}) 0,7V tipikal, ketika pin Arduino memberikan tegangan 5V dan nilai resistor 1kΩ maka nilai arus *base* dapat dicari sebagai berikut:

$$V_{Rbase} = I_{base} * R_{base} \tag{3.4.}$$

$$V_{in} - V_{BE} = I_{base} * 1000\Omega \tag{3.5.}$$

$$I_{base} = \frac{(5V - 0,7V)}{1000\Omega} = 4,3mA \tag{3.6.}$$

Dari hasil perhitungan didapatkan nilai arus *base* sebesar 4,3mA sehingga konsumsi arus dari *pin* mikrokontroler juga sebesar 4,3mA. Jika dicocokkan dengan ketentuan arus maksimal sesuai dengan *datasheet*, arus *output* berada dalam batas aman.

Dalam keadaan saturasi tegangan V_{CE} = 0,2V tipikal, tegangan pada terminal *base* 12V, resistor *pull-up* di antara terminal *collector* dan *gate* MOSFET sebesar 10kΩ, sehingga arus *collector* dapat dicari sebagai berikut:

$$V_{Rcollector} = I_{collector} * R_{collector} \tag{3.7.}$$

$$V_{in} - V_{CE} = I_{collector} * 10000\Omega \tag{3.8.}$$

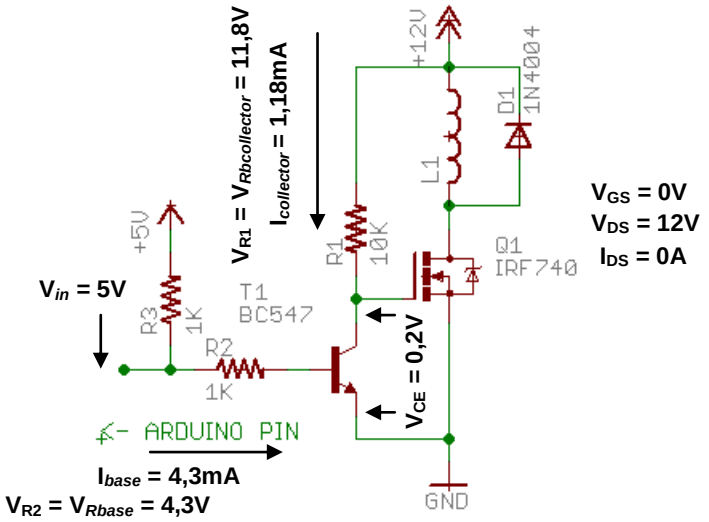
$$I_{collector} = \frac{(12V - 0,2V)}{10000\Omega} = 1,18mA \tag{3.9.}$$

Berikut ini diberikan kondisi transistor berada dalam kondisi *cut-off*. Arus *base* nilainya kecil sekali hingga mendekati nol (karena dapat dikatakan tidak ada arus yang mengalir), maka transistor tidak aktif/*open circuit*. Sehingga pada terminal *collector* arus yang mengalir menuju terminal *gate* MOSFET juga sangat kecil.

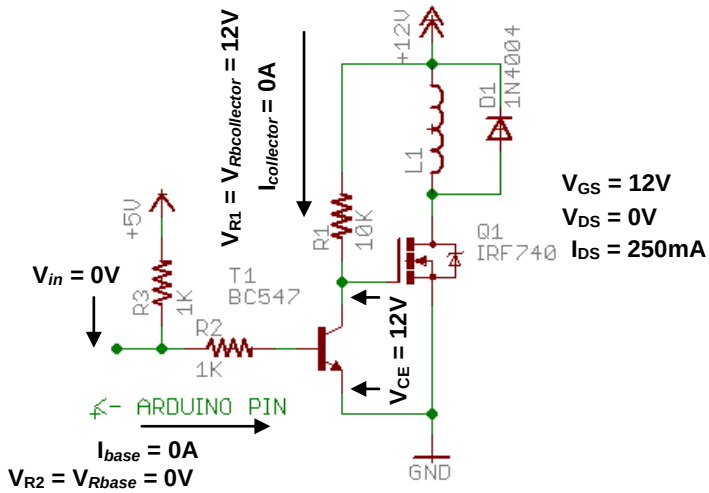
$$V_{in} - V_{CE} = 0 * 10000\Omega \tag{3.10.}$$

$$V_{in} = V_{CE} = 12V \tag{3.11.}$$

Arus *collector* ini mengalir saat transistor dalam keadaan saturasi transistor akan bekerja seperti saklar tertutup. Kemudian terminal *gate* MOSFET yang terhubung dengan terminal *collector* dari transistor terhubung dengan *ground*, mengakibatkan MOSFET tidak aktif karena V_{GS} = 0V dan MOSFET akan bekerja seperti saklar terbuka. Sedangkan arus *collector* tidak mengalir saat transistor dalam keadaan *cut-off* dan transistor akan bekerja seperti saklar terbuka, terminal *gate* MOSFET mendapatkan tegangan +12V karena resistor *pull-up* mengakibatkan MOSFET aktif karena V_{GS} = +12V, MOSFET akan bekerja seperti saklar tertutup.



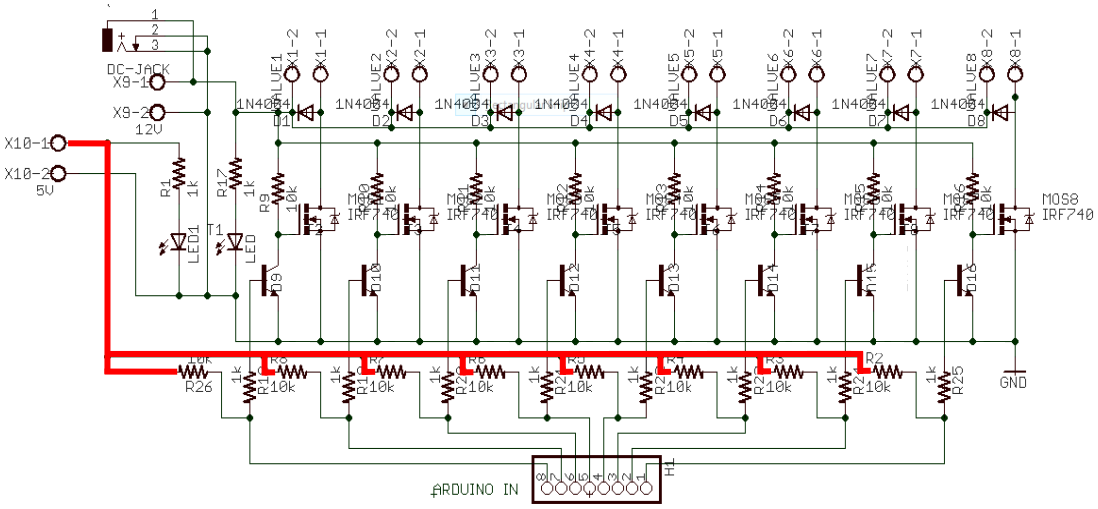
Gambar 3.5. Mekanisme rangkaian *solenoid valve driver* dengan input *logic* 1



Gambar 3.6. Mekanisme rangkaian *solenoid valve driver* dengan input *logic* 0

Dari hasil perhitungan ini didapatkan karakteristik bahwa pada saat diberikan tegangan 5V di sisi *input*, maka MOSFET menjadi *off* dan *solenoid valve* tidak akan menyala. Ketika diberikan tegangan 0V di sisi *input*, maka MOSFET menjadi *on* dan *solenoid valve* menyala. Sehingga sifat rangkaian ini adalah *active low*, kemudian sifat ini dapat dijadikan panduan untuk membuat gambar *monochrome bitmap* berdasarkan *logic active low*.

Rangkaian ini selanjutnya dijadikan rangkaian lengkap dengan memperbanyak sebanyak 8 buah yang seluruhnya disusun secara paralel. Dalam rangkaian ini terdapat dua jenis tegangan yaitu +12V dan +5V, tegangan +12V digunakan untuk suplai *solenoid valve* dan MOSFET, sedangkan +5V untuk resistor *pull-up* dari *input* rangkaian yang terhubung dengan Arduino.



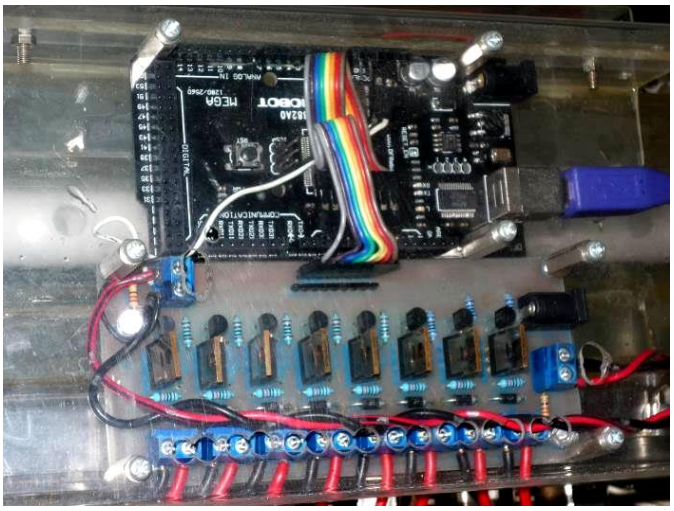
Gambar 3.7. Skematik lengkap rangkaian *solenoid valve driver* dan jalur tegangan +5V untuk resistor *pull-up* yang ditandai dengan garis warna merah

3.2.2. Koneksi Modul Mikrokontroler dengan *Solenoid Valve Driver*

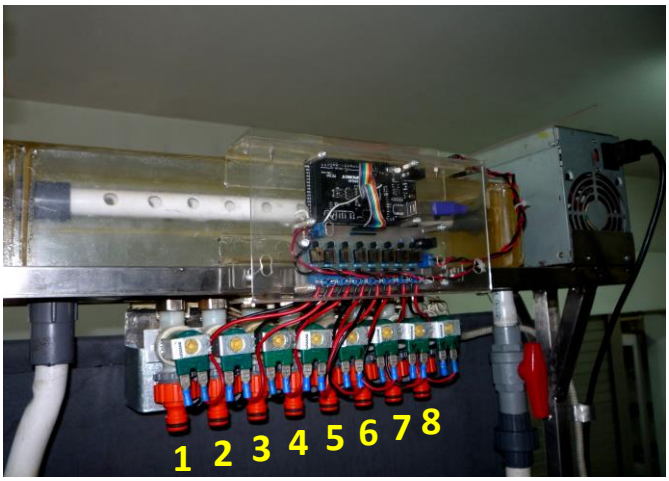
Pada modul mikrokontroler DFRduino ini *digital port* 2-9 dipilih sebagai *port* untuk koneksi dengan *solenoid valve driver*. *Digital port* 0 dan 1 tidak dipakai karena merupakan TX/RX *default* dan akan mengganggu fungsi *port* tersebut jika dipakai sebagai I/O. Mengacu pada desain *port mapping* untuk modul Arduino MEGA2560 yang terdapat pada bab 2 yaitu gambar 2.12, *digital port* 2-9 diambil masing-masing dari PortE.4, PortE.5, PortG.5, PortE.3, dan PortH.3–PortH.6. Meskipun berbeda posisi tetap bisa dipakai berurutan karena

sudah diatur sesuai dengan *pin mapping* yang terdapat pada modul Arduino MEGA2560.

Koneksi modul mikrokontroler dengan *driver* dilakukan dengan menghubungkan *digital port* 2-9 langsung dengan bagian *input* dari *driver* secara urut, dimana *digital pin* 2 dihubungkan dengan *driver input* urutan terakhir sebagai LSB/*Least Significant Bit* dan *digital pin* 9 dengan *driver input* urutan pertama sebagai MSB/*Most Significant Bit*. Posisi LSB adalah *solenoid valve* ke 1 sedangkan posisi MSB adalah *solenoid valve* ke 8. Masing-masing ditandai dengan kabel berwarna abu-abu dan kabel berwarna coklat pada gambar 3.8.



Gambar 3.8. Koneksi modul mikrokontroler dengan *solenoid valve driver*

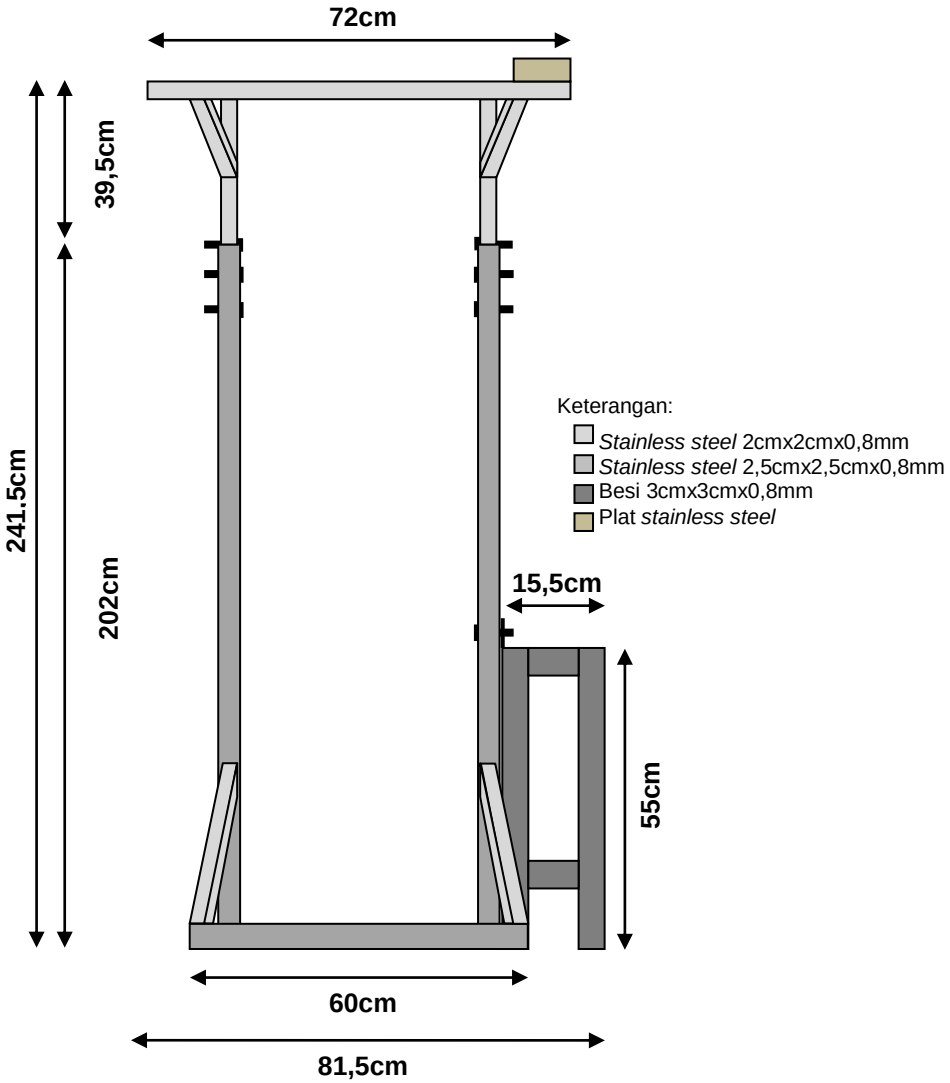


Gambar 3.9. Rangkaian modul penampil karakter dan *power supply* beserta urutan posisi *solenoid valve* yang digunakan

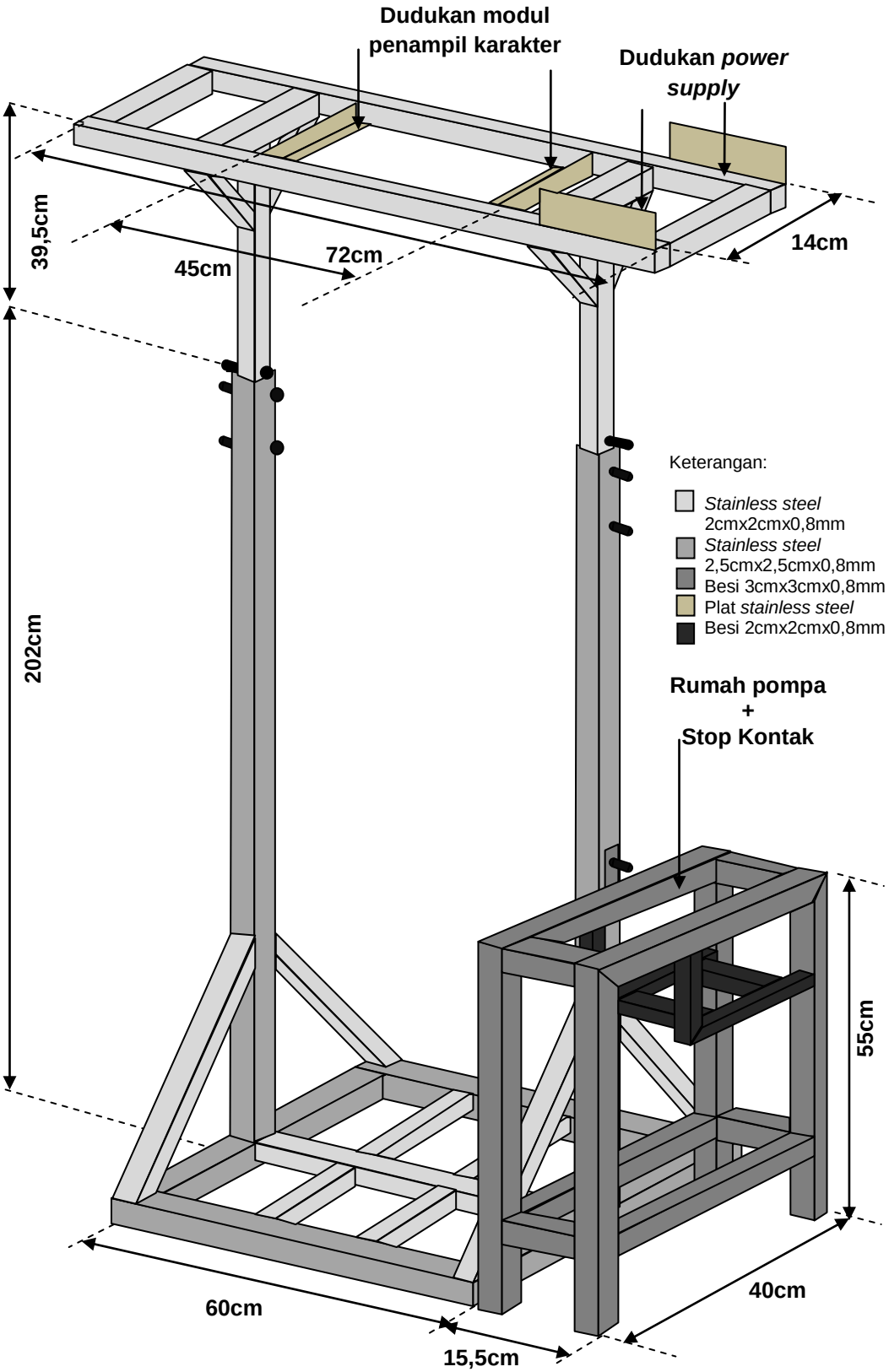
3.3. Desain Maket

3.3.1. Perancangan Desain Kerangka Maket

Kerangka maket yang digunakan untuk *waterfall sign display* ini memiliki dimensi total yaitu panjang 81,5cm, lebar 40cm, dan tinggi 241,5cm. Bahan kerangka utama yang digunakan untuk maket ini menggunakan batangan *stainless steel* dengan dua buah ukuran yaitu panjang dan lebarnya 2x2cm dengan tebal 0,8mm dan 2,5cmx2,5cm dengan tebal 0,8mm. Bahan kerangka untuk rumah pompa menggunakan batangan besi dengan ukuran panjang dan lebar 3cmx3cm dengan tebal 0,8mm.



Gambar 3.10. Desain kerangka maket beserta dimensinya tampak sisi depan (gambar tidak untuk diskala)

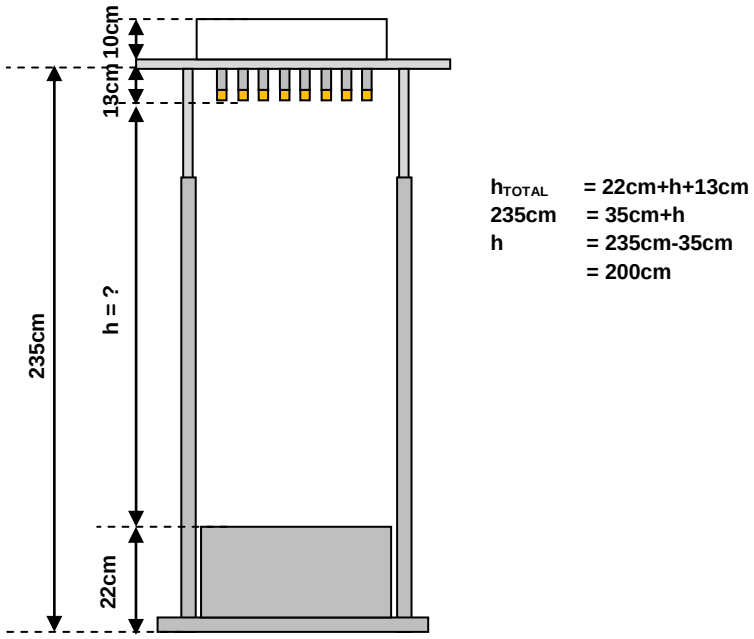


Gambar 3.11. Desain kerangka maket beserta dimensinya tampak perspektif (gambar tidak untuk diskala)

Pada kerangka sisi atas yang ditunjukkan di dalam gambar 3.10. ditahan menggunakan 6 buah baut panjang. Karena ukuran batang *stainless steel* yang digunakan untuk kerangka sisi atas memiliki dimensi berbeda dengan kedua tiang yang menopangnya. Dimana ukuran batang *stainless steel* yang digunakan untuk tiang berukuran 2,5cmx2,5cm tebal 0,8mm sedangkan untuk kerangka sisi atas berukuran 2cmx2cm dengan tebal yang sama 0,8mm (keterangan warna untuk gambar 3.11. sama dengan yang terdapat pada gambar 3.10. dengan beberapa tambahan). Kerangka sisi atas digunakan untuk menaruh modul penampil karakter yang berada di tengah dan *power supply* yang ditempatkan di sisi kanan kerangka.

Pada kerangka sisi bawah terdapat susunan kerangka yang digunakan untuk bak penampung air. Kerangka samping yang berukuran lebih kecil daripada kerangka utama digunakan sebagai rumah pompa dan terdapat tempat untuk menaruh stop kontak ditunjukkan dengan warna abu-abu gelap pada gambar 3.11.

Tinggi total daerah tampilan karakter yang didapatkan adalah 200cm, dari sisi bawah diukur dari mulut bak penampung air yang memiliki tinggi 22cm. Sedangkan dari sisi atas diukur dari sisi bawah *box* dari modul penampil karakter (diukur dari sambungan *box* akrilik dengan *solenoid valve* sampai mulut *nozzle*) yang memiliki tinggi 13cm.



Gambar 3.12. Perhitungan tinggi daerah tampilan (gambar tidak untuk diskala)

3.3.2. Perancangan Komponen-komponen Maket

3.3.2.1. Perancangan *Box* Pada Modul Penampil Karakter

Box yang terdapat pada modul penampil karakter seperti yang terlihat pada gambar 3.9. dibuat menggunakan akrilik transparan dengan tebal 5mm dan memiliki volume total sebesar 3168cm^3, dimana proses perancangannya adalah sebagai berikut.

Nozzle yang digunakan memiliki diameter dalam sebesar 0,9cm dan tinggi tampilan yang diinginkan adalah 2 meter. Sehingga dapat diibaratkan air yang keluar melalui *nozzle* dari satu *solenoid valve* ini adalah sebuah "tabung" dengan volume sebagai berikut:

$$V = \pi r^2 h \tag{3.12.}$$

$$V = 3{,}14\left(\frac{0{,}9}{2}\right)^2 (200) = 127{,}17 cm^3 \tag{3.13.}$$

Jumlah *nozzle* yang digunakan juga sama dengan jumlah *solenoid valve* yang digunakan, yaitu 8 buah. Total volume yang didapatkan adalah hasil volume yang didapatkan pada persamaan 3.13. dan mengalikannya dengan 8, sehingga volume akhir yang didapatkan adalah 1017,36cm^3.

Oleh karena ada beberapa pengurangan volume akibat sambungan pipa yang masuk di dalam *box* maka sebaiknya desain *box* memiliki volume yang lebih besar daripada yang sudah dihitung, sebagai contoh dua kali lipat atau lebih.

Panjang dan lebar *box* sudah ditentukan pada bab 3.3.1 agar cocok dengan ukuran dudukan modul penampil karakter pada kerangka. Karena panjang, lebar dan tinggi *box* diketahui maka volume *box* dapat dicari:

$$V = plt \tag{3.14.}$$

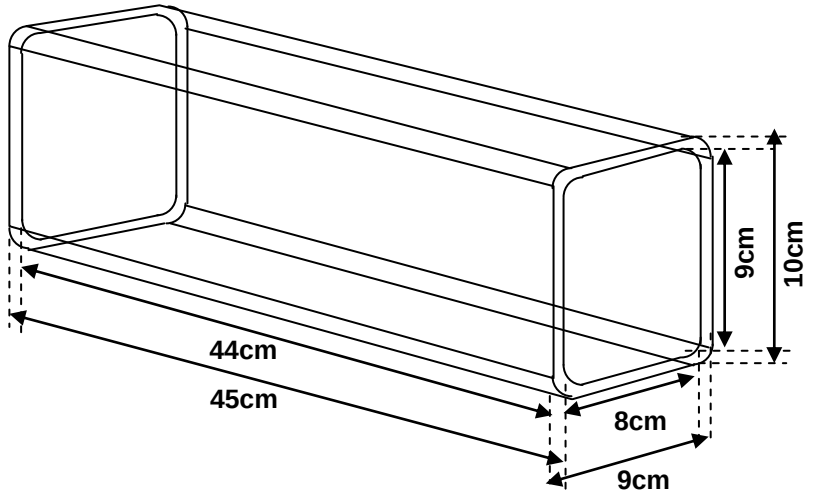
$$V = (45)(9)(10) = 4050 cm^3 \tag{3.15.}$$

Akrilik yang digunakan untuk membuat *box* memiliki tebal 5mm (0,5cm), maka volume sebenarnya yang didapatkan tidaklah 4050cm^3, melainkan di bawah itu karena yang dihitung adalah volume dalam *box* dan bukan volume luar *box*. Sehingga dapat dikatakan bahwa panjang sebelah dalam menjadi 44cm, lebarnya

8cm dan tingginya 9cm (masing-masing dikurangi 1cm, yaitu 0,5cm dari kiri dan 0,5cm dari kanan). Maka volume akhirnya menjadi:

$$V = plt \tag{3.16.}$$

$$V = (44)(8)(9) = 3168cm^3 \tag{3.17.}$$



Gambar 3.13. Perancangan *box* untuk modul penampil karakter

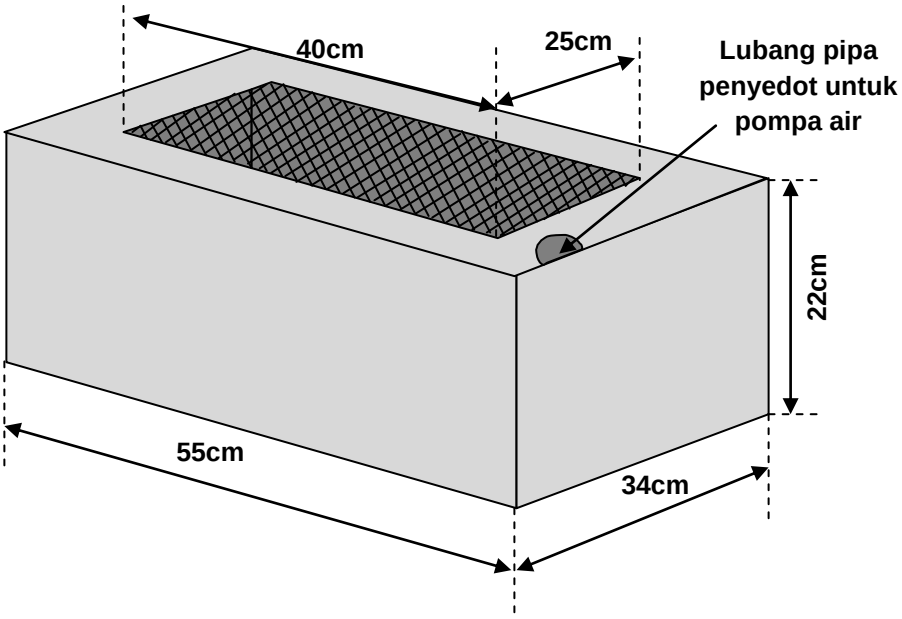
3.3.2.2. Perancangan Bak Penampung Air

Bak penampung air berguna sebagai penangkap air yang jatuh dari modul penampil karakter dan juga sebagai suplai air untuk sistem *waterfall sign display*. Volume bak juga harus diperhitungkan mengingat pipa-pipa sambungan yang terdapat pada sistem *waterfall sign display* ini merupakan benda bervolume. Sehingga bak ini juga dapat memberikan suplai air yang cukup bagi pipa-pipa sambungan maupun modul penampil karakter itu sendiri.

Bak penampung air dibuat dengan menggunakan plat galvalum, bahan ini dipilih karena sifatnya yang lentur, ringan, dan mudah dibentuk. Tidak lupa juga bahan pelapis anti bocor diaplikasikan terhadap sisi dalam dari bak ini agar dapat mencegah kebocoran air.

Panjang dari bak penampung air sudah dirancang dengan ukuran 55cm dan lebarnya 34cm serta memiliki tinggi 22cm, sehingga memiliki volume total 41140cm^3. Karena bak ini dibuat dari plat galvalum tipis maka tebal plat dapat

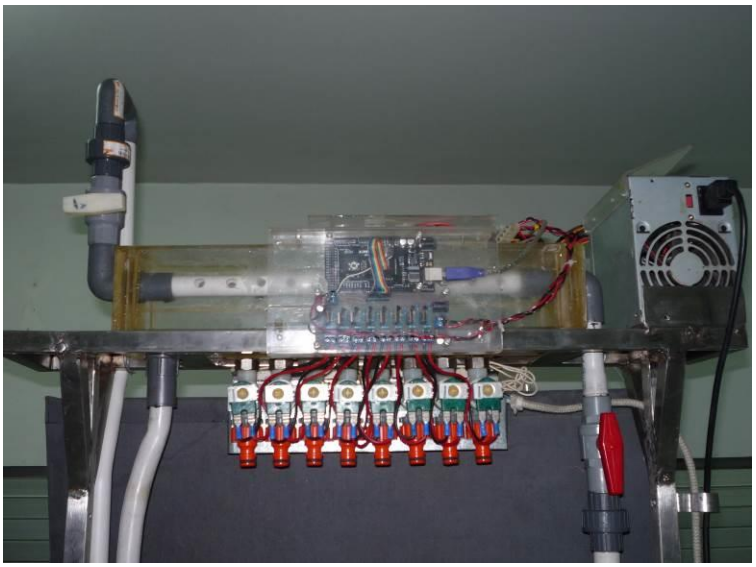
diabaikan, ukuran ini sudah ditentukan agar cocok dengan desain kerangka. Selain itu bak penampung air ini juga akan memiliki bentuk berupa balok berongga yang memiliki lubang di tengahnya agar air yang jatuh dapat ditampung oleh bak penampung air, bentuk ini dapat diumpamakan seperti kotak *tissue*. Pada lubang ini juga ditempatkan lembaran kasa nyamuk agar meminimalkan semburan air dari dalam bak penampung air ketika air jatuh dari modul penampil karakter.



Gambar 3.14. Perancangan bak penampung air

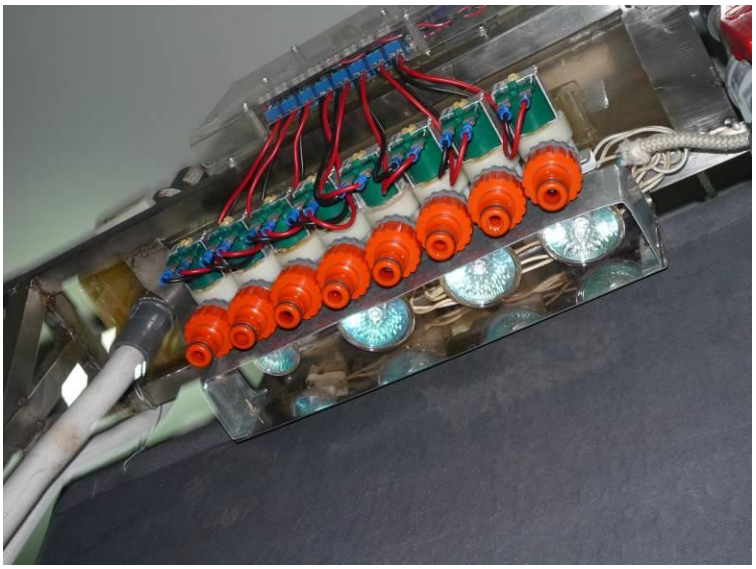
3.3.3. Penataan Letak Komponen-komponen Pada Kerangka Maket

Kerangka maket ini digunakan untuk menaruh komponen-komponen dari *waterfall sign display* ini. Beberapa komponen yang akan diletakkan di antaranya adalah modul penampil karakter, bak penampung air, *power supply*, lampu halogen, filter air, pompa air, serta stop kontak.



Gambar 3.15. Penempatan modul penampil karakter pada sisi atas kerangka maket bersamaan dengan *power supply*

Gambar 3.16. Penempatan bak penampung air pada sisi bawah kerangka maket



Gambar 3.17. Penempatan lampu halogen pada sisi belakang *solenoid valve*

Gambar 3.18. Penempatan filter air pada sisi atas rumah pompa

Gambar 3.19. Penempatan pompa air dan stop kontak pada sisi dalam rumah pompa

Gambar 3.20. Maket *waterfall sign display* secara keseluruhan

3.3.4. Faktor Desain yang Berpengaruh Terhadap Tampilan Karakter

3.3.4.1. Pengaruh Tekanan Pompa Terhadap Tampilan Karakter

Karakter yang ditampilkan melalui *waterfall sign display* ini membutuhkan distribusi air terus menerus, maka pompa digunakan sebagai sarana distribusinya. Karena pompa air memanfaatkan putaran *impeller* (bentuknya seperti baling-baling, fungsinya untuk menambah tekanan air) pada *rotor* pompa agar air dapat dialirkan ke ketinggian tertentu. Tetapi putaran *impeller* menyebabkan air yang dipompa menjadi tidak *laminar* (alirannya halus dan seragam), melainkan *turbulence* (alirannya tidak beraturan bentuknya) akibat putaran *impeller* yang berkecepatan tinggi. *Box* yang dirancang memiliki volume kecil tidak dapat mengatasi aliran air yang mengalami *turbulence* ini sehingga pengaruh tekanan pompa harus diperhatikan terlebih dahulu. Pengaruh aliran air yang mengalami *turbulence* ini adalah aliran air yang keluar melalui *solenoid valve* tidak halus bentuknya, sehingga mempengaruhi kualitas tampilan karakter.

Tekanan pompa ini diperlukan mengingat spesifikasi *solenoid valve* (yang sudah dibahas pada bab 3.2.2) memiliki tekanan kerja yang berada pada *range* 0,2-10bar. Jika tekanan kerja tidak mencukupi atau berlebih, maka *solenoid valve* tidak akan bekerja semestinya. Karena ada tekanan yang bekerja terhadap *solenoid valve*, maka ada kecepatan awal saat air keluar melalui *solenoid valve*.

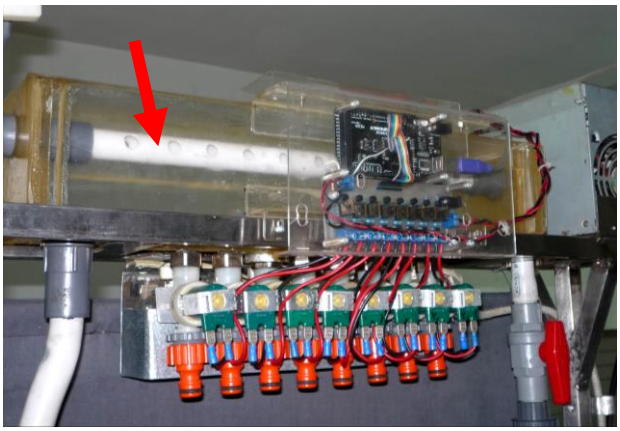
Besarnya tekanan pompa dicari dengan cara bertahap sebagai berikut. Pada saat pompa dinyalakan, air akan dipompa ke dalam *box* akrilik sampai penuh. Waktu yang dibutuhkan ketika *box* dalam keadaan kosong sampai penuh adalah 18 detik (diukur dengan menggunakan *stopwatch*) dan ketinggian airnya 6,5cm. Volume dari *box* yaitu 3168cm^3, sehingga *flow rate* yang didapatkan adalah:

$$Q = \frac{V}{t} = \frac{3168cm^3}{18s} = 176cm^3s^{-1} \qquad (3.18.)$$

Dari hasil perhitungan didapatkan *flow rate* sebesar 176cm^3 per detik. Kecepatan air yang mengalir juga dapat dicari dengan menggunakan rumus *flow rate* yang dinyatakan sebagai berikut.

$$Q = vA \qquad (3.19.)$$

Dimana v adalah kecepatan dan A adalah luas penampang. Aliran air melewati pipa berlubang yang terhubung dengan sisi masukan air dari pompa pada *box*. Lubangnya berbentuk lingkaran dengan diameter 0,5cm dengan jumlah 16 lubang, sehingga luas penampang totalnya adalah 3,14cm^2.



Gambar 3.21. Pipa berlubang sebagai tempat keluaran air

Setelah *flow rate* dan luas penampangnya diketahui, maka kecepatan aliran airnya dapat dicari:

$$176cm^3s^{-1} = v * 3{,}14cm^2 \tag{3.20.}$$

$$v = \frac{176cm^3s^{-1}}{3{,}14cm^2} = 56{,}05cms^{-1} = 0{,}56ms^{-1} \tag{3.21.}$$

Dari hasil perhitungan didapatkan bahwa kecepatan aliran air yang melewati lubang-lubang pada pipa tersebut adalah 0,56ms^{-1}. Menurut Jukka untuk membuktikan bahwa aliran air *laminar* atau mengalami *turbulence*, digunakan konstanta Reynolds sebagai indikatornya (dalam *Darcy* 3):

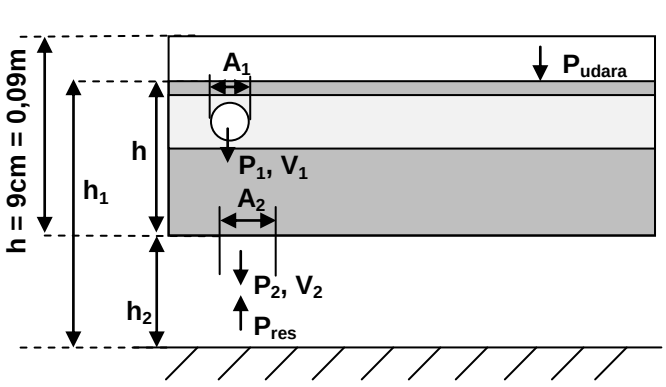
$$\mathrm{Re} = \frac{vD}{\upsilon} \tag{3.22.}$$

Dimana v adalah kecepatan aliran zat cair, D adalah diameter dari lubang keluaran zat cair dan υ adalah *kinematic viscosity* dari zat cair. Karena air adalah zat cair yang mengalir di dalam pipa, maka υ yang digunakan adalah υ dari air yaitu 1,004*10^{-6} m^{2}s^{-1} pada suhu 25°C. Diameter total dari 16 lubang pada pipa adalah 0,08m. Sehingga konstanta Reynolds dihitung sebagai berikut:

$$\mathrm{Re} = \frac{(0{,}56 ms^{-1})(8 * 10^{-2} m)}{1{,}004 * 10^{-6} m^2 s^{-1}} = 44820{,}72 \qquad (3.23.)$$

Menurut Jukka apabila nilai dari konstanta Reynolds di bawah 2300, maka aliran air tersebut laminar. Sedangkan jika nilai konstanta Reynolds di atas 4000, maka aliran air tersebut mengalami *turbulence* (dalam *Darcy* 3). Karena konstanta Reynolds yang didapatkan adalah 44820,72 (lebih besar daripada 4000), maka aliran air dari pipa berlubang mengalami *turbulence*.

Kondisi perhitungan ini adalah ketika *box* dalam keadaan tertutup, sehingga tekanan di dalam *box* dapat dicari dengan cara sebagai berikut:



h_1 = 221,5cm = 2,215m
h_2 = 215cm = 2,15m
$h = h_1 - h_2$ = 0,065m
$A_1 = 16*(3{,}14*(0{,}5/2)) = 3{,}14cm^2$
$= 3{,}14*10^{-4}\ m^2$
$A_2 = 3{,}14*(0{,}9/2)^2 = 0{,}636cm^2$
$= 6{,}36*10^{-5}\ m^2$
$A_1*V_1 = A_2*V_2$
$V_2 = (A_1*V_1)/A_2$
$V_2 = ((3{,}14*10^{-4}\ m^2)*0{,}56ms^{-1})/(6{,}36*10^{-5}\ m^2)$
$V_2 = 2{,}76ms^{-1}$
$P_{udara}*V_{udara} = n*R*T$
$P_{udara} = (1\ mol*0{,}082*(273+25)K)/V_1$
$P_{udara} = (24{,}436)/(0{,}88) = 27{,}76atm$
$P_{udara} = 2812782Pa$
$P_2 = (P_{udara} - P_{res}) + \rho gh$

Gambar 3.22. Tekanan aliran air melewati *solenoid valve* saat keran tertutup (gambar tidak untuk diskala)

Konstanta Reynolds dari air yang keluar (dengan kecepatan v_2) melalui lubang *solenoid valve* adalah:

$$\mathrm{Re} = \frac{(2{,}76 ms^{-1})(9 * 10^{-3} m)}{1{,}004 * 10^{-6} m^2 s^{-1}} = 24900{,}39 \qquad (3.24.)$$

Tekanan P_1 dicari dengan menggunakan *head total* dari pompa, dimana *head* ini adalah rugi-rugi yang disebabkan oleh tekanan pompa ("Converting Pump Head to Pressure" par. 3):

$$P = 0{,}098 * h_{TOTAL} * SG \qquad (3.25.)$$

$$h_{TOTAL} = h_{LOSS} + h_g \tag{3.26.}$$

Sedangkan *head loss* dihitung dengan menggunakan persamaan Darcy-Weisbach ("ESE/Hydrology/Pipes/Head Loss", par. 1):

$$h_{LOSS} = \frac{f_D * L * v^2}{2 * D * g} \tag{3.27.}$$

Dimana SG adalah *specific gravity*, h_g adalah *geodetic head*, f_D adalah koefisien friksi Darcy dan L adalah panjang pipa. Menurut Jukka nilai dari f_D (koefisien friksi Darcy) dicari dengan menggunakan persamaan Colebrook sebagai berikut (dalam *Darcy* 2):

$$f_D = \frac{0,25}{\left(\log\left(\frac{\frac{e}{D}}{3,7} + \frac{5,74}{(\mathrm{Re})^{0,9}}\right)\right)^2} \tag{3.28.}$$

Dimana e adalah *internal roughness* dari pipa yang digunakan. Karena jenis pipa yang digunakan adalah PVC, maka nilai e adalah 0,005mm. Sehingga koefisien friksi f_D dicari sebagai berikut:

$$f_D = \frac{0,25}{\left(\log\left(\frac{\frac{0,005*10^{-3}m}{80*10^{-3}m}}{3,7} + \frac{5,74}{(44820,72)^{0,9}}\right)\right)^2} \tag{3.29.}$$

$$f_D = \frac{0,25}{\left(\log\left(1,69*10^{-5} + 3,74*10^{-4}\right)\right)^2} \tag{3.30.}$$

$$f_D = \frac{0,25}{(-3,407)^2} = 0,0215 \tag{3.31.}$$

Dari hasil perhitungan didapatkan bahwa nilai koefisien friksi Darcy adalah 0,0215. Nilai koefisien ini selanjutnya digunakan untuk mencari *head loss*:

$$h_{LOSS} = \frac{0,0215 * 2,48m * (0,56ms^{-1})^2}{2 * (8*10^{-2}m) * 10ms^{-2}} = 0,011m \tag{3.32.}$$

Geodetic head diukur berdasarkan perbedaan level air pada sisi bawah terhadap permukaan air tertinggi pada sisi atas. Sehingga besarnya h_g adalah 2,48m saat diukur dari kedalaman air pada bak penampung air hingga permukaan air di dalam *box* akrilik. Kemudian *head total* dapat dihitung sebagai berikut:

$$h_{TOTAL} = 0{,}011m + 2{,}48m = 2{,}491m \quad (3.33.)$$

Karena *head total* sudah diketahui dan nilai *specific gravity* dari air yang diketahui adalah 1, maka tekanan P_1 dapat dicari:

$$P = 0{,}098 * 2{,}491 * 1 = 0{,}244bar = 24{,}4kPa \quad (3.34.)$$

Sedangkan tekanan pada P_2 dipengaruhi oleh beberapa faktor. Antara lain tekanan udara pada permukaan air, tekanan resistansi udara (mengarah ke atas terhadap lubang *solenoid valve*) dan tekanan air pada dasar *box*. Untuk mencari nilai P_2 menggunakan persamaan Bernoulli:

$$\tfrac{1}{2}\rho V_1^2 + \rho g h_1 + P_1 = \tfrac{1}{2}\rho V_2^2 + \rho g h_2 + P_2 \quad (3.35.)$$

$$P_2 = P_1 + \tfrac{1}{2}\rho(V_1^2 - V_2^2) + \rho g(h_1 - h_2) \quad (3.36.)$$

$$(P_{udara} - P_{res}) + \rho g h = P_1 + \tfrac{1}{2}\rho(V_1^2 - V_2^2) + \rho g h \quad (3.37.)$$

Tekanan resistansi udara dapat dicari dengan persamaan berikut:

$$P_{res} = P_{udara} - P_1 - \tfrac{1}{2}\rho(V_1^2 - V_2^2) \quad (3.38.)$$

Pada persamaan 3.38 seluruh variabelnya sudah diketahui sehingga tekanan resistansi udara terhadap *solenoid valve* dapat dicari:

$$P_{res} = 2812782Pa - 24400Pa - \tfrac{1}{2}(1000kgm^{-3})(V_1^2 - V_2^2) \quad (3.39.)$$

$$P_{res} = 2788382Pa - 500kgm^{-3}(0{,}56m^2s^{-2} - 2{,}76m^2s^{-2}) \quad (3.40.)$$

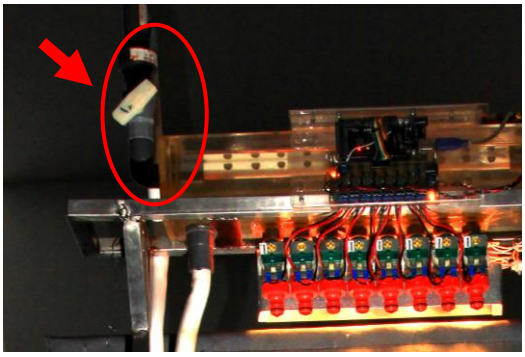
$$P_{res} = 2788382Pa - (-3652Pa) = 2792034Pa \quad (3.41.)$$

Sehingga tekanan yang dialami oleh *solenoid valve* saat membuka adalah:

$$P_2 = 2812782Pa - 2792034Pa + (1000kgm^{-3} * 10ms^{-2} * 0{,}065m) \quad (3.42.)$$

$$P_2 = 20784Pa + 650Pa = 21398Pa = 0{,}21bar \quad (3.43.)$$

Untuk mengurangi efek *turbulence* di dalam *box* maka tekanan udara harus dikurangi. Caranya adalah dengan membuka keran warna putih yang berada di sisi kiri atas dari *box* modul penampil karakter, ketika keran dibuka maka sebagian udara yang sebelumnya berada di dalam *box* dan aliran air akan turun lewat pipa menuju bak penampung air.



Gambar 3.23. Pengaturan tekanan udara secara manual dengan membuka keran

Ketika keran ini dibuka ternyata waktu pengisiannya menjadi lebih lama bila dibandingkan dengan pada saat keran tertutup. Waktu pengisian air yang dibutuhkan sampai penuh adalah 28 detik dan ketinggian airnya 6cm. Sehingga *flow rate* yang didapatkan adalah:

$$Q = \frac{V}{t} = \frac{3168cm^3}{28s} = 113{,}14cm^3s^{-1} \quad (3.44.)$$

Karena *flow rate* dan luas penampang lubang pada pipa diketahui, maka kecepatan aliran airnya adalah:

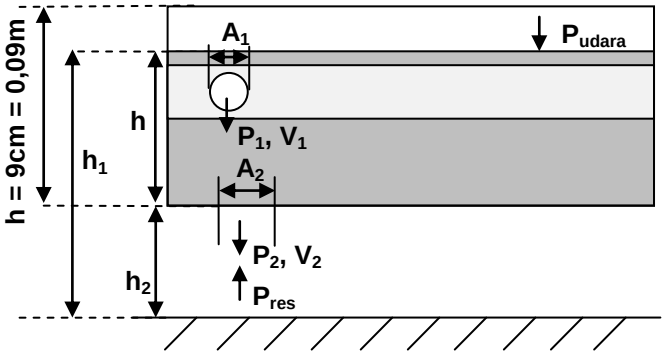
$$113{,}14cm^3s^{-1} = v * 3{,}14cm^2 \quad (3.45.)$$

$$v = \frac{113{,}14cm^3s^{-1}}{3{,}14cm^2} = 36{,}03cms^{-1} = 0{,}36ms^{-1} \quad (3.46.)$$

Dari hasil perhitungan didapatkan bahwa kecepatan aliran air yang melewati lubang-lubang pada pipa tersebut adalah 0,36ms^{-1}. Sehingga konstanta Reynolds dapat dihitung:

$$\mathrm{Re} = \frac{(0{,}36 ms^{-1})(8*10^{-2} m)}{1{,}004*10^{-6} m^2 s^{-1}} = 27888{,}45 \qquad (3.47.)$$

Nilai konstanta Reynolds yang didapatkan sebesar 27888,45 (lebih besar daripada 4000), maka aliran air dari pipa berlubang masih mengalami *turbulence*. Kemudian tekanan P_1 dapat dicari dengan cara sebagai berikut:



h_1 = 221,5cm = 2,215m
h_2 = 215,5cm = 2,155m
h = h_1 – h_2 = 0,06m
A_1 = 16*(3,14*(0,5/2)) = 3,14cm^2
= 3,14*10^{-4} m^2
A_2 = 3,14*(0,9/2)^2 = 0,636cm^2
= 6,36*10^{-5} m^2
$A_1*V_1 = A_2*V_2$
$V_2 = (A_1*V_1)/A_2$
V_2 = ((3,14*10^{-4} m^2)*0,36ms^{-1})/(6,36*10^{-5} m^2)
V_2 = 1,78ms^{-1}
$P_{udara}*V_{udara} = n*R*T$
$P_{udara} = (1\ mol*0{,}082*(273+25)K)/V_1$
P_{udara} = (24,436)/(1,06) = 23,05atm
P_{udara} = 2335541,25Pa
$P_2 = (P_{udara} – P_{res}) + \rho gh$

Gambar 3.24. Tekanan aliran air melewati *solenoid valve* saat keran dibuka setengah (gambar tidak untuk diskala)

Konstanta Reynolds dari air yang keluar (dengan kecepatan v_2) melalui lubang *solenoid valve* adalah:

$$\mathrm{Re} = \frac{(1{,}78 ms^{-1})(9*10^{-3} m)}{1{,}004*10^{-6} m^2 s^{-1}} = 15936{,}25 \qquad (3.48.)$$

Koefisien friksi Darcy harus dicari terlebih dahulu untuk menghitung tekanan P_1 di dalam *box* akrilik:

$$f_D = \frac{0{,}25}{\left(\log\left(\frac{\frac{0{,}005*10^{-3} m}{80*10^{-3} m}}{3{,}7} + \frac{5{,}74}{(27888{,}45)^{0{,}9}}\right)\right)^2} \qquad (3.49.)$$

$$f_D = \frac{0{,}25}{\left(\log\left(1{,}69*10^{-5} + 5{,}73*10^{-4}\right)\right)^2} \qquad (3.50.)$$

$$f_D = \frac{0,25}{(-3,229)^2} = 0,024 \quad (3.51.)$$

Dari hasil perhitungan didapatkan bahwa nilai koefisien friksi Darcy adalah 0,024. Nilai koefisien ini selanjutnya digunakan untuk mencari *head loss*:

$$h_{LOSS} = \frac{0,024 * 2,48m * (0,36ms^{-1})^2}{2 * (8 * 10^{-2} m) * 10ms^{-2}} = 0,005m \quad (3.52.)$$

Geodetic head hanya berkurang sedikit bila dibandingkan pada saat keran tertutup. Karena permukaan air di dalam *box* berkurang 0,5cm, maka nilai h_g menjadi 247,5cm. Kemudian *head total* dapat dicari:

$$h_{TOTAL} = 0,005m + 2,475m = 2,48m \quad (3.53.)$$

Karena *head total* sudah diketahui dan nilai *specific gravity* dari air yang diketahui adalah 1, maka tekanan P_1 adalah:

$$P = 0,098 * 2,48 * 1 = 0,243bar = 24,3kPa \quad (3.54.)$$

Karena seluruh variabel di dalam turunan persamaan Bernoulli (pada persamaan 3.37) sudah diketahui nilainya, maka tekanan P_2 dapat dihitung. Akan tetapi tekanan resistansi udara harus dicari terlebih dahulu sebelum menghitung tekanan P_2:

$$P_{res} = 2335541,25Pa - 24300Pa - \tfrac{1}{2}(1000kgm^{-3})(V_1^2 - V_2^2) \quad (3.55.)$$

$$P_{res} = 2311241,25Pa - 500kgm^{-3}(0,36m^2s^{-2} - 1,78m^2s^{-2}) \quad (3.56.)$$

$$P_{res} = 2311241,25Pa - (-1519,5Pa) = 2312760,75Pa \quad (3.57.)$$

Sehingga tekanan yang dialami oleh *solenoid valve* saat membuka adalah:

$$P_2 = 2335541,25Pa - 2312760,75Pa + 600Pa \quad (3.58.)$$

$$P_2 = 22780,5Pa + 600Pa = 23380,5Pa = 0,23bar \quad (3.59.)$$

Tekanan terhadap *solenoid valve* saat keran dibuka bertambah 0,02bar sebagai akibat pengurangan kecepatan awal. Karena kecepatan awal juga

berkurang, maka konstanta Reynolds juga berkurang. Dari nilai konstanta Reynolds sebesar 24900,39 saat keran tertutup, menjadi 15936,25 pada saat keran dibuka. Sehingga efek *turbulence* dapat dikurangi dengan mengurangi kecepatan aliran air terhadap *solenoid valve*.

3.3.4.2. Pengaruh Kecepatan Awal Terhadap *Timing* Tampilan Karakter

Pada bab sebelumnya sudah dibahas mengenai pengaruh kecepatan awal terhadap waktu tempuh air jatuh ke bak. Jika ada kecepatan awal, maka waktu tempuh air jatuh ke bak dapat dicari dengan menggunakan persamaan kinematika yang sudah dibahas pada bab 2.1.3.

$$h = v_o t + \frac{1}{2} g t^2 \tag{3.60.}$$

Kecepatan awal v_o sudah diketahui melalui perhitungan pada bab 3.3.4.1 yaitu $v_0 = 2{,}76 ms^{-1}$ pada saat keran tertutup dan $v_0 = 1{,}78 ms^{-1}$ saat keran dibuka. Besarnya nilai v_0 untuk mencari waktu tampil karakter dengan melakukan substitusi nilai kecepatan awal ke persamaan 3.60. Berikut adalah perhitungan waktu tampil karakter untuk masing-masing nilai kecepatan awal:

$$2 = (2{,}76)t + \frac{1}{2}(10)t^2 \tag{3.61.}$$

$$5t^2 + 2{,}76t - 2 = 0 \tag{3.62.}$$

$$t^2 + 0{,}552t - 0.4 = 0 \tag{3.63.}$$

Dari persamaan aljabar ini variabel waktu *t* difaktorkan:

$$t_{1,2} = \frac{-0{,}552 \pm \sqrt{0{,}552 - 4(1)(-0{,}4)}}{2(1)} \tag{3.64.}$$

$$t_{1,2} = \frac{-0{,}552 \pm 1{,}467}{2} \tag{3.65.}$$

$$t_1 = 0{,}457 s; t_2 = -1{,}009 s \tag{3.66.}$$

Dari hasil perhitungan didapatkan nilai t_1 = 0,457s dan t_2 = –1,009s pada kecepatan awal = 2,76ms^{-1}. Pada kecepatan awal = 1,78ms^{-1} perhitungan waktu tampilnya adalah sebagai berikut:

$$2 = (1{,}78)t + \frac{1}{2}(10)t^2 \tag{3.67.}$$

$$t^2 + 0{,}356t - 0{,}4 = 0 \tag{3.68.}$$

Dari persamaan aljabar ini variabel waktu *t* difaktorkan:

$$t_{1,2} = \frac{-0{,}356 \pm \sqrt{0{,}356 - 4(1)(-0{,}4)}}{2(1)} \tag{3.69.}$$

$$t_{1,2} = \frac{-0{,}356 \pm 1{,}398}{2} \tag{3.70.}$$

$$t_1 = 0{,}521s; t_2 = -0{,}877s \tag{3.71.}$$

Pada nilai kecepatan awal = 1,78ms^{-1} didapatkan bahwa nilai t_1 = 0,521s dan t_2 = –0,877s. Untuk perhitungan *timing* diambil nilai t yang positif. Sehingga waktu tampil karakter yang didapatkan pada saat v_0 = 1,78ms^{-1} lebih lama bila dibandingkan dengan v_0 = 2,76ms^{-1}.

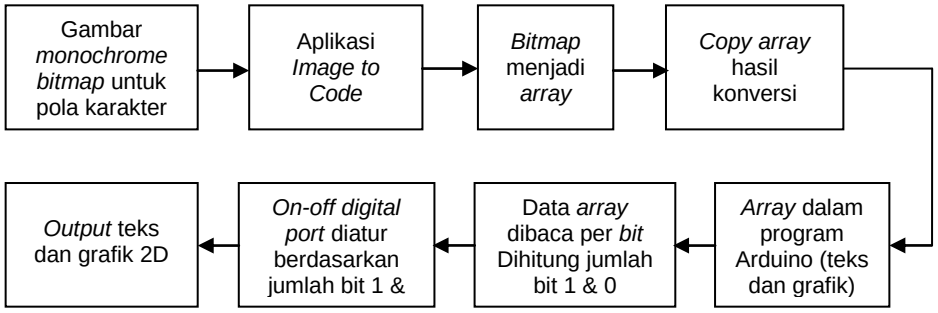
Untuk keperluan pengujian digunakan *timing* sebesar 45ms dengan karakter yang memiliki dimensi 8*8 *pixel*. Sehingga secara perhitungan pada saat *valve* membuka selama 45ms, waktu tampilnya adalah 360ms. Untuk waktu tampil pada saat v_0 = 1,78ms^{-1}, terdapat sisa waktu sebesar 161ms (dengan mengurangi 521ms dengan 360ms). Sedangkan pada saat v_0 = 2,76ms^{-1}, terdapat sisa waktu sebesar 97ms (dengan mengurangi 457ms dengan 360ms).

3.4. Desain Algoritma dan Program

Desain algoritma dan program yang akan digunakan di dalam *waterfall sign display* ini terdiri dari perencanaan *flowchart* program, penggunaan *software Image to Code* untuk mengkonversi gambar *bitmap* ke *array* dan perencanaan pembuatan program Arduino.

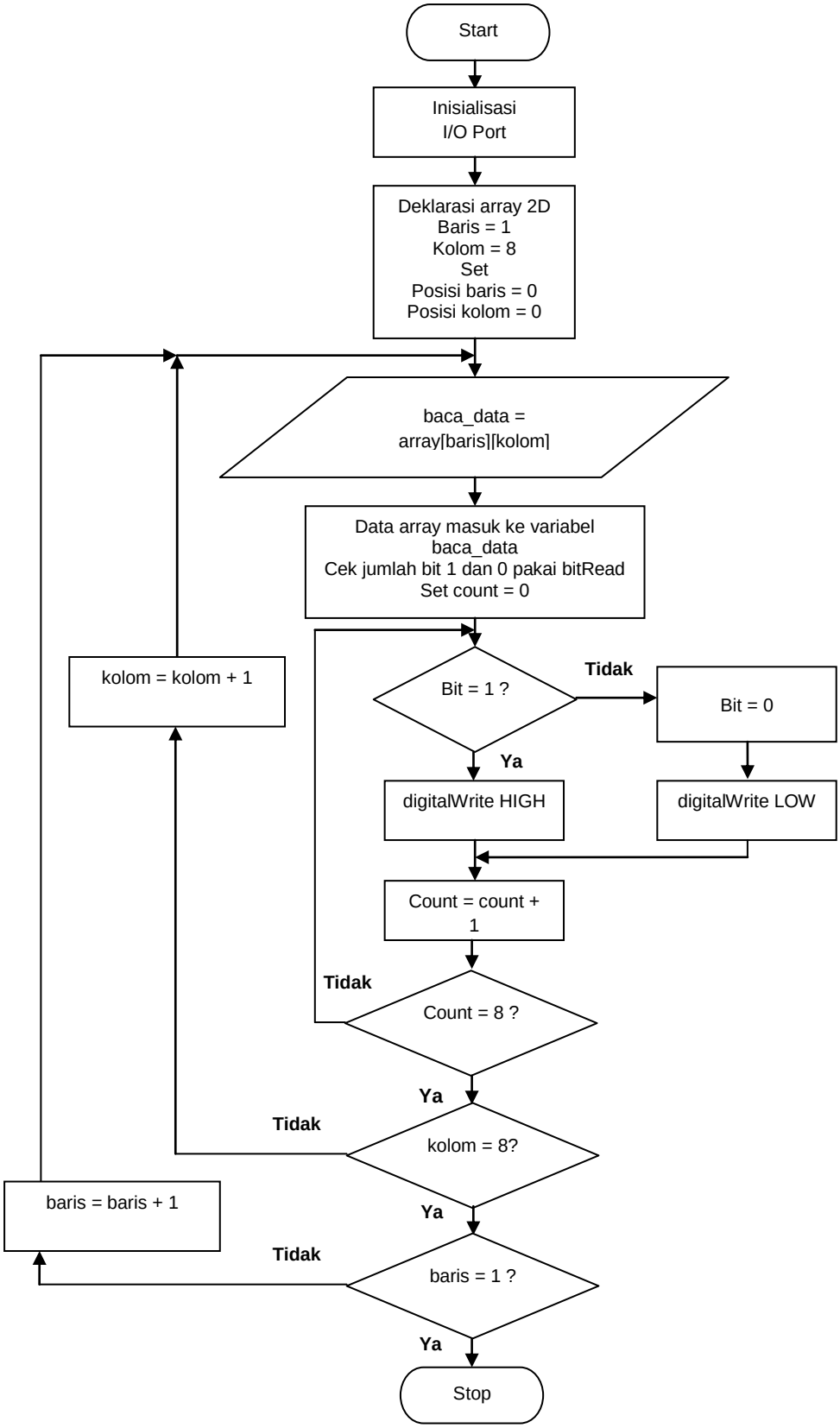
3.4.1. Desain Program Secara Umum

Dalam membuat desain program dibuat blok diagram terlebih dahulu untuk memudahkan pandangan secara garis besar terhadap sisi program. Blok diagram program digunakan untuk menggambarkan program yang akan dibuat secara garis besarnya. Pada alur pertama gambar *bitmap* dipersiapkan untuk membuat pola karakter, selanjutnya dikonversi menjadi *array* dengan aplikasi *Image to Code*. Hasil konversi akan di-*copy* ke dalam program dan selanjutnya akan diolah dengan mengambil data dari *array* (setiap data yang diambil berjumlah satu *byte*) dan menghitung jumlah *bit* 1 dan 0 yang terdapat pada data tersebut untuk mengetahui dalam satu *frame* berapa *solenoid valve* yang diatur *on-off* untuk membentuk pola karakter yang akan ditampilkan.



Gambar 3.25. Blok diagram perancangan program

Cara kerja secara mendetail dapat dibuat sebuah *flowchart* secara umum untuk menegaskan komponen-komponen perancangan program yang berada pada blok diagram di atas.

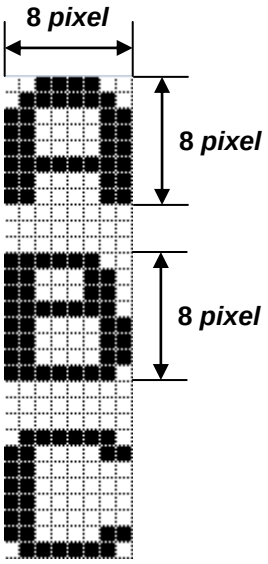


Gambar 3.26. *Flowchart* dasar perancangan program

3.4.2. Penggunaan *Software Image to Code*

Gambar *monochrome bitmap* yang akan dikonversi berupa huruf, angka dan simbol. Terdapat 4 gambar dan masing-masing gambar tersebut memiliki resolusi 8x120 *pixel* kecuali pada gambar terakhir. Di dalam masing-masing gambar tersebut tiap karakternya memiliki resolusi sebesar 8x8 *pixel.*

Gambar 3.27. Gambar *monochrome bitmap* sebagai referensi pola karakter air



Gambar 3.28. Detail gambar *bitmap* per karakter

Terdapat jeda sebanyak 3 *pixel* antar karakter, hal ini bertujuan agar pada saat konversi gambar menjadi *array* tidak membingungkan karena ada jeda.

Konversi gambar dilakukan menurut panduan yang ada pada bab 2.6 mengenai penggunaan aplikasi *Image to Code*. Setelah gambar-gambar di atas ini selesai dikonversi maka hasil konversi tersebut dimasukkan ke dalam *array* program yang ditampilkan pada bab 3.4.3.1 mengenai deklarasi variabel global.

3.4.3. Perancangan Program Arduino

Pemrograman Arduino dibuat dalam *program editor* Arduino versi 1.0.2. Adapun algoritma yang digunakan untuk pemrograman telah dibahas melalui blok diagram dan *flowchart* pada bab 3.4.1. Di dalam program ini ada berbagai macam fungsi yang akan mengolah data dari *array* yang akan dikeluarkan melalui *output* mikrokontroler Arduino.

3.4.3.1. Deklarasi Variabel-variabel Global dan Inisialisasi Arduino

Variabel-variabel yang dideklarasikan di luar dua fungsi *setup* dan *loop* maupun fungsi-fungsi tambahan lainnya merupakan variabel global, yang berarti variabel-variabel tersebut dapat digunakan oleh berbagai fungsi lainnya. Karena jika variabel diletakkan dalam satu fungsi tertentu maka variabel tersebut hanya dapat digunakan di dalam satu fungsi itu saja, pada fungsi lainnya variabel tersebut tidak dikenali.

```
//deklarasi variabel global
int Arduinopin=0;
byte x=0;
byte y=0;
byte z=0; //untuk counter tabel_huruf
byte a=0;
byte b=0;
byte c=0; //untuk counter tabel_angka
byte d=0;
byte e=0;
byte f=0; //untuk counter tabel_simbol

byte tabel_huruf[26][8]={                  //barisxkolom
  {0x3C,0x3C,0x00,0x3C,0x3C,0x3C,0x81,0xC3}, //Huruf A
  {0x01,0x3C,0x3C,0x3C,0x01,0x39,0x39,0x03}, //Huruf B
  {0x81,0x3C,0x3F,0x3F,0x3F,0x3F,0x3C,0x81}, //Huruf C
  {0x03,0x39,0x3C,0x3C,0x3C,0x3C,0x39,0x03}, //Huruf D
  {0x00,0x3F,0x3F,0x3F,0x01,0x3F,0x3F,0x00}, //Huruf E
  {0x3F,0x3F,0x3F,0x3F,0x01,0x3F,0x3F,0x00}, //Huruf F
  {0x81,0x3C,0x3C,0x30,0x3F,0x3F,0x3C,0x81}, //Huruf G
  {0x3C,0x3C,0x3C,0x00,0x00,0x3C,0x3C,0x3C}, //Huruf H
  {0x81,0xE7,0xE7,0xE7,0xE7,0xE7,0xE7,0x81}, //Huruf I
  {0x81,0x38,0xFC,0xFC,0xFC,0xFC,0xFC,0x00}, //Huruf J
  {0x3C,0x39,0x33,0x07,0x0F,0x27,0x33,0x39}, //Huruf K
  {0x00,0x00,0x3F,0x3F,0x3F,0x3F,0x3F,0x3F}, //Huruf L
  {0x3C,0x3C,0x3C,0x3C,0x24,0x24,0x18,0x3C}, //Huruf M
  {0x3C,0x38,0x30,0x34,0x2C,0x0C,0x1C,0x3C}, //Huruf N
  {0x81,0x3C,0x3C,0x3C,0x3C,0x3C,0x3C,0x81}, //Huruf O
  {0x3F,0x3F,0x3F,0x3F,0x01,0x3C,0x3C,0x01}, //Huruf P
```

```
  {0x82,0x39,0x34,0x3C,0x3C,0x3C,0x3C,0x81}, //Huruf Q
  {0x3C,0x3C,0x39,0x33,0x01,0x3C,0x3C,0x01}, //Huruf R
  {0x81,0x3C,0xFC,0xE3,0x9F,0x3F,0x3C,0x81}, //Huruf S
  {0xE7,0xE7,0xE7,0xE7,0xE7,0xE7,0x00,0x00}, //Huruf T
  {0x81,0x00,0x3C,0x3C,0x3C,0x3C,0x3C,0x3C}, //Huruf U
  {0xE7,0xC3,0x99,0x99,0x3C,0x3C,0x3C,0x3C}, //Huruf V
  {0x99,0x00,0x24,0x24,0x24,0x3C,0x3C,0x3C}, //Huruf W
  {0x3C,0x3C,0x99,0xE7,0xE7,0x99,0x3C,0x3C}, //Huruf X
  {0xE7,0xE7,0xE7,0xC3,0x99,0x3C,0x3C,0x3C}, //Huruf Y
  {0x00,0x00,0x9F,0xCF,0xE7,0xF3,0xF9,0x00}  //Huruf Z
  };

byte tabel_angka[10][8]={                 //barisxkolom
  {0x81,0x3C,0x3C,0x3C,0x3C,0x3C,0x3C,0x81}, //Angka 0
  {0x81,0xE7,0xE7,0xE7,0xE7,0x87,0xC7,0xE7}, //Angka 1
  {0x00,0x00,0x9F,0xE7,0xF9,0xFC,0x3C,0x81}, //Angka 2
  {0x81,0x3C,0xFC,0xFC,0xE1,0xFC,0x3C,0x81}, //Angka 3
  {0xFC,0xFC,0xFC,0xFC,0x00,0x3C,0x9C,0xC0}, //Angka 4
  {0x81,0x3C,0xFC,0xFC,0x01,0x3F,0x3F,0x00}, //Angka 5
  {0x81,0x3C,0x3C,0x3C,0x01,0x3F,0x3C,0x81}, //Angka 6
  {0xE7,0xE7,0xE7,0xF3,0xF9,0xFC,0x00,0x00}, //Angka 7
  {0x81,0x3C,0x3C,0x3C,0x81,0x3C,0x3C,0x81}, //Angka 8
  {0x81,0x3C,0xFC,0xFC,0x80,0x3C,0x3C,0x81}  //Angka 9
};

byte tabel_simbol[12][8]={                //barisxkolom
  {0xE7,0xC3,0x81,0xE7,0xE7,0xE7,0xE7,0xE7}, //Panah Bawah
  {0xE7,0xE7,0xE7,0xE7,0xE7,0x81,0xC3,0xE7}, //Panah Atas
  {0xDF,0x9F,0x00,0x00,0x9F,0xDF,0xFF,0xFF}, //Panah Kiri
  {0xFB,0xF9,0x00,0x00,0xF9,0xFB,0xFF,0xFF}, //Panah Kanan
  {0xC3,0xBD,0x7E,0xFF,0xFF,0xFF,0xFF,0xDB}, //Smiley Senyum
  {0x7E,0xBD,0xC3,0xFF,0xFF,0xFF,0xFF,0xDB}, //Smiley Sedih
  {0xE7,0xC3,0x81,0x00,0x00,0x00,0x00,0x99}, //Heart
  {0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00}, //Blok penuh
  {0x77,0xBB,0xDD,0xEE,0x77,0xBB,0xDD,0xEE}, //Grafik berulang 1
  {0x7E,0xBD,0xDB,0xE7,0xE7,0xDB,0xBD,0x7E}, //Grafik berulang 2
  {0xE7,0xC3,0x81,0x00,0x00,0x81,0xC3,0xE7}, //Grafik berulang 3
  {0xFF,0x7E,0x3C,0x99,0xC3,0xE7,0xFF,0xFF}, //Grafik berulang 4
};
```

Dalam deklarasi variabel global di atas ditunjukkan ada deklarasi 3 *array* dua dimensi yaitu tabel_huruf, tabel_angka dan tabel_simbol yang berarti 3 *array* tersebut dapat dibaca isi datanya oleh fungsi-fungsi lainnya. Inisialisasi dari program Arduino dilakukan di dalam fungsi *setup*.

```
void setup()
{
  int Arduinopin=2;  //pin digital 0 dan 1 reserved utk TX/RX
  while(Arduinopin<10)
  {
    digitalWrite(Arduinopin,HIGH); //mencegah pin Arduino untuk menyala
```

```
    pinMode(Arduinopin,OUTPUT);  //(kalau output butuh active low) pada saat booting
    Arduinopin++;
  }
}
```

3.4.3.2.*Array Processing*

Data-data yang tersedia pada *array* di program merupakan hasil konversi dari gambar *monochrome bitmap* akan diproses untuk ditampilkan melalui *output* Arduino. Cara untuk mengambil data-data yang berada di dalam *array* tabel_huruf adalah sebagai berikut:

```
byte input=tabel_huruf[z][x];
```

Cara mengambil data dari *array* adalah dengan mengakses alamat data di *array* itu berada dan kemudian memberikan hasilnya ke dalam sebuah variabel.

```
for(z=0;z<26;z++)   //loop untuk lebih dari satu karakter
  {                 //jumlah loop menentukan jumlah karakter
    for(x=0;x<8;x++) //loop untuk satu karakter saja
    {               //jumlah loop menentukan jumlah baris
      byte input=tabel_huruf[z][x];
    }
  }
```

Fungsi di atas menunjukkan bahwa pada saat program dimulai maka yang dilakukan adalah mengambil data pada *array* tabel_huruf pada alamat tabel_huruf[0][0] yang menunjukkan data akan diambil pada alamat tersebut. Kemudian dalam *looping* for(x=0;x<8;x++) akan diselesaikan terlebih dahulu (di-*increment* sampai 8) sehingga yang terjadi adalah pengaksesan alamat tabel_huruf[0][0], tabel_huruf[0][1], hingga tabel_huruf[0][7].

Pada saat mengakses alamat *array* selanjutnya dilakukan pembacaan isi dari alamat tabel_huruf[0][0], kemudian memindahkannya ke dalam variabel *input* dan menghitung jumlah *bit* 1 dan 0 dari data yang ada di variabel *input* ini. Karena proses ini berada dalam *looping* for(y=0;y<8;y++), maka penghitungan tiap *bit* dari variabel *input* tidak akan diselesaikan sampai 8 kali *looping*. Proses ini diulang terus menerus sampai 8 kali hingga alamat tabel_huruf[0][7]. Sehingga variabel *x* menentukan *frame* dari karakter dan *y* menentukan per *bit* dari *frame* karakter, sedangkan *z* adalah satu karakter itu sendiri.

```
for(z=0;z<26;z++)   //loop untuk lebih dari satu karakter
 {                  //jumlah loop menentukan jumlah karakter
   for(x=0;x<8;x++) //loop untuk satu karakter saja
   {                //jumlah loop menentukan jumlah baris
     int Arduinopin=9;
     byte input=tabel_huruf[z][x];
     for(y=0;y<8;y++)
     {
       bitRead(input,y);

       if(input&(1<<y)) //jika hasil AND = 1, set output ke HIGH
       {
         digitalWrite(Arduinopin,HIGH);
         Arduinopin--;
       }
       else //jika hasil AND != 1, set output ke LOW
       {
         digitalWrite(Arduinopin,LOW);
         Arduinopin--;
       }
       delay(45); //delay disini untuk mengatur jeda per baris dari satu karakter
     }
     //masukkan fungsi off untuk mematikan seluruh valve
     delay(250);
 }
```

Fungsi bitRead digunakan untuk membaca sebuah isi variabel tiap *bit*-nya, ketika bitRead membaca isi variabel tersebut maka bitRead akan mengembalikan nilai *bit* yang dibaca. Urutan pembacaan *bit* dimulai dari LSB, yaitu *bit* yang berada pada posisi paling kanan. Jadi bitRead berfungsi untuk menghitung jumlah *bit* 1 dan 0 yang terdapat dalam variabel *input* dengan cara saat membaca variabel tersebut jika mendapatkan nilai *bit* 1 maka akan menonaktifkan *port* dengan memberikan *logic* 1 dan sebaliknya jika nilai mendapatkan *bit* 0 maka akan mengaktifkan *port* dengan memberikan *logic* 0. Pada deklarasi variabel Arduinopin diberikan angka 9. Artinya adalah ketika program melakukan pembacaan *bit* untuk pertama kali, *output* yang menyala terlebih dahulu adalah *digital pin* 9 sampai kepada *digital pin* 2 secara urut.

Timing per *frame* karakter selama 45ms dimasukkan setelah bitRead memberikan nilai *logic* 1 atau 0 untuk menentukan kondisi *output* Arduino. Karena resolusi per karakter yang ditampilkan adalah 8x8 *pixel*, maka karakter akan ditampilkan selama 8x45ms = 360ms. Sedangkan delay(250) berarti antara satu karakter dengan karakter lainnya ada jeda 250ms.