Jurnal Nasional Teknik Elektro, Vol. 7, No. 1, Maret 2018
p-ISSN: 2302-2949, e-ISSN: 2407-7267

Pengontrolan Motor *Stepper* Menggunakan *Driver* DRV 8825 Berbasis *Signal Square Wave* dari Timer Mikrokontroler AVR

Arief Wisnu Wardhana*, Daru Tri Nugroho
Jurusan Teknik Elektro, Universitas Jenderal Soedirman
Corresponding authors e-mail : arief.wardhana@unsoed.ac.id

Abstrak—Pada penelitian ini dirancang suatu rangkaian elektronik untuk mengendalikan motor stepper. Suatu IC kontroler digunakan sebagai piranti kontrolnya. *Microcontroller Unit (MCU)* digunakan untuk menghasilkan sinyal input. Rangkaian ini dicobakan pada dua jenis motor yaitu motor stepper *Permanent Magnet* dan motor stepper *Hybrid*. Berbagai mode *stepping* dilakukan pada penelitian ini, yaitu *full-stepping, half-stepping* dan *microstepping*. Beberapa profil pergerakan motor *stepper* yang biasa digunakan pada berbagai aplikasi juga dicobakan. Parameter pergerakan motor diamati, seperti besarnya kecepatan motor, seberapa halus suara motor yang dihasilkan, serta profil percepatannya. Hasil yang diperoleh adalah bahwa dengan menggunakan level *microstepping* yang lebih tinggi menyebabkan suara motor *stepper* menjadi lebih tidak terdengar. Hasil selanjutnya didapat bahwa dengan mengaplikasikan suatu profil percepatan di dalam program mikrokontroler, maka bisa ditargetkan suatu kecepatan motor *stepper* yang diperkirakan melebihi kecepatan *start-on* yang di support oleh motor *stepper* tersebut.

Kata Kunci : stepper motor, microcontroller unit, integrated circuit, microstepping, Permanent Magnet, Hybrid

Abstract—In this research, an electronic circuit system was designed for controlling *stepper motors*. A stepper motor driver IC was used to implement those control mechanism. A *Microcontroller Unit (MCU)* is used to generate the square-wave needed for input to motor driver. The circuit was used on two types of stepper motors, namely permanent magnet stepper motor and hybrid stepper motor. Several modes of controls are conducted here, which are full-stepping, half-stepping, and microstepping. Some useful motor movements which are often used in many applications were tried here. Observations were made on the speed of the motor for various input waveform frequencies, on the stepper motor sound for a range of microstepping level, and on the acceleration profile. Some significant results were obtained in that by using higher microstepping levels, a smoother stepper motor movement can be achieved. Another important result is that by applying an acceleration profile in the microcontroller program, we may target a speed which is well above the rated start-on speed of the stepper motor.

Keywords : stepper motor, microcontroller unit, stepper motor driver integrated circuit, full-stepping, half-stepping, microstepping

1. Pendahuluan

Pada bidang keteknikan, kita sering memanfaatkan berbagai tipe pergerakan dalam arah rotasional, bisa berbentuk satu putaran penuh, atau setengah putaran, seperempat dan seterusnya. Banyak instrumen, peralatan dan perangkat elektronik disekitar kita yang juga beroperasi dengan basis pergerakan rotasional ini. Beberapa aplikasi yang memanfaatkan tipe pergerakan ini diantaranya adalah *automatic teller machines, money handling machines, video security cameras, printers, scanners, office automation machines, gaming machines, factory automation* dan *robotics.*

Untuk membantu menggerakkan berbagai peralatan elektronik tersebut, tentunya dibutuhkan suatu perangkat alat atau instrumen bantu. Suatu alat yang bernama motor *stepper* bisa mengakomodasi kebutuhan ini.

Penelitian ini secara terperinci membahas gerakan rotasi pada piranti elektronik motor *stepper*. Beberapa tipe gerakan rotasi yang dianggap bisa dimanfaatkan oleh berbagai aplikasi dibahas pada penelitian ini.

Ada dua tujuan yang ingin didapatkan pada penelitian ini. Pertama adalah untuk membandingkan keluaran pergerakan motor

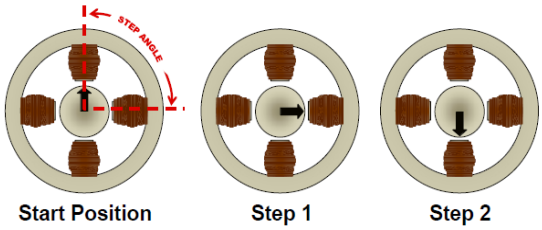
Received date 2018-02-08, Revised date 2018-03-02, Accepted date 2018-03-02
https://doi.org/10.25077/jnte.v7n1.530.2018

stepper yang dihasilkan dari berbagai masukan pola *stepping*. Kedua, untuk merancang suatu profil pola kecepatan untuk memastikan bahwa motor *stepper* tidak akan pernah macet ketika sedang dipercepat atau diperlambat.

2. Teori Motor *Stepper*

Tidak seperti motor DC yang berputar secara kontinu setelah tegangan pengoperasiannya terlampaui, motor *stepper* berputar dengan langkah-langkah yang diskrit, dengan setiap langkahnya yang harus dipicu oleh rangkaian aplikasi nya. Dikarenakan setiap langkah pemutaran motor *stepper* berupa suatu sudut yang konstan, maka motor *stepper* adalah suatu piranti yang sangat presisi yang setiap gerakannya bisa diulang dengan mudah. Oleh karena itu, motor *stepper* sangat tepat untuk digunakan pada beberapa aplikasi seperti *printer*, *plotter*, *disk drive*.

Motor *stepper* bergerak dalam langkah-langkah yang sangat jelas terlihat selama pergerakan rotasinya. Setiap langkahnya didefinisikan dengan sebuah istilah yang disebut dengan *step angle*. Apabila diperhatikan pada contoh Gambar 1 di bawah ini, terlihat bahwa terdapat 4 langkah yang jelas agar rotor bisa menghasilkan satu rotasi lengkap sebesar 360° (360 derajat). Gambar tersebut mendefinisikan sebuah *step angle* sebesar 90° (90 derajat).



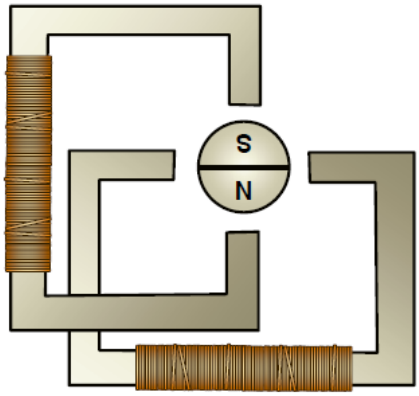
Gambar 1. *Stepper Motor* [5].

Motor *stepper* merupakan suatu motor *digital* karena motor ini bergerak secara diskrit. Karakteristik ini yang menjadikan motor *stepper* sangat sesuai untuk sistem antarmuka dengan mikrokontroler.

2.1 Stepper Motor Permanent Magnent

Tipe dasar motor *stepper* terdiri atas dua jenis, yaitu tipe motor *stepper Permanent Magnet (PM)* dan tipe motor *stepper Variable Reluctance (VR)*. Selain kedua tipe tersebut, terdapat juga tipe motor stepper *Hybrid* yang menggabungkan fitur yang terdapat pada *Permanent Magnet* dan *Variable Reluctance*.

Motor s*tepper PM* pada dasarnya terdiri dari sebuah *rotor* yang terbuat dari magnet permanen dan dua pasangan kumparan stator (yang disebut fasa). Pada Gambar 2 ditunjukkan motor *stepper PM* yang memiliki satu *rotor* dan sepasang kumparan *stator*.

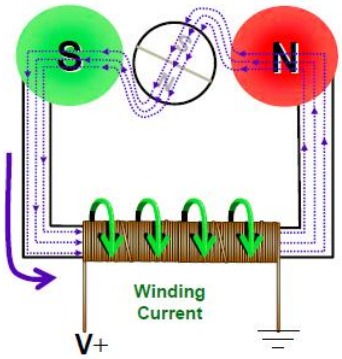


Gambar 2. *Rotor* dan sepasang kumparan *stator* pada sebuah stepper motor PM [5].

Jika suatu tegangan diaplikasikan ke salah satu kumparan, seperti terlihat pada Gambar 3, arus listrik akan mengalir melalui kumparan tersebut. Jika digunakan teori kaidah tangan kanan, posisi jari-jari yang melingkupi kumparan menunjukkan arah arus yang mengalir ke kumparan, sedangkan ibu jari menunjukkan arah *magnetic flux*. Hal ini menyebabkan setiap ujung stator menjadi termagnetisasi dengan kutub yang berlawanan. *Magnetic flux* tersebut akan mengalir dari kutub utara (*north pole*), dan kemudian berlanjut melewati *magnetic rotor* menuju ke kutub stator lainnya yaitu kutub selatan (*south pole*). *Magnetic flux* ini mempunyai kecenderungan untuk mengikuti lintasan dengan resistansi yang paling kecil atau kecenderungan untuk mengikuti lintasan dengan *magnetic reluctance* paling kecil.

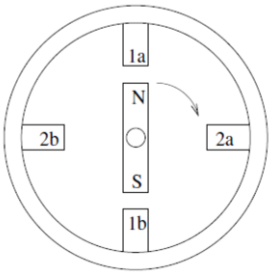
Dengan demikian, *rotor* akan berputar untuk meminimalkan lintasan *flux* (meminimalkan *reluctance*). Jadi, jika suatu kumparan dialiri arus, kumparan tersebut akan menarik *rotor*, sehingga *rotor* akan menyesuaikan perputarannya sesuai dengan kumparan yang menariknya dan akhirnya *rotor* akan berhenti.

https://doi.org/10.25077/jnte.v7n1.530.2018
jnte.ft.unand.ac.id



Gambar 3. Aplikasi tegangan ke sebuah kumparan [5].

Selanjutnya pada Gambar 4, apabila tegangan diberikan secara bergantian pada kumparan 1a, 2a, 1b, dan 2b maka akan menyebabkan *rotor* berputar searah jarum jam dalam empat langkah yang berbeda dengan besarnya setiap langkah adalah 90°, sehingga terjadi satu putaran penuh.

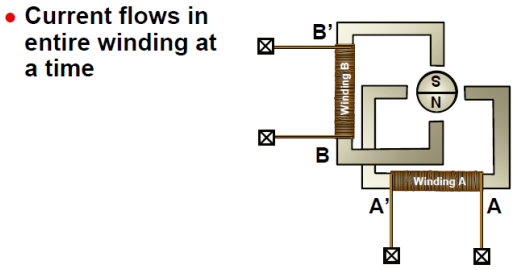


Gambar 4. *Motor stepper permanent magnet* [2].

Secara fisiknya, *permanent magnet stepper motor* dapat berupa jenis *bipolar* atau *unipolar*. Pada penelitian ini digunakan motor *stepper bipolar*, yaitu tipe NEMA, *size* 17, 1.8° *2-phase bipolar stepper motor* 17HD40005-22B [7]. Sebagai pembanding, pada penelitian Ivan Virgala et. al. dilakukan percobaan dengan *motor stepper bipolar* SY28STH32-0674A [4].

Suatu *stepper motor* jenis *bipolar* mempunyai empat kabel yang masing-masing kabelnya dihubungkan dengan kumparan fasa. Hal ini ditunjukkan pada Gambar 5.

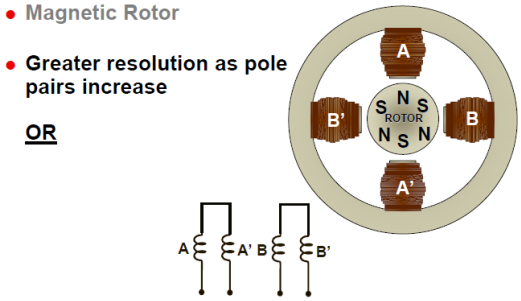
Polaritas dari sebuah fasa ditentukan oleh arah arus yang yang sedang melalui kumparannya. Untuk menjalankan motor *stepper*, dibutuhkan sebuah *H-bridge*, yaitu suatu perangkat *hardware* tambahan yang digunakan untuk menjalankan motor *stepper*. Perangkat tambahan ini juga disebut dengan *stepper motor driver*. Walaupun dibutuhkan perangkat tambahan ini, keuntungan yang diperoleh adalah motor *stepper* akan memiliki *torque* yang maksimum sehingga *static position error* menjadi menjadi lebih kecil.



Gambar 5. Motor *stepper bipolar* [5].

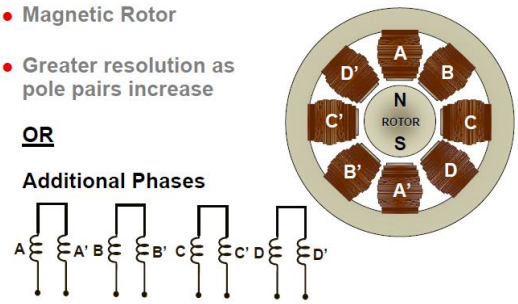
2.2 Memperbaiki Resolusi Rotor

Ada beberapa metode agar resolusi rotasi *rotor* atau memperkecil besarnya *step angle* dari sebuah *rotor permanent magnet* dapat diperbaiki. Metode pertama adalah dengan menaikkan jumlah pasangan *pole* pada bagian *rotor* itu sendiri. Hal ini ditunjukkan pada Gambar 6 berikut:



Gambar 6. Resolusi rotasi *rotor* menjadi lebih baik dengan menambah jumlah pasangan *pole* pada *rotor* [5].

Cara lain untuk memperbaiki resolusi rotasi *rotor* adalah dengan menambahkan banyak *stator* dan fasa. Metode ini ditunjukkan pada Gambar 7, dimana resolusi rotasi *rotor* diperbaiki secara *hardware*.

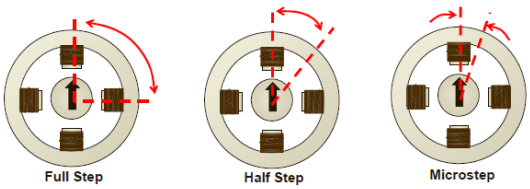


Gambar 7. Resolusi rotasi *rotor* menjadi lebih baik dengan penambahan *stator* dan fasa [5].

https://doi.org/10.25077/jnte.v7n1.530.2018
jnte.ft.unand.ac.id

Metode selanjutnya untuk menaikkan resolusi rotasi rotor adalah dengan cara *software*, yaitu menggunakan berbagai teknik *stepping* [5]. Teknik inilah yang dilakukan pada penelitian ini.

Ada tiga pola *stepping* berbeda yang dapat digunakan untuk mempengaruhi posisi terindeks dari *shaft (rotor)*. Pola-pola tersebut adalah *full-stepping*, *half-stepping* dan *microstepping*, seperti ditunjukkan pada Gambar 8. Pola *stepping* ini akan dijelaskan di sub bab selanjutnya.

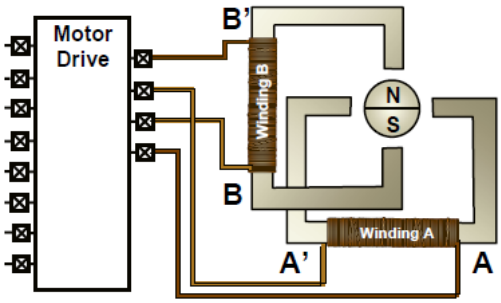


Gambar 8. Pola *stepping* [6].

2.3 Speed Control

Algoritma untuk *Full-Stepping* ditunjukkan oleh diagram pada Gambar 9 berikut ini. Diagram ini diperoleh dari penjelasan pada subbab 3 diatas. Pada Gambar 9 ini ditunjukkan bahwa setiap kumparan dapat diaktifkan secara secara bergantian.

Step	1	2	3	4
Winding A	A	0	A'	0
Winding B	0	B	0	B'

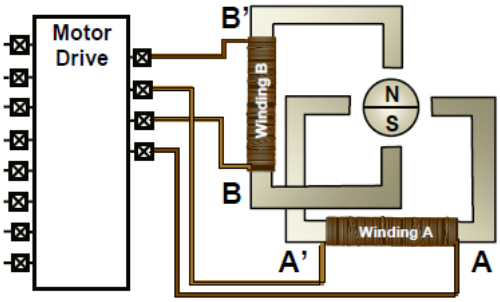


Gambar 9. Algoritma *Full Stepping* [6].

Pada motor *stepper* tipe *Permanent Magnet*, telah disebutkan bahwa mengaktifkan dua fasa secara bersamaan akan menaikkan *torque*, tetapi juga akan menaikkan konsumsi daya. Walaupun demikian, ada satu efek lain yang akhirnya membuat teknik ini menjadi berguna. Jika diperhatikan kembali Gambar 4, dengan mengaktifkan kumparan 1a + 2a secara bersamaan, maka akan membawa rotor pada posisi pertengahan antara 1a dan 2a. Jadi, jika digunakan sekuensi pengaktifan 1a, 1a + 2a, 2a, 2a + 1b, 1b, . . . , cara ini akan membagi dua step *angle* dari motor *stepper* tersebut, sehingga akan melipatgandakan jumlah *step* untuk setiap revolusinya. Teknik ini disebut dengan *half-stepping*.

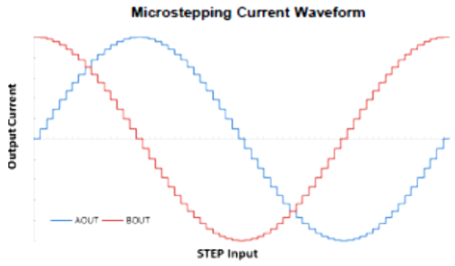
Jadi, dengan penjelasan di atas dan dengan melihat diagram dan skema pada Gambar 10, algoritma untuk *half-stepping* adalah sebagai berikut [6] :

Step	1	2	3	4	5	6	7	8
Winding A	A	A	0	A'	A'	A'	0	A
Winding B	0	B	B	B	0	B'	B'	B'



Gambar 10. Algoritma *Half Stepping* [6]

Selanjutnya, dengan merujuk kembali ke Gambar 10, apabila kedua kumparan tidak diaktifkan secara penuh, tetapi dimulai dengan kumparan A dan kemudian secara perlahan lahan menaikkan besarnya arus pada kumparan B sementara secara bersamaan mengurangi besarnya arus pada kumparan A, maka akan didapatkan *step granularity* yang lebih halus antara kumparan A dan kumparan B. Teknik ini disebut *micro-stepping*. Gambar 11 berikut menunjukkan profil arus input pada kumparan dengan teknik *microstepping* ini.

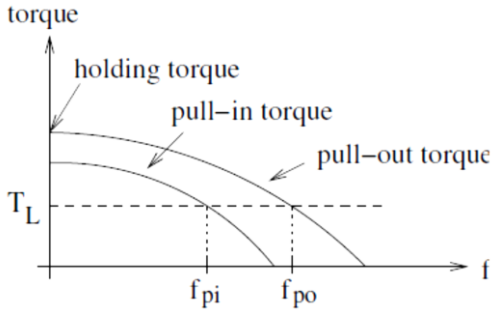


Gambar 11. Profil arus input di kumparan A dan kumparan B untuk *microstepping* [8]

https://doi.org/10.25077/jnte.v7n1.530.2018
jnte.ft.unand.ac.id

Half-stepping dan *microstepping* menyebabkan motor stepper akan berputar dengan cara yang lebih halus sehingga suara putaran *rotor* menjadi lebih tidak terdengar. Jumlah *step* bisa naik secara signifikan, bahkan bisa mencapai 256 kali lipat. Jadi, untuk sebuah motor *stepper* dengan *step angle* θ sebesar 1.8° akan memberikan 256 x 200 = 51200 step untuk setiap satu putaran nya. Akan tetapi, presisi untuk *microstepping* lebih buruk dibandingkan presisi untuk *full-stepping*.

Kecepatan sebuah motor *stepper* dapat dinyatakan dalam *revolutions per minute* (rpm). Tetapi, ketika merujuk pada kecepatan operasionalnya, secara umum satuan kecepatan *stepping frequency* dinyatakan dalam Hz atau pps *(pulse per second)*. Selain dipengaruhi oleh *switching speed* dari *motor stepper driver*, besarnya *stepping frequency* (f_{step}) yang dibutuhkan juga dipengaruhi oleh karakteristik motor *stepper* itu sendiri dan beban yang sedang dijalankan. Berkaitan dengan beban yang sedang dijalankan oleh motor *stepper*, hubungan antara kecepatan dengan *torque* pada sebuah motor stepper dinyatakan dalam bentuk kurva *kecepatan-torque*.



Gambar 12. Kurva kecepatan vs *torque* dari sebuah motor *stepper* [2]

Gambar 12 menunjukkan kurva besarnya *torque* beban maksimum yang tersedia untuk setiap kecepatan. *Torque* beban harus lebih kecil dibandingkan dengan *torque* beban maksimum yang diberikan untuk setiap kecepatan tersebut. Demikian pula sebaliknya, apabila nilai torque beban diketahui, maka diagram pada Gambar 12 dapat digunakan untuk menentukan besarnya kecepatan maksimum yang dapat dicapai. Berdasarkan kurva tersebut, besarnya torque beban maksimum yang diperbolehkan akan berkurang seiring dengan bertambahnya kecepatan, sehingga menghasilkan suatu *limit stepping* frekuensi maksimum yang sesuai untuk sebuah nilai torque pada beban tertentu. *Stepping frequency* dari berbagai motor stepper biasanya berada dalam rentang Hz-kHz.

Kurva pada Gambar 12 terdiri atas dua kurva *torque*. Kurva yang kecil adalah kurva *pull-in torque*, yang berlaku *(valid)* hanya pada saat motor *stepper* dinyalakan. Untuk suatu *torque* beban tertentu T_L, f_{pi} adalah frekuensi maksimum di mana motor bisa dinyalakan tanpa kehilangan *step*. Jika motor stepper dalam keadaan sedang *running*, *torque* akan lebih tinggi. *Torque* ini diberikan pada kurva *pull-out torque*. Pada kondisi ini, motor stepper dapat dioperasikan dengan frekuensi $f_{po} > f_{pi}$. Sebagai konsekuensinya, pada saat terjadi percepatan, motor *stepper* harus di *"ramped"*. Artinya, pengoperasian motor stepper dimulai dengan suatu *stepping rate* yang lambat dan kemudian secara berangsur angsur dinaikkan hingga mencapai *pull-out rate* maksimum [2]. Hal yang sama juga berlaku pada saat motor *stepper* diperlambat, dimana kecepatan motor diturunkan secara berangsur-angsur sebelum berhenti. Jika hal ini tidak dilakukan, motor *stepper* bisa kehilangan langkah ketika dinyalakan dan dapat dieksekusi *step* tambahan ketika motor dihentikan. Sementara *torque* maksimum pada kecepatan nol disebut dengan *holding torque*. Apabila torque ini dilampaui, motor *stepper* tidak dapat menahan beban pada posisi nya.

2.4 Stepper Motor Control

Kumparan-kumparan pada suatu motor *stepper* dapat dikontrol secara langsung (dengan rangkaian driver yang sesuai) yaitu dengan mengatur level arus pada setiap kumparan, seperti dijelaskan pada subbab 2.3 di atas (lihat Gambar 11). Akan tetapi sekarang telah tersedia berbagai *driver* IC *(integrated circuit)* yang bisa meringankan tugas mengatur level arus pada motor *stepper*.

Pada penelitian ini, digunakan IC motor *stepper driver* dengan jenis DRV8825 yang dirancang oleh *Texas Instrument*. Driver tipe ini merupakan jenis driver untuk *bipolar stepper motor*. Beberapa perusahaan motor *stepper driver* yang lainnya juga menawarkan IC yang serupa, misalnya motor *stepper driver* A9880.

Ciri umum dari semua IC *driver* tersebut adalah motor stepper dapat dikontrol hanya dengan menggunakan dua sinyal yang dapat dioperasikan pada IC driver tersebut, yaitu sinyal *STEP* dan sinyal *DIRECTION*. Pin DIRECTION akan mengontrol arah perputaran motor stepper searah jarum jam atau berlawanan arah jarum jam.

Kemudian, apabila suatu gelombang berbentuk pulsa (*square waveform*) diaplikasikan pada pin *STEP*, maka motor *stepper* akan berputar satu langkah dengan arah yang telah ditentukan. Logika kumparan mana yang akan dinyalakan selanjutnya telah diimplementasikan di dalam IC driver tersebut.

Selain kemampuan dasar di atas, beberapa driver juga dilengkapi dengan logika untuk *half-stepping* dan *microstepping*. Beberapa IC juga menawarkan suatu *free-running mode* dengan *user-selectable frequencies*, dimana motor *stepper* akan berputar dengan sendirinya segera setelah mode tersebut diaktifkan. Hal ini sangat berguna pada situasi dimana motor stepper hanya diperlukan untuk sekedar berputar tanpa dipengaruhi oleh posisinya.

Untuk mengontrol kecepatan sebuah motor *stepper* dengan menggunakan sebuah IC motor *stepper driver*, persyaratan yang dibutuhkan adalah *microcontroller* yang digunakan harus menghasilkan suatu sinyal step periode dengan frekuensi tertentu. Sebuah keluaran *squarewave* biasa atau *Pulse Width Modulation (PWM)* sangat cocok digunakan untuk menghasilkan sebuah sinyal seperti ini.

Driver ICs tersebut pada umumnya mempunyai sebuah nilai *step frequency* maksimum yang besarnya sama atau lebih besar dari frekuensi *stepper motor* agar bisa memanfaatkan motor stepper secara maksimal. Seperti telah dijelaskan pada subbab 2.3, jika kecepatan motor harus dinaikkan, laju stepping harus diubah secara berulang-ulang, dimulai dengan *pull-in rate* dan terus meningkat hingga mencapai *pull-out rate*. Setiap *intermediate rate* harus ditahan dalam waktu tertentu sebelum *switching* ke *rate* lebih tinggi berikutnya. Hal yang sama juga berlaku untuk perlambatan, tetapi biasanya motor dapat diperlambat secara lebih cepat dibandingkan pada saat dipercepat.

Beberapa *high-end microcontroller* telah dilengkapi dengan fitur *ramping* yang diberikan dalam bentuk *ramp tables*, namun pada *microcontrollers* sederhana, profil *ramping* tersebut harus dirancang terlebih dahulu.

2.5 DRV8825 *Stepper Motor Control* IC

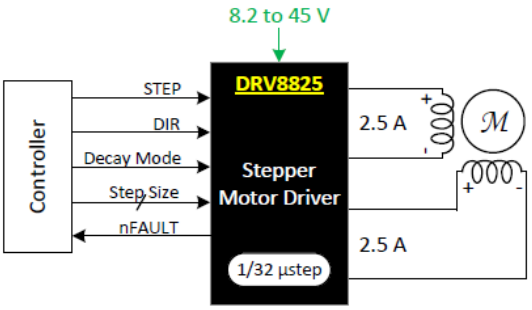
Motor kontroler IC DRV8825 yang digunakan pada penelitian ini menyediakan suatu solusi *integrated motor driver* yang dapat digunakan pada *printer*, *scanner* dan aplikasi peralatan otomasi lainnya. Perangkat ini mempunyai dua *H-bridge driver* dan satu *microstepping indexer* yang diperuntukkan untuk menjalankan suatu *stepper motor bipolar*. Blok *output driver* terdiri dari beberapa *N-channel power MOSFET* yang dikonfigurasikan sebagai *full H-bridges* sebagai *drive* kumparan motor stepper. Driver ini mempunyai kemampuan untuk menggerakkan arus hingga sebesar 2.5 *Ampere* dari tiap-tiap *output* motor tersebut (dengan *heat sinking* yang tepat pada 24 *Volt* dan 25°C).

DRV8825 dapat diberi daya dengan suatu tegangan supply antara 8.2 V sampai 45 V dan mampu menyediakan keluaran arus hingga 2.5 A pada skala penuh. Sementara, fitur *STEP/DIR* sederhana memungkinkan kemudahan antarmuka ke rangkaian *controller* (pada penelitian ini adalah rangkaian mikrokontroller). Fasilitas *internal indexer* pada driver ini mempunyai kemampuan untuk mengeksekusi *high-accuracy microstepping* tanpa mengharuskan mikrokontroler mengontrol level arus.

Pada driver ini terdapat tiga pin *MODE* yang dapat digunakan untuk mengkonfigurasi rangkaian pengontrol ini. Pin-pin *MODE* tersebut dapat digunakan untuk konfigurasi motor stepper mulai dari mode *full-step* sampai mode 32 $\frac{\mu steps}{step}$.

3. Metode Penelitian

Penelitian ini dilakukan dengan merancang suatu sistem rangkaian seperti terlihat pada blok diagram pada Gambar 13 berikut ini:



Gambar 13. Sistem Rangkaian yang dibuat [8]

Pada Gambar 13 terlihat bahwa sistem ini terdiri atas tiga bagian utama. Bagian pertama adalah rangkaian kontroler, pada penelitian ini berupa mikrokontroller yang berfungsi sebagai penyedia gelombang *square wave* (pembangkitan gelombang *squarewave* menggunakan fitur *Timer*

https://doi.org/10.25077/jnte.v7n1.530.2018
jnte.ft.unand.ac.id

MCU). Bagian kedua adalah rangkaian motor *stepper driver*, yaitu DRV8825. Pada bagian ini berbagai pin yang terdapat pada IC tersebut dihubungkan ke rangkaian pengontrol, motor *stepper*, dan suplai tegangan. Bagian ketiga sistem ini adalah motor *stepper*. Ada empat kabel pada motor stepper yang digunakan. Keempat kabel tersebut merupakan kabel dari kumparan didalam motor. Jadi, dapat diketahui bahwa semua koneksi pada rangkaian ini menggunakan kabel. Sebagai suatu perbandingan, terdapat penelitian yang dilakukan oleh Ifrah Jaffri1, Zeeshan Nafees, Shoaib Zaidi, Oliver Faust yang merancang suatu *wireless stepper motor control system* [3].

3.1 Pembangkitan *Waveform*

Langkah pertama dalam konfigurasi DRV8825 memerlukan data tentang kecepatan motor yang diinginkan dan level *microstepping* yang akan digunakan. Apabila aplikasi target (aplikasi yang dikendalikan oleh motor *stepper*) membutuhkan suatu kecepatan konstan, maka suatu *square wave* dengan frekuensi sebesar f_{step} harus diaplikasikan ke pin *STEP* pada DRV8825.

Pembangkitan *waveform square wave* tersebut dapat dilakukan dengan dua cara. Pertama, dengan menghubungkan pin *STEP* pada DRV8825 dengan salah satu pin I/O pada GPIO dari MCU. Kemudian pin tersebut diatur dengan kondisi *HIGH* dan *LOW* dengan periode *HIGH* dan *LOW* yang dapat diatur pada bagian programnya. Cara ini sebenarnya hanya memanfaatkan fitur *hardware* I/O yang terdapat pada MCU tersebut. Cara lain adalah menggunakan pembangkitan *waveform* melalui fitur *Timer* yang tersedia pada *microcontroller* AVR. *Timer* 0 yang terdapat pada Mikrokontroler AVR ini dapat dioperasikan dengan beberapa mode, salah satunya adalah mode *Clear Timer on Compare Match (CTC*).

Untuk bisa menghasilkan suatu *waveform* keluaran (berupa *squarewave waveform*) pada mode *CTC* ini, keluaran *OC0A* (PD6) bisa disetel untuk men-*toggle* level logika pada setiap *compare match* dengan cara mengatur *Compare Output mode bit* pada *mode toggle* (COM0A1:0= 1). Nilai *OC0A* tidak akan terlihat pada *port* pin (pada PD6) sebelum data *direction* untuk pin tersebut diatur ke *mode output*. *Waveform* yang dihasilkan akan mempunyai frekuensi maksimum sebesar $f_{OC0} = f_{clk\_IO}/2$ pada saat *OCR0A* di set ke nol (0x00). Frekuensi *waveform* didefinisikan oleh persamaan (1) berikut ini :

$$f_{OCnx} = \frac{f_{clk\_IO}}{2.N.(1+OCRnx)} \quad (1)$$

Dimana variabel N merepresentasikan faktor prescale (1, 8, 64, 256, atau 1024) [1].

3.2 Perhitungan Besarnya Frekuensi *Waveform*

Apabila kecepatan *startup* dari motor *stepper* terlalu tinggi, motor stepper tidak akan berputar. Maka harus dipastikan bahwa motor *stepper* dapat menunjang kecepatan yang ditargetkan. Metode lainnya adalah dengan mengimplementasikan suatu profil pemrograman percepatan yang akan menggerakkan motor *stepper* hingga kecepatan yang ditargetkan tersebut.

Untuk suatu kecepatan motor *stepper* yang diinginkan sebesar (v), level *microstepping* yang dipilih sebesar (n_m), dan *step angle* tertera dari motor stepper sebesar (θ_{step}), maka diperoleh [8]:

$$f_{step} \text{ (µsteps / second)} = \frac{v\left(\frac{rotations}{minute}\right) x\ 360\left(\frac{^\circ}{rotation}\right) x\ n_m\left(\frac{\mu steps}{step}\right)}{60\left(\frac{seconds}{minute}\right) x\ \theta_{step}\left(\frac{^\circ}{step}\right)} \quad (2)$$

Persamaan (2) diperoleh sebagai berikut: motor *stepper* yang digunakan pada penelitian ini mempunyai *step angle* sebesar 1.8°. Untuk melakukan satu putaran penuh (360°) maka diperlukan sejumlah $\frac{360^\circ}{1.8^\circ}$ = 200*step*. Perhitungan jumlah *µsteps* secara keseluruhan untuk satu kali rotasi didapat dengan mengkalikan 200 step di atas dengan level $\frac{\mu steps}{step}$ yang dipilih. Jadi, untuk satu kali putaran penuh, jumlah *µsteps* tergantung pada level $\frac{\mu steps}{step}$ yang dipilih.

Untuk melakukan rotasi dengan kecepatan v $\frac{rotasi}{menit}$, maka jumlah *µsteps* yang diperlukan adalah jumlah *µsteps* untuk satu kali rotasi dikalikan dengan jumlah $\frac{rotasi}{menit}$ yang diinginkan. Jika diubah dalam satuan rotasi per detiknya, maka nilai tersebut dibagi dengan 60 $\frac{detik}{menit}$.

Level *microstepping* pada IC *driver* DRV8825 bisa diatur dengan tiga pin *MODE* dengan konfigurasi yang tertera pada Tabel 1. *Microstepping* yang lebih tinggi atau level $\frac{\mu steps}{step}$ yang lebih tinggi berarti akan menghasilkan pergerakan motor yang lebih halus sehingga suara motor stepper menjadi lebih tidak terdengar. Akan tetapi, hal ini akan menaikkan *switching losses*

dan membutuhkan suatu *input* f_{step} yang lebih tinggi untuk mendapatkan kecepatan motor *stepper* yang sama (lihat rumus).

Tabel 1. Tiga *Mode* pin untuk memilih level *Microstepping*

MODE2	MODE1	MODE0	STEP MODE
0	**0**	**0**	Full step (2-phase excitation) with 71% current
0	**0**	**1**	$1/2$ step (1-2 phase excitation)
0	**1**	**0**	$1/4$ step (W1-2 phase excitation)
0	**1**	**1**	8 microsteps/step
1	**0**	**0**	16 microsteps/step
1	**0**	**1**	32 microsteps/step
1	**1**	**0**	32 microsteps/step
1	**1**	**1**	32 microsteps/step

3.3 Percobaan *Mode Stepping*

Built-in indexer logic yang terdapat pada DRV8825 memungkinkan serangkaian konfigurasi *stepping* yang berbeda-beda yang dapat dilakukan. Pada driver tersebut terdapat tiga pin *MODE* yaitu *MODE0* hingga *MODE2* yang bisa digunakan untuk konfigurasi *format stepping* seperti yang ditunjukkan pada Tabel 1.

Langkah pertama yang dilakukan adalah dengan mencoba mode *full-stepping* dengan cara mengatur *MODE0* ke *LOW*, *MODE1* ke *LOW*, dan *MODE2* ke *LOW*. Demikian juga seterusnya untuk berbagai *stepping* lainnya dimana nilai-nilai *MODE1, MODE2* dan *MODE3* disesuaikan.

4. Hasil Penelitian

Beberapa jenis percobaan dilakukan pada penelitian ini. Tabel 1 menunjukkan hasil percobaan ketika digunakan level *microstepping* yang lebih tinggi, sehingga menghasilkan suara motor *stepper* menjadi lebih tidak terdengar dan menjadi lebih halus.

Tabel 2. Hasil percobaan berbagai level stepping

Step	f_{clock_IO} (KHz)	OCR0A	$f_{stepping}$ (Hz)	Suara Motor
Full	250	0x66	1213,59	SMOOTHER
Half	250	0x33	2403,85	
¼	2000,00	0xBB	5319,15	
8µ	2000,00	0x66	9708,74	
16µ	2000,00	0x33	19.230,00	
32µ	16.000,00	0xCC	37.558,69	

Pada penelitian ini juga dilakukan percobaan untuk menggerakkan perputaran motor *stepper* dengan dua arah yaitu arah *clockwise* dan arah *anti-clockwise*. Ini dilakukan dengan mengubah nilai pada *port* pin di mikrokontroler yang dihubungkan dengan pin *DIR* pada motor *stepper driver*. Nilai *port* tersebut diubah dari nilai *HIGH* ke *LOW* atau sebaliknya. Pada percobaan ini, diperoleh hasil bahwa ketika nilai *port* pin pada mikrokontroler *HIGH*, motor *stepper* berputar ke satu arah. Sebaliknya, ketika nilai *port* pin *LOW*, maka motor *stepper* berputar ke arah yang sebaliknya.

Percobaan selanjutnya adalah mengerakkan motor *stepper* dengan interval waktu tertentu. Kondisi motor *ON* untuk suatu interval waktu tertentu dan kemudian *OFF* untuk suatu interval waktu tertentu lainnya. Percobaan ini dilakukan untuk mengantisipasi tipe pergerakan motor *stepper* yang tidak kontinu. Pada beberapa perangkat yang menggunakan motor *stepper,* bisa diamati bahwa tipe perputaran motor *stepper* kadang tidak secara kontinu, yaitu terdapat jeda diam diantara dua perputaran berurutan pada motor *stepper* tersebut. Motor stepper ini dihentikan dengan cara men-*disconnect*-kan OC0A *output* pada *Timer 0* mikrokontroler.

Percobaan selanjutnya dilakukan untuk menganalisa percepatan dan perlambatan pada motor *stepper*. Level microstepping yang digunakan adalah $16\frac{\mu steps}{step}$ dan f_{clk_IO} sebesar 2 MHz. Hasil percobaan ini ditunjukkan pada Tabel 3.

Percobaan ini dimulai dengan memberikan nilai OCR0A dengan nilai frekuensi maksimum yang memungkinkan tanpa menyebabkan motor *stepper* kehilangan *step*. Kemudian dilanjutkan dengan menaikkan f_{step} secara berangsur angsur (dilakukan dengan mengurangi nilai *OCR0A*). Hal ini menyebabkan motor *stepper* dipercepat secara berangsur-angsur. Apabila dimulai dengan frekuensi *step* yang tinggi, misalnya memilih secara langsung frekuensi *step* 27027.03 Hz, motor *stepper* tidak akan berputar. Hal yang sama berlaku pada saat motor *stepper* diperlambat dengan cara pemberian nilai OCR0A yang tinggi secara tidak langsung.

Tabel 3. Hasil percepatan dan perlambatan motor *stepper*

$\frac{\mu steps}{step}$	OCR0A	Frekuensi f_{step}(Hz)	Hasil
16	0x35	18518.52	
16	0x34	18867.92	P
16	0x33	19230.77	E
16	0x32	19607.84	R
16	0x31	20000.00	C
16	0x30	20408.16	E
16	0x29	23809.52	P
16	0x28	24390.24	A
16	0x27	25000.00	T
16	0x26	25641.03	A
16	0x25	26315.79	N
16	0x24	27027.03	
16	0x25	26315.79	P
16	0x26	25641.03	E
16	0x27	25000.00	R
16	0x28	24390.24	L
16	0x29	23809.52	A
16	0x30	20408.16	M
16	0x31	20000.00	B
16	0x32	19607.84	A
16	0x33	19230.77	T
16	0x34	18867.92	A
16	0x35	18518.52	N

Percobaan tambahan yang juga merupakan percobaan terakhir pada penelitian ini adalah aplikasi rangkaian ini dengan menggunakan motor *stepper* tipe *hybrid*. Percobaan ini menunjukkan hasil yang sama seperti hasil yang dilakukan pada motor stepper tipe *permanent magnet*.

5. Kesimpulan

Dari hasil penelitian ini dapat disimpulkan beberapa hal sebagai berikut:

a. Perbaikan resolusi rotor dapat dilakukan dengan metode *software* yaitu dengan menggunakan berbagai mode *stepping*.
b. Motor *stepper driver* menyediakan fasilitas *internal indexer* yang mempunyai kemampuan untuk mengeksekusi *high accuracy microstepping* tanpa mengharuskan mikrokontroler untuk mengontrol level arus.
c. *Timer* pada mikrokontroler dapat digunakan untuk menghasilkan suatu *square wave* yang akan menjadi *input* pada pin *STEP* dari motor *stepper driver*.
d. Pada percobaan dengan menggunakan masukan berbagai pola *stepping*, terbukti bahwa dengan menggunakan level *microstepping* yang lebih tinggi menyebabkan suara motor *stepper* menjadi lebih tidak terdengar.
e. Pada percepatan dan perlambatan, bisa dilihat bahwa dengan merancang suatu program dengan profil percepatan dan perlambatan, maka pergerakan motor stepper tidak akan mengalami kemacetan.
f. Berbagai pergerakan rotasional yang bermanfaat bisa dilakukan dengan cara mengatur program (*software*) yang berisi pembangkitan gelombang *square wave.*
g. Rangkaian pengontrol ini juga dapat digunakan untuk tipe motor *stepper hybrid*.

Daftar Pustaka

[1] Atmel, "Atmel 8-Bit Microcontroller With 4/8/16/32kbytes In-System Programmable Flash Datasheet", Atmel Corporation, 2014

[2] Günther Gridling, Bettina Weiss, "Introduction to Microcontrollers", Vienna University of Technology Institute of Computer Engineering Embedded Computing Systems Group, 2007

[3] Ifrah Jaffri1, Zeeshan Nafees, Shoaib Zaidi, Oliver Faust, "BlueSteps: A Bluetooth Based Stepper Motor Control System", School of Science and Engineering, Habib University, Karachi, Pakistan Electrical, Electronic Control Engineering, Sheffield Hallam University, Sheffield, England, UK *Journal of Embedded Systems, , Vol. 3, No. 1, 21-27,2015*

[4] Ivan Virgala, Michal Kelemen, Alexander Gmiterko, Tomáš Lipták, " Control of Stepper Motor by Microcontroller", Department of Mechatronics, Technical University of Košice, Faculty of Mechanical Engineering, Košice, Slovakia, Journal of Automation and Control, 2015, Vol. 3, No. 3, 131-134

[5] Marc McComb, "Introduction to Stepper Motors Part 1 : Types of Stepper Motors", Microchip, 2007

[6] Marc McComb, "Introduction to Stepper Motors Part 2 : Stepper Motor Control", Microchip, 2007

[7] Schneider Electric, "NEMA size 23 1.8° 2-phase bipolar stepper motor Data Sheet", Schneider Electric Motion USA,

[8] Texas Instrument, "DRV8825 Stepper Motor Controller", 2014

Biodata Penulis

Arief Wisnu Wardhana, Menyelesaikan pendidikan S1 Engineering dalam bidang Electronic & Information dari University of Huddersfield, Huddersfield, The United Kingdom pada tahun 1997. Kemudian menyelesaikan Pasca Sarjana S2 Konsentrasi Sistem Isyarat Elektronik di Jurusan Teknik Elektro Fakultas Teknik Universitas Gadjah Mada Yogyakarta pada tahun 2016. Saat ini, penulis adalah dosen Jurusan Teknik Elektro, Fakultas Teknik Universitas Jenderal Soedirman, Purwokerto, Indonesia, bidang keahlian Sistem Isyarat dan Kendali. Interest utama pada area embedded system dan embedded programming.

Daru Tri Nugroho, Menyelesaikan pendidikan S1 Teknik dalam bidang Teknik Elektro dari Institut Teknologi Sepuluh November Surabaya, Indonesia pada tahun 2001. Kemudian menyelesaikan S2 Teknik dalam bidang Teknik Elektro dari Universitas Indonesia, Indonesia pada tahun 2006. Saat ini penulis adalah dosen Jurusan Teknik Elektro, Fakultas Teknik Universitas Jenderal Soedirman, Purwokerto, Indonesia, bidang sistem tenaga.