Institute of Research and Publication Indonesia
IJEERE: Indonesian Journal of Electrical Engineering and Renewable Energy
Journal Homepage: https://journal.irpi.or.id/index.php/ijeere
Vol. 2 Iss. 1 Juni 2022, pp: 39-46
P-ISSN: 2797-1155 E-ISSN: 2797-0868

Programming Algorithm on MCS-51 for Alphanumeric Liquid Crystal Display(LCD) Simplified Driver Circuit

Algoritma Pemrograman Berbasis MCS-51 Untuk Simplifikasi Rangkaian Driver Alphanumeric-Liquid Crystal Display(LCD)

Putut Son Maria[1*], Elva Susianti[2]

[1]Teknik Elektro, UIN Sultan Syarif Kasim, Indonesia
[2]Teknik Elektronika, Politeknik Caltex Riau, Indonesia

Corresponden E-Mail: [1]putut.son@uin-suska.ac.id

Makalah: Diterima 30 Mei 2022; Diperbaiki 8 Juni 2022; Disetujui 12 Juni 2022
Corresponding Author: [1]putut.son@uin-suska.ac.id

Abstrak

Pengkabelan antara mikrokomputer(minimum sistem atau mikrokontroler) dengan LCD alfanumerik secara standar mode 8 bit membutuhkan 11 pin sambungan kabel. Pada *chip* mikrokomputer yang kompak seperti AT89C2051 yang hanya memiliki 15 pin *input-output*(I/O) pengkabelan dengan LCD alfanumerik dengan cara standar menjadi tidak efisien. Sebanyak 11 pin dari 15 pin *resource* I/O dari *chip* mikrokomputer tersebut akan habis terpakai hanya untuk keperluan tampilan ke LCD. Penggunaan *resource* I/O untuk LCD dapat dihemat dengan menggunakan KIT I2C berbasis ICPCF8574C dengan konsekwensi timbulnya biaya dan konfigurasi ulang fungsi pin. Namun kendala lain adalah *library driver* I2C untuk setiap *brand* mikrokomputer belum tentu tersedia secara *default* pada *compiler* standar. Penelitian ini menampilkan alternatif solusi atas permasalahan tersebut dengan memanfaatkan fungsi *serial to paralel* yang dimiliki IC TTL 74595. Fungsi logika dari IC 74595 diimplementasikan dalam bentuk *script* program yang berperan sebagai program *driver* LCD. Uji simulasi membuktikan bahwa algoritma yang merepresentasikan fungsi *serial to paralel* pada penelitian ini berhasil menampilkan kinerja sistem secara baik dan normal dimana penggunaan *resource* I/O yang diperlukan hanya 3 pin yang artinya terjadi penghematan sebesar 72.72% dibandingkan jika menggunakan *wiring* secara standar 11 pin.

Keyword: MCS-51, LCD, program *driver*

Abstract

Wiring between microcomputer(minimum system either microcontroller) and alphanumeric LCD on 8 bit standart mode requires 11 pins. Assignment on large number of I/O pins designated for LCD interfacing is inefficient and unsuitable when applied for compact microcontroller AT89C2051 which has only 15 I/O pins. Interfacing to LCD can be simplified using I2C LCD KIT with additional cost and reconfiguring the functionality of I/O as consequences. The availability of library as a driver program on IDE compiler is uncertain since there are many brands compete on microcomputer as their product. This research exhibits serial to parallel method to simplify and save I/O resources utilizing a digital 74595 functional logic into a script of program to drive and communicate with LCD. The algorithm was tested on VSM Proteus application program and result shows a good agreement between design and visualization.This study has shown that the wiring technique and algorithm works well yielding 72.2% more efficient compared to standard wiring.

Keyword: MCS-51, LCD, driver program

1. Pendahuluan

Mikrokomputer merupakan perangkat digital berdimensi kompak yang dapat diprogram, banyak digunakan untuk aplikasi praktis, menjadikannya populer dipelajari siapapun, dan menjadi materi penting di kurikulum akademik baik perguruan tinggi dan bahkan sekolah menengah. Mikrokomputer dapat berupa sebuah minimum sistem ataupun mikrokontroler yang tersedia dengan berbagai varian dan banyak pilihan *brand*. Pada aplikasi praktis, mikrokomputer sering dipasangkan dengan LCD sebagai perangkat penampil informasi. LCD juga merupakan salah satu perangkat I/O dasar yang tertulis dalam buku untuk pembelajaran mikrokontroler[1] dan menjadi salah satu topik materi dalam pengajaran di perguruan tinggi[2].

Standar LCD alfanumerik yang mudah didapatkan biasanya berbasis *chip* HD44780 dengan spesifikasi 16 kolom - 2 baris yang memiliki 16 pin untuk pengkabelan dimana 8 pin sebagai jalur data, 3 pin sebagai kontrol dan sisanya untuk jalur catu daya dan kecerahan, dengan demikian untuk pengkabelan antara mikrokomputer dengan LCD membutuhkan 11 pin jalur. Pengkabelan dengan cara standar cukup banyak menggunakan *resource* I/O dan kurang efisien diterapkan pada sebuah *chip* mikrokontroler kompak seperti pada AT89C2051 yang memprioritaskan optimasi *resource* dari sistem. Cara mudah yang biasa dilakukan untuk mengurangi kebutuhan jumlah pin pengkabelan adalah menggunakan KIT I2C LCD yang berbasis chip ICPCF8574C yang pada dasarnya adalah *chip* dengan fungsi *serial to paralel*[3]. Cara tersebut juga populer digunakan pada sistem yang berbasis arduino[4-6] dimana pengguna dimudahkan dengan adanya *library* untuk I2C LCD pada arduino IDE. Konsekuensi dari cara tersebut adalah perlu biaya tambahan untuk pembelian KIT dan pengguna perlu memahami urutan kerja sistem I2C yang sudah terstandarkan dimana mekanismenya terdiri dari banyak pewaktuan untuk *handshaking* antara mikrokomputer dengan perangkat eksternal.

Membangun pengkabelan antara mikrokomputer dengan LCD secara lebih ringkas dapat dilakukan dengan menetapkan konfigurasi jalur data LCD pada mode 4 bit [7][9], dengan demikian secara logis maka dengan menggunakan IC TTL 74595 yang memiliki fungsi dasar sebagai *serial to paralel* tentu akan dapat digunakan sebagai *driver* LCD. Rangkaian *driver* harus dikendalikan oleh program yang fungsi kerjanya mengacu pada keselarasan pewaktuan sinyal antara IC TTL 74595 dengan *chip* HD44780.

Pada penelitian ini algoritma untuk mengendalikan rangkaian *driver* LCD berbasis IC TTL 74595 diimplementasikan menggunakan bahasa program C yang ditujukan untuk dijalankan pada mikrokomputer dari keluarga MCS-51. Hal ini dilakukan karena *library* untuk mengendalikan LCD secara serial tidak tersedia untuk mikrokomputer MCS-51 dan sekaligus meralat biasnya penjelasan terkait pemrograman *driver* LCD menggunakan IC TTL 74595 pada [8] sehingga informasi yang ditampilkan dalam paper ini akan membantu menjelaskan *programmer* yang ingin melakukan kreasi *library* sendiri jika akan diaplikasikan untuk *brand* mikrokomputer berbeda.

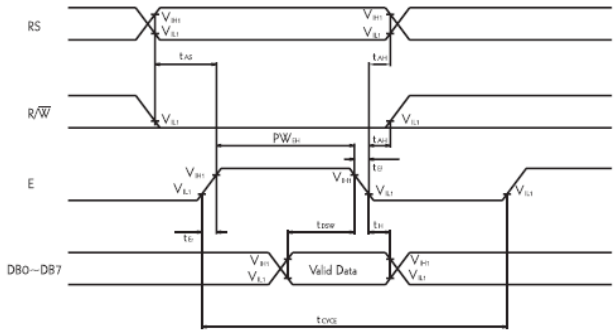
2. Materi dan Metode

Obyek target pada penelitian ini adalah LCD berbasis *chip* HD44780 dan IC TTL 74595 dimana mekanisme kerja LCD adalah bahwa data(*input*) harus tersedia secara simultan sedangkan pada IC TTL 74595 *output*-nya membentuk urutan pola secara sekuensial. Pewaktuan dan mekanisme pada jalur data LCD tidak dapat direkayasa selain harus paralel. Pada IC TTL 74595 walaupun pada jalur *output*-nya tersusun atas *flip-flop* yang terangkai secara *cascade* namun dengan adanya pin yang berfungsi sebagai *clock* maka data yang tersedia pada jalur *output* dapat dikondisikan sehingga pada saat LCD membaca data dari IC TTL 74595 akan diterima sebagai data simultan. Hal ini dapat tercapai dengan mengatur pewaktuan antara pin *clock* pada IC TTL 74595 dan pin *enable* pada *chip* HD44780.

2.1 Diagram Pewaktuan LCD

Berdasarkan [9] tentang diagram pewaktuan proses tulis(*write*) LCD seperti pada gambar 1, syarat dan momen penting dalam penulisan data adalah bahwa pin R/W harus berlogika 0 dan pin *enable*(E) harus terjadi transisi dari logika *high* menuju *low*(*falling edge*) setelah semua data siap tersedia pada jalurnya. Dengan demikian, penghematan dapat dilakukan dengan cara menetapkan pin R/W langsung terhubung dengan *ground*. Premis yang akan digunakan sebagai acuan pada algoritma adalah(sesuai urutan waktu) :

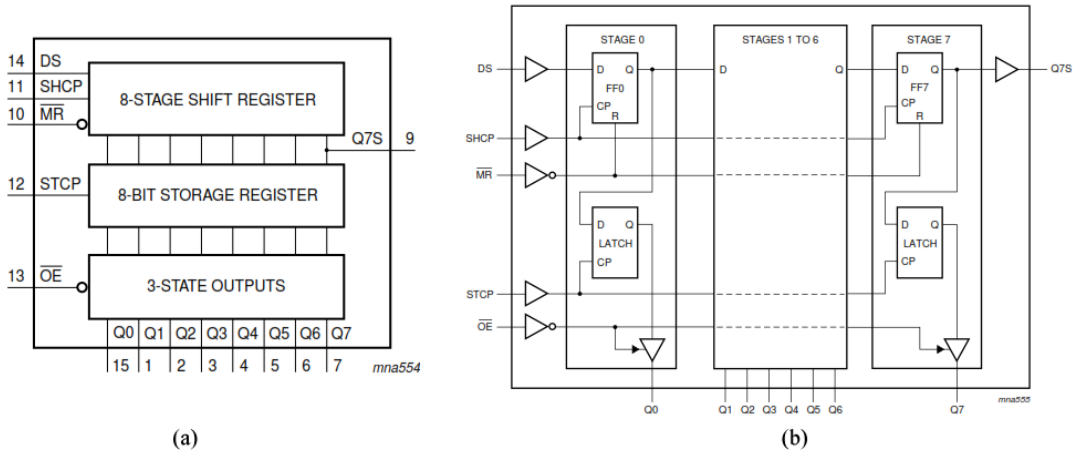
- data telah siap tersedia pada jalurnya; pada penelitian ini menggunakan mode 4 bit.
- pin RS berlogika sesuai register yang akan ditulisi
- pin E diberikan logika *falling edge*



Gambar 1. Diagram pewaktuan proses tulis(*write*) LCD

2.2 Diagram Pewaktuan IC TTL 74595

Blok diagram IC TTL 74595 ditunjukkan pada gambar 2.a. Mekanisme kerjanya adalah dimulai dari 8-*stage shift register* yaitu unit yang terdiri dari delapan *flip-flop* tipe D yang dirangkai secara *cascade*. Setiap pulsa *clock* (pin SHCP) pada unit ini mengakibatkan data keluaran *flip-flop* akan bergeser dari tingkat n ke n+1. Unit *8-bit storage register* berperan sebagai unit penduplikat data dari *shift register*, setiap pulsa yang masuk pada pin *clock*(STCP) akan menghasilkan data *output* dari *shift register* menjadi terduplikasi pada *flip-flop storage register* sehingga data akan tersedia secara *paralel* setelah melewati unit *buffer*. Data *output* akhir dari IC TTL 74595 hanya akan tersedia pada terminal Q0 sampai Q7 jika pin kontrol OE berlogika *low* karena terdapat gerbang NOT seperti pada gambar 2.b.



Gambar 2. Blok Diagram IC TTL 74595

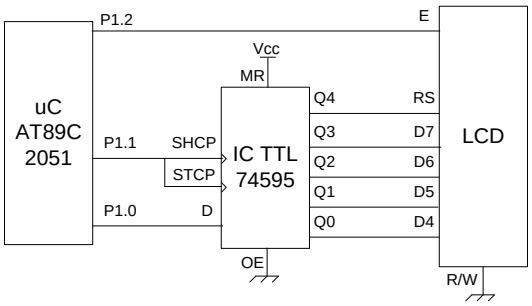
Fungsi logika dari IC TTL 74595 ditampilkan pada tabel 1. Pin kontrol SHCP dan STCP dapat saling ditautkan untuk menghemat pengkabelan dengan tetap mempertahankan fungsi *serial to paralel* dengan kondisi pin OE terhubung ke *ground*. Berdasarkan tabel 1 dijelaskan bahwa *raising edge* pada pin SHCP dan STCP akan menggeser data dan sekaligus mempertahankan data agar tetap tersedia pada terminal *output* Q0-Q7. Struktur rangkaian *cascade* pada FF0 dan *latch* seperti pada gambar 2.b artinya membutuhkan satu *clock* ekstra untuk mengeksitasi *output* dari unit *shift register* agar terduplikasi ke unit *storage register*.

Tabel 1. Fungsi logika IC TTL 74595

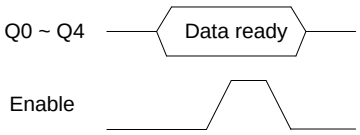
Control				Input	Output		Function
SHCP	STCP	$\overline{OE}$	$\overline{MR}$	DS	Q7S	Qn	
X	X	L	L	X	L	NC	a LOW-level on $\overline{MR}$ only affects the shift registers
X	↑	L	L	X	L	L	empty shift register loaded into storage register
X	X	H	L	X	L	Z	shift register clear; parallel outputs in high-impedance OFF-state
↑	X	L	H	H	Q6S	NC	logic HIGH-level shifted into shift register stage 0. Contents of all shift register stages shifted through, e.g. previous state of stage 6 (internal Q6S) appears on the serial output (Q7S).
X	↑	L	H	X	NC	QnS	contents of shift register stages (internal QnS) are transferred to the storage register and parallel output stages
↑	↑	L	H	X	Q6S	QnS	contents of shift register shifted through; previous contents of the shift register is transferred to the storage register and the parallel output stages

2.3 Rancangan Pengkabelan dan Pewaktuan Sinyal

Strategi dalam penulisan *script* program didasarkan pada rancangan skema pada gambar 3. Jalur data D4~D7 dari LCD terhubung dengan Q0~Q3 dari IC TTL 74595. Pin RS dihubungkan dengan Q4 yang akan berperan untuk seleksi register perintah atau register data pada LCD[9]. Logika pada pin RS dapat dipersiapkan secara serentak bersamaan dengan 4 bit(*nibble*) data pada Q0~Q3 sebelum *clock* pada pin *enable* diberikan. Diagram pewaktuan dari strategi pemrograman ditunjukkan pada gambar 4. Transisi *falling edge* pulsa pada pin *enable* dapat diberikan selama data *nibble* dan logika untuk pin RS sudah dipersiapkan sesaat sebelum terjadinya transisi, tetapi pin *enable* tidak seharusnya berlogika *high* jika masih terjadi perubahan pada jalur data D4~D7 atau data belum *steady*, jika terjadi demikian maka LCD akan membaca data yang ada sebagai data baru. Ketentuan inilah yang gagal dijelaskan oleh [8] seperti terlihat pada contoh *script* program yang ditampilkan dalam laman situs web-nya.



Gambar 3. Blok Diagram Pengkabelan Yang Diusulkan



Gambar 4. Diagram Pewaktuan Pengiriman Satu *Nibble*

3. Hasil dan Pembahasan

3.1 Hasil *Run* Kode Program

Algoritma program untuk pengendali *driver* LCD diimplementasikan penulisannya dalam bahasa program C dan dikompilasi menggunakan *small device c-compiler*(SDCC) dalam program aplikasi *compiler* M-IDE Studio for MCS-51[10]. *Library* yang diperlukan terdiri dari satu *library* standar *compiler* C : string.h, satu *library* dari M-IDE Studio : at89x51.h dan satu *library* yang dikreasikan sendiri : delay.h. Tiga pin pengendali : pin_data(D), pin_clk(SHCP dan STCP) dan pin_enable(E) yang dirancang pada gambar 3 dinyatakan pada bagian definisi di awal program untuk mempermudah pembacaan program seperti pada gambar 5. Eksitasi pada pin *enable* dan *clock* memiliki kesamaan yaitu berupa transisi dari logika *high* menuju *low*, namun karena jalur yang digunakan berbeda, maka penulisan *function* tetap harus dibuat secara tersendiri.

```
#include<at89x51.h>
#include<string.h>
#include<delay.h>

#define pin_data P1_0
#define pin_clk P1_1
#define pin_enable P1_2

unsigned char Mask, RS;

void pulse_enable()
{
pin_enable = 1;delay_ms(1);pin_enable = 0;
}

void clock(){
pin_clk = 1;delay_ms(1);pin_clk = 0;
}
```

Gambar 5. Kode Program Definisi Fungsi Pin

Penterjemahan fungsi konversi dari serial menjadi paralel dinyatakan seperti *script* pada gambar 6. Logika untuk pin RS harus dikirimkan pertama, disusul dengan data untuk bit 3 hingga 0 secara berurutan. Hal ini karena setiap setelah terjadi *clock* maka *output* FF ke-(n) akan digeser ke FF ke-(n+1). Sesuai dengan pemaparan di pembahasan poin 2.2, *clock* pertama berfungsi sebagai *clock* ekstra karena digunakan untuk mengeksitasi FF0 agar *output*-nya dapat diterima oleh FF *latch(storage register)*. Dengan demikian untuk jumlah bit data sebanyak 1 *nibble*(4 bit) dan 1 bit untuk RS maka diperlukan *clock* sejumlah 6 pulsa *clock*.

```
void kirim_nibble(unsigned char data_in)
{
pin_data = RS;
clock(); // RS -> Q FF0, Q0 = x
Mask = 0x08;
pin_data = data_in & Mask;
clock(); // RS -> latch 0 & D(bit 3) -> Q FF0
Mask = Mask >> 1;
pin_data = data_in & Mask;
clock(); // RS -> latch 1 & D(bit 2) -> Q FF0
Mask = Mask >> 1;
pin_data = data_in & Mask;
clock(); // RS -> latch 2 & D(bit 1) -> Q FF0
Mask = Mask >> 1;
pin_data = data_in & Mask;
clock(); // RS -> latch 3 & D(bit 0) -> Q FF0
clock(); // RS -> Q4 , D3~D0 -> Q3~Q0
}
```

Gambar 6. Kode Program Konversi *Serial to Paralel*

LCD menterjemahkan data dalam satuan *byte*(8 bit), sehingga jika mode pengkabelan ditetapkan 4 bit maka data harus dikirimkan sebanyak dua kali untuk agar menjadi genap satu *byte*. Kode program untuk pengiriman data delapan bit baik untuk pemberian kode perintah atau data tampilan ditunjukkan pada gambar 7. Perbedaan kedua *function* tersebut adalah pada target register yang dituju, dimana pengiriman kode perintah harus diberikan dengan kondisi RS diberikan logika nol, dan untuk pengiriman data karakter maka RS harus berlogika satu[9]. Sesuai dengan perencanaan pada gambar 4, pulsa untuk *enable* diberikan setelah data tersedia pada jalur Q0~Q4, secara penulisan program maka *function* untuk *enable* diletakkan setelah *function* pengiriman *nibble*.

```
void kirim_data(unsigned char data_8bit)
{
RS = 1;
kirim_nibble(data_8bit/16);delay_ms(1); //high nibble
pulse_enable();
kirim_nibble(data_8bit);delay_ms(1); //low nibble
pulse_enable();
}

void kirim_perintah(unsigned char data_8bit)
{
RS = 0;
kirim_nibble(data_8bit/16);delay_ms(1); //high nibble
pulse_enable();
kirim_nibble(data_8bit);delay_ms(1); //low nibble
pulse_enable();
}
```

Gambar 7. Kode Program Pengiriman Data 8 Bit

LCD harus diinisialisasi dengan cara mengirimkan data tertentu yang akan diterjemahkan sebagai pengaturan konfigurasi di register LCD. Prosedur untuk inisialisasi LCD sesuai dengan [11] adalah mereset dan mengirimkan *byte* data tertentu secara berurutan seperti pada gambar 8.

```
void init_LCD()
{
// reset
kirim_perintah(0x03);
delay_ms(1);
kirim_perintah(0x02);
delay_ms(1);
//inisialisasi
kirim_perintah(0x28);
delay_ms(1);
kirim_perintah(0x0c);
delay_ms(1);
kirim_perintah(0x06);
delay_ms(1);
kirim_perintah(0x80);
delay_ms(1);
}
```

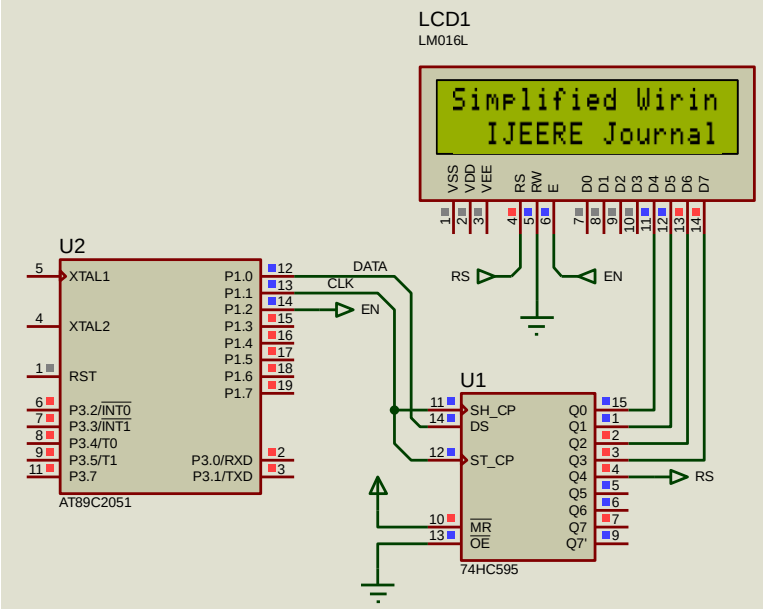
Gambar 8. Kode Program Inisialisasi LCD

Pada penelitian ini menyasar untuk pengiriman *string* menggunakan *function* yang sudah dibahas sebelumnya. Pengiriman data bertipe *string* menghemat baris pada *script* program dan memudahkan pengelolaan variabel dalam program. Contoh *function* untuk pengiriman *string*, pengaturan kursor dan program utama ditunjukkan pada gambar 9. Pada dasarnya data bertipe *string* secara teknis akan dikirimkan secara karakter per karakter. Pada *brand* lain, *function* untuk pekerjaan tersebut telah tersedia pada aplikasi *compiler*-nya, tetapi untuk MCS-51 hanya tersedia untuk mode pengkabelan standar 8 bit. Pada penelitian ini juga mencontohkan *function* untuk pengaturan posisi tampilan. Penentuan posisi tampilan dilakukan dengan cara mengarahkan kursor pada alamat tertentu pada DD-RAM dalam LCD dan data yang dikirimkan untuk menentukan koordinat kursor ditujukan pada register perintah[11].

```
void kirim_pesan(char pesan[])
{
unsigned char indek;
for(indek=0;indek < strlen(pesan);indek++)
        {
        kirim_data(pesan[indek]);
        }
}

void lcd_gotoxy(unsigned char row, unsigned char col)
{
if (row = 1) kirim_perintah(0x80 + col);
if (row = 2) kirim_perintah(0xC0 + col);
}

void main()
{
char message1[] = "Simplified Wiring";
char message2[] = "IJEERE Journal";

init_LCD();
kirim_pesan(message1);lcd_gotoxy(2,2);
kirim_pesan(message2);
}
```

Gambar 9. Kode Program Utama dan Pengiriman *String*

Rangkaian uji simulasi ditunjukkan pada gambar 10. Program simulator yang digunakan pada penelitian ini adalah Proteus Versi 8. Sistem penggambaran skema rangkaian pada Proteus tidak selalu harus menggambarkan semua komponen secara rinci dan komplit. Hal ini karena ada pengaturan dan fitur yang sudah mengakomodir substansi-substansi pokok seperti catu daya, frekuensi sinyal, *initial condition* dan lain sebagainya. Oleh karena itu skema pada gambar 10 sengaja tidak memasang komponen kristal osilator, rangkaian reset pada mikrokomputer AT89C2051dan catu daya untuk LCD karena alasan tersebut di atas. Hasil *running* program yang sudah dipaparkan di atas menunjukkan bahwa tampilan pada LCD sesuai dengan *script* program pada gambar 9. Semua *function* yang dirancang dapat bekerja secara normal meskipun dengan konfigurasi pengkabelan yang sederhana. Dibandingkan dengan pengkabelan secara standar 8 bit paralel dimana membutuhkan 11 pin *resource* I/O, ternyata dengan memanfaatkan fitur IC TTL 74595 dan sedikit program pengendali membuktikan bahwa informasi visual tetap dapat diwujudkan dengan keuntungan terjadi penghematan *resource* I/O sebanyak 8 pin.



Gambar 10. Hasil Simulasi Menggunakan Aplikasi Proteus

3.2 Perhitungan Capaian Penghematan *Resource* I/O

Pengkabelan standar(mode 8 bit-data) memerlukan total sebanyak 11 jalur I/O untuk komunikasi antara mikrokomputer dengan LCD, terdiri dari 8 bit untuk data dan 3 bit untuk kontrol. Teknik penyederhanaan yang diusulkan pada penelitian ini menggunakan hanya 3 pin yang berperan sebagai kontrol dan data. Perbandingan antara mode standar dengan mode simplifikasi ditunjukkan pada tabel 2.

Tabel 1. Perbandingan mode standar 8 bit dan mode tersederhanakan

Mode standar	Mode simplifikasi
Data : 8 pin	Data : 1 pin
Kontrol : 3 pin	Kontrol : 2 pin
Total : 11 pin	Total : 3 pin

Selisih jumlah pin yang terpakai berkurang sebanyak 11 – 3 = 8 pin. Rasio antara selisih jumlah pin dibandingkan dengan total jumlah pin terpakai menggunakan mode standar, senilai dengan penghematan sebesar :

$$\text{Penghematan } \textit{resource} \text{ I/O} = \frac{8}{11} x100\ \% = 72.72\ \%$$

4. Kesimpulan

Efisiensi penggunaan *resource* I/O pada sebuah mikrokomputer dapat dioptimalkan dengan cara mengubah jalur komunikasi data secara serial. Penggunaan mode komunikasi secara serial dapat dicapai dengan cara menyeselarasankan antara program pengendali(kontrol) dan perangkat keras pendukung sebagai *driver* LCD. Algoritma kendali yang ditampilkan dan diuji pada penelitian ini berhasil membuktikan bahwa komunikasi antara mikrokomputer dengan LCD bisa dilakukan dengan memanfaatkan satu buah IC TTL 74595 sebagai *driver* dengan hasil capaian terjadi penghematan *resource* I/O sebesar 72.72% dibandingkan teknik pengkabelan LCD secara standar paralel. Angka tersebut terhitung berdasarkan reduksi dari 11 pin menjadi hanya 3 pin pada contoh model skema yang diusulkan pada penelitian ini.

Referensi

[1] D. Suprianto, V.A.H. Firdaus, R. Agustina, D.W. Wibowo, *Microcontroler Arduino Untuk Pemula*, Penerbit Jasakom-Malang, 2019.

[2] Modul Praktikum Mikrokontroler Universitas Pembangunan Jaya, [Daring]. Tersedia pada : https://ocw.upj.ac.id/files/RPS-INF204-INF204-Modul-Mikrokontroler-dan-Project.pdf [Diakses : 20 Mei 2022]

[3] H. Suryantoro, A. Budiyanto, "Prototipe Sistem Monitoring Level air Berbasis Labview & Arduino Sebagai Sarana Pendukung Praktikum Instrumentasi Sistem Kendali," *Indonesian Journal of*

Laboratory, vol. 1, no. 3, pp. 20–32, 2019.
[4] R. Sandra, V. Simbar, A. Syahrin, "Prototipe Sistem Monitoring Temperatur Menggunakan Arduino Uno R3 Dengan Komunikasi Wireless," *Jurnal Teknologi Elektro*, vol. 8, no. 1, pp. 80–86, 2017.
[5] MD. A. Saputra, Amarudin, N. Utami, R. Setiawan, "Rancang Bangun Alat Pemberi Pakan Ikan Menggunakan Mikrokontroler," *J. ICTEE*, vol. 1, no. 1, p. 15-19, 2020.
[6] Akinwole OO, Oladimeji TT (2018) Design and Implementation of Arduino Microcontroller Based Automatic Lighting Control with I2C LCD Display. *J Electr Electron Syst* 7: 258. doi: 10.4172/2332-0796.1000258
[7] T. Pan, Y. Zhu, "Designing Embedded Systems with Arduino," Springer Nature Singapore, 2018.
[8] R. Bhatt, "3-Wire Serial LCD Using a Shift Register," [Daring]. Tersedia pada: https://www.electronics-lab.com/project/3-wire-serial-lcd-using-a-shift-register/ [Diakses : 20-Mei-2022]
[9] M1632 Datasheet, [Daring]. Tersedia pada : https://www.datasheetarchive.com/M1632*%20lcd%20display-datasheet.html [Diakses : 18-Mei-2022]
[10] Worapoht Kornkaewwattanakul, "M-IDE Studio for MCS-51,"Tersedia pada : www.opcube.com
[11] LCD datasheet, Tersedia pada : https://www.datasheetarchive.com/M1632*%20lcd%20display-datasheet.html