Simulasi Port I/O pada Atmega8535 Terhubung dengan Proteus

Diperuntukkan untuk memenuhi salah satu tugas praktikum Mata Kuliah Aplikasi Mikrokontroler

Praktikum	: Aplikasi Mikrokontroler
Praktikum ke	: 6
Tanggal Praktikum	: Kamis, 5 November 2020
Tanggal Pengumpulan Laporan	: Minggu, 8 November 2020
Nama dan NIM	: 1. Amir Husein (181344003)
Kelas	: 3-TNK
Instruktur	: 1. Ferry Satria, BSEE., M.T
	2. Rahmawati Hasanah, S.ST., M.T

Politeknik Negeri Bandung

Tahun Ajaran 2020/2021

I. TUJUAN

- Mahasiswa dapat memahami prinsip penggunaan port pada Atmega8535
- Mahasiswa mampu melakukan koneksi antara simulator AVR Studio dengan simulator Proteus

II. LANDASAN TEORI

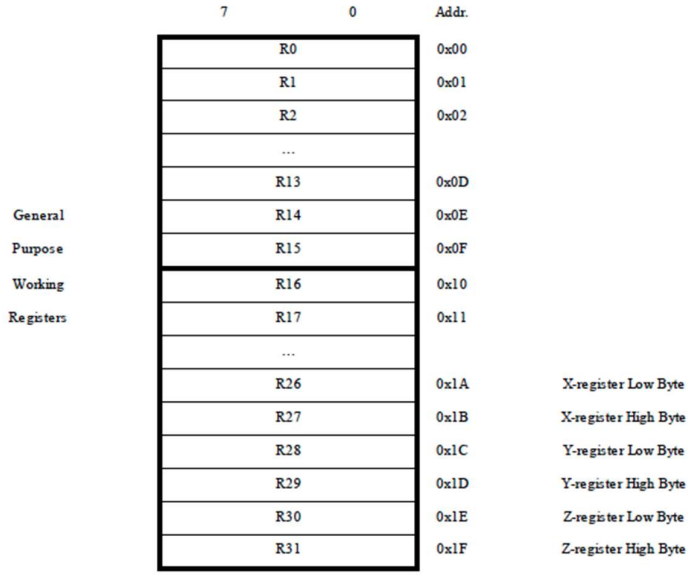
1. Mikrokontroler Atmega8535

   Mikrokontroler merupakan suatu chip komputer mini, dimana di dalamnya sudah terdapat sebuah mikprosesor disertai memori, baik itu RAM, ROM, maupun EEPROM. Selain itu, mikrokontroler telah memiliki sistem integrasi Input dan Output (I/O) yang telah dikemas sedemikian rupa pada rangkaian Integrated Circuitnya, sehingga memudahkan dalam melakukan tugas atau operasi tertentu.

   Atmega8535 merupakan sebuah mikrokontroler 8-bit yang dikeluarkan oleh perusahaan Atmel pada tahun 2006. Mikrokontroler ini pun memiliki flash memori sebesar 8kb serta EEPROM 512 byte. Selain itu, pada papan mikro ini pun sudah terdapat sebuah ADC dan 3 buah Timer sebagai pencacah waktu pemrosesan data.

2. General Purpose Register

   Register ini merupakan register umum yang dapat digunakan sebagai kontrol pengisian dan penyesuaian data. Pada Atmega8535, terdapat 32 general purpose register, yang penamaannya berupa rentang angka dari R0 sampai dengan R31.



Gambar 1. General Purpose Register pada Atmega8535

Terlihat pada gambar diatas, diantara R15 dan R16 terdpat sebuah garis tebal, hal ini menandakan bahwa register R15 kebawah tidak berlaku instruksi *immediate* atau langsung, sedangkan register R16 keatas berlaku. Pada general purpose register ini juga

terdapat register khusus sebagai pointer alamat 16-bit yaitu R2 hingga R21 yang terbagi menjadi 3 segmen yaitu pointer X, Y, dan Z.

3. Instruksi LPM dan LD

   Instruksi LPM (Load Program Memory) digunakan untuk melakukan *load* dari memori dengan tujuannya adalah register pointer. LPM digunakan untuk mendapatkan nilai yang tepat dari setiap alamat program memori yang dideklarasikan.

   Instruksi LD (Load from Data) digunakan untuk melakukan *load* dari memori, apabila LPM dari program area, maka LD dari data area.

4. I/O pada Atmega8535

   ATMEGA 8535 memilki empat buah port yang benar-benar memiliki fungsi untuk Read-Modify-Write ketika digunakan sebagai port General digital I/O. Driver pin pada I/O port cukup kuat untuk menggerakkan tampilan LED secara langsung. Semua pin port memiliki resistor pull-up yang dapat dipilih secara individual dengan resistansi invarian tegangan suplai.

   Dalam mengatur sebuah port I/O pada Atmega8535, kita perlu mengatur bit pada DDRX, X disini berdiri sebagai jenis port. Misal jika kita ingin mengatur port A, maka digunakan DDRA, begitupun dengan B hingga port D. Jika DDRX diberikan nilai 1 (high), maka pin digunakan sebagai output. Jika DDRX diberikan nilai 0 (low), maka pin difungsikan sebagai input. Selain DDRX, ada juga register PORTX digunakan untuk mengaktifkan pull-up resistor (pada saat pin difungsikan sebagai input), dan memberikan nilai keluaran pin high/low (pada saat difungsikan sebagai output). Terdapat juga PINX, sebuah register yang berfungsi untuk mengetahui keadaan tiap-tiap pin pada mikrokontroller.

III. SOAL LAB

1. Buat program untuk Binary up-counter 5 bit yang outputnya berubah 1 detik sekali. Tidak berulang.
2. Buat program untuk Binary up-counter 5 bit yang outputnya berubah 1 detik sekali, menghitung berulang terus menerus.
3. Buat program untuk Binary up-counter 8 bit yang outputnya berubah 4 kali tiap detik.
4. Buat program untuk Decimal up-counter Modulo 10 yang outputnya berubah 1 detik sekali. Gunakan 7 segmen common anode.
5. Buat program untuk Decimal up-counter Modulo 10 yang outputnya berubah 1 detik sekali. Gunakan 7 segmen common cathode.
6. Buat program untuk Hexa-Decimal down-counter Modulo 16 (MENGHITUNG SECARA BERULANG DARI F SAMPAI 0) yang outputnya berubah 1 detik sekali. Gunakan 7 segmen common anode.
7. Buat program untuk Hexa-Decimal up-counter Modulo 16 (MENGHITUNG SECARA BERULANG DARI 0 SAMPAI F) yang outputnya berubah 1 detik sekali. Gunakan 7 segmen common cathode.
8. Buat program untuk Oktal down-counter Modulo 8 (MENGHITUNG SECARA BERULANG DARI 7 SAMPAI 0) yang outputnya berubah 1 detik sekali. Gunakan 7 segmen common cathode.

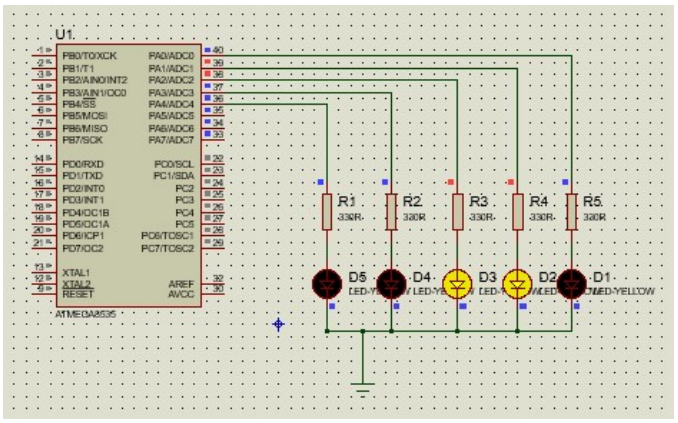
9. Buat program untuk Oktal up-counter 2 digit (MENGHITUNG SECARA BERULANG DARI 00 SAMPAI 77) yang outputnya berubah 1 detik sekali. Gunakan 2 unit 7 segmen common cathode.

IV. PROGRAM & HASIL

1. Program:

```
.INCLUDE "M8535DEF.INC"
.ORG 0

RJMP MAIN
MAIN:
LDI R16, LOW(RAMEND)
OUT SPL, R16
LDI R16, HIGH(RAMEND)
OUT SPH, R16

//PORT A SEBAGAI OUTPUT
LDI R20, $FF
OUT DDRA, R20

//R20 AKAN BERTINDAK SEBAGAI COUNTER
AWAL:
LDI R20, 0
MULAI:
OUT PORTA, R20
RCALL DELAY
INC R20 //00000111
CPI R20, 32  //DESIMAL  2^5
BRNE MULAI
//RJMP AWAL
AKHIR: RJMP AKHIR

//SUBRUTIN DELAY 1 DETIK
DELAY:
LDI R21, $33
L21:
LDI R22, 0
L22:
LDI R23, 0
L23:
DEC R23
BRNE L23
DEC R22
BRNE L22
DEC R21
BRNE L21
RET
```

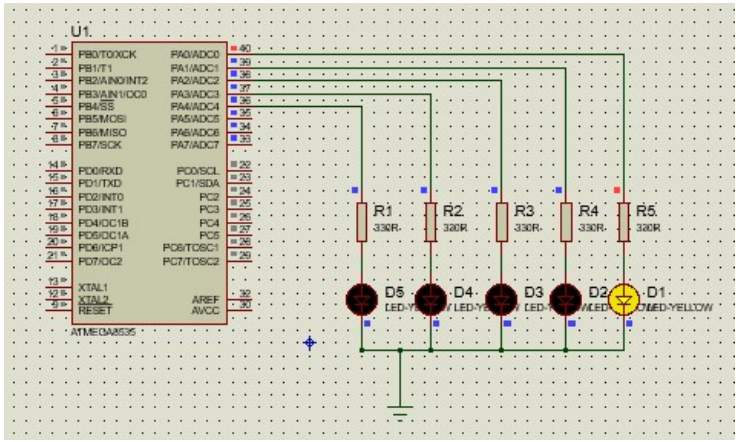
Hasil:



2. Program:

```
.INCLUDE "M8535DEF.INC"
.ORG 0

RJMP MAIN
MAIN:
LDI R16, LOW(RAMEND)
OUT SPL, R16
LDI R16, HIGH(RAMEND)
OUT SPH, R16

//PORT A SEBAGAI OUTPUT
LDI R20, $FF
OUT DDRA, R20

//R20 AKAN BERTINDAK SEBAGAI COUNTER
AWAL:
LDI R20, 0
MULAI:
OUT PORTA, R20
RCALL DELAY
INC R20 //00000111
CPI R20, 32  //DESIMAL  2^5
BRNE MULAI
RJMP AWAL
//AKHIR: RJMP AKHIR

//SUBRUTIN DELAY 1 DETIK
DELAY:
LDI R21, $33
L21:
LDI R22, 0
L22:
LDI R23, 0
L23:
DEC R23
BRNE L23
DEC R22
BRNE L22
DEC R21
BRNE L21
RET
```

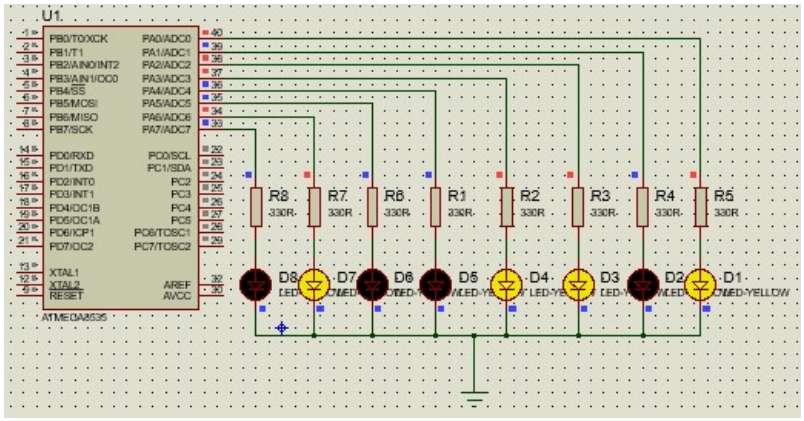
Hasil:



3. Program:

```
.INCLUDE "M8535DEF.INC"
.ORG 0

RJMP MAIN
MAIN:
LDI R16, LOW(RAMEND)
OUT SPL, R16
LDI R16, HIGH(RAMEND)
OUT SPH, R16

//PORT A SEBAGAI OUTPUT
LDI R20, $FF
OUT DDRA, R20

//R20=COUNTER
AWAL:
LDI R20, 0
MULAI:
OUT PORTA, R20
RCALL DELAY
INC R20
CPI R20, $00

BRNE MULAI
RJMP AWAL

//SUBRUTIN DELAY 1 DETIK
DELAY:
LDI R21, 8
L21:
LDI R22, 0
L22:
LDI R23, 0
L23:
DEC R23
BRNE L23
DEC R22
BRNE L22
DEC R21
BRNE L21
RET
```

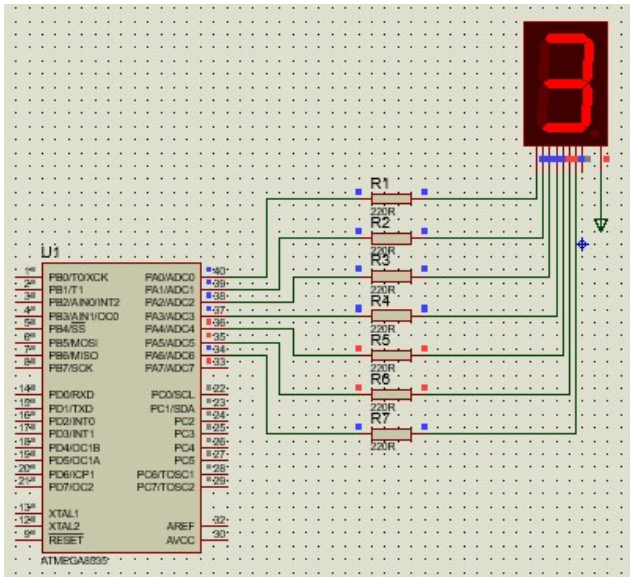
Hasil:



4. Program:

```
.INCLUDE "M8535DEF.INC"
.ORG 0

RJMP MAIN
MAIN:
LDI R16, LOW(RAMEND)
OUT SPL, R16
LDI R16, HIGH(RAMEND)
OUT SPH, R16
//PORT A SEBAGAI OUTPUT
LDI R20, $FF
OUT DDRA, R20
//INISIALISASI COUNTER
AWAL:
LDI R16,0
//INISIALIASI POINTER Z
LDI ZL, LOW(2*DAT)
LDI ZH, HIGH(2*DAT)
//MULAI
L1:
LPM R20, Z+
OUT PORTA, R20
RCALL DELAY
INC R16
CPI R16, $A
BRLO L1
RJMP AWAL

//SUBRUTIN DELAY 1 DETIK
DELAY:
LDI R21, 11
L21:
LDI R22, 0
L22:
LDI R23, 0
L23:
DEC R23
BRNE L23
DEC R22
BRNE L22
DEC R21
BRNE L21
RET

.ORG 80
DAT: .DB $C0, $F9, $A4, $B0, $99, $92, $83, $F8, $80, $90
```

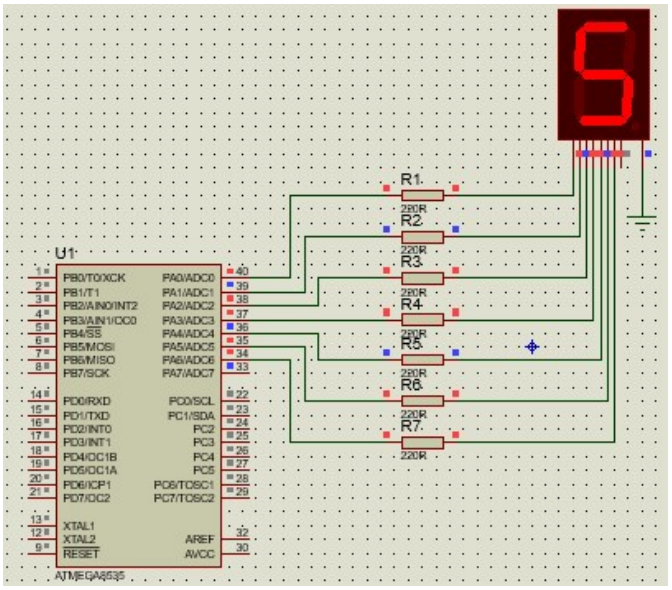
Hasil:



5. Program:

```
.INCLUDE "M8535DEF.INC"
.ORG 0

RJMP MAIN
MAIN:
LDI R16, LOW(RAMEND)
OUT SPL, R16
LDI R16, HIGH(RAMEND)
OUT SPH, R16

//PORT A SEBAGAI OUTPUT
LDI R20, $FF
OUT DDRA, R20

//INISIALISASI COUNTER
AWAL:
LDI R16,0

//INISIALIASI POINTER Z
LDI ZL, LOW(2*DAT)
LDI ZH, HIGH(2*DAT)

//MULAI
L1:
LPM R20, Z+
OUT PORTA, R20
RCALL DELAY
INC R16
CPI R16, $A
BRLO L1
RJMP AWAL
//SUBRUTIN DELAY 1 DETIK
DELAY:
LDI R21, 11
L21:
LDI R22, 0
L22:
LDI R23, 0
L23:
DEC R23
BRNE L23
DEC R22
BRNE L22
DEC R21
BRNE L21
RET

.ORG 80
DAT: .DB $3F, $06, $5B, $4F, $66, $6D, $7C, $07, $7F, $6F
```

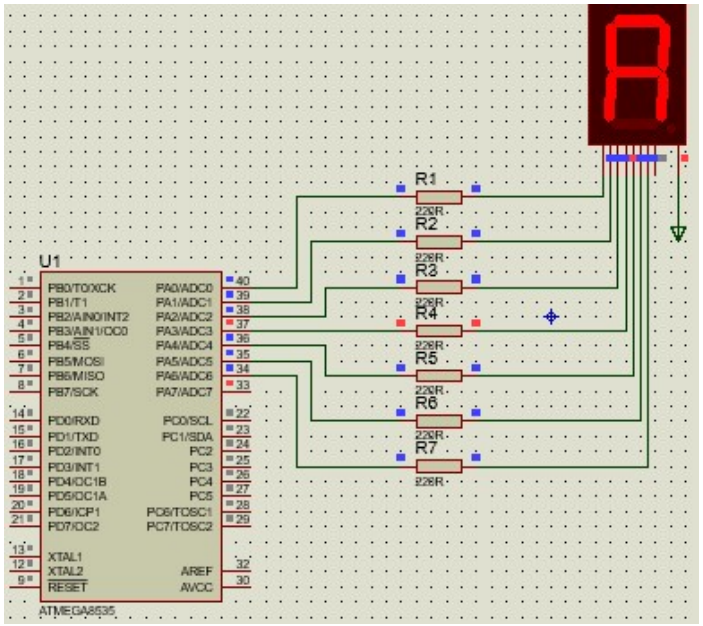
Hasil:



6. Program:

```
.INCLUDE "M8535DEF.INC"
.ORG 0

RJMP MAIN
MAIN:
LDI R16, LOW(RAMEND)
OUT SPL, R16
LDI R16, HIGH(RAMEND)
OUT SPH, R16

//PORT A SEBAGAI OUTPUT
LDI R20, $FF
OUT DDRA, R20

//INISIALISASI COUNTER
AWAL:
LDI R16,0

//INISIALIASI POINTER Z
LDI ZL, LOW(2*DAT)
LDI ZH, HIGH(2*DAT)

//MULAI
L1:
LPM R20, Z+
OUT PORTA, R20
RCALL DELAY
INC R16
CPI R16, $10
BRLO L1
RJMP AWAL
```

Hasil:

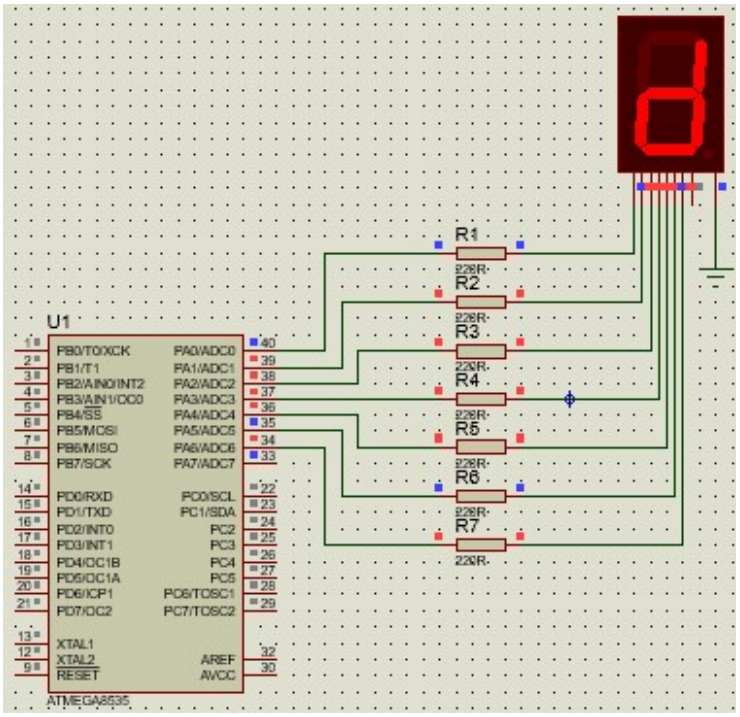


```
//SUBRUTIN DELAY 1 DETIK
DELAY:
LDI R21, 11
L21:
LDI R22, 0
L22:
LDI R23, 0
L23:
DEC R23
BRNE L23
DEC R22
BRNE L22
DEC R21
BRNE L21
RET

.ORG 80
DAT:
.DB $8E, $86, $A1, $C6, $83, $88, $90, $80, $F8, $82, $92, $99, $B0, $A4, $F9, $C0
```

7. Program:

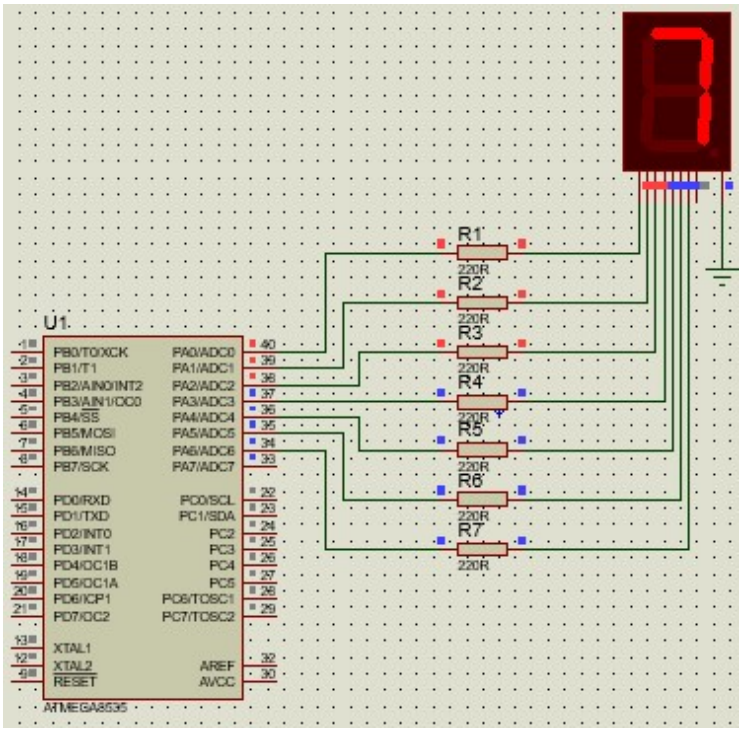
```
.INCLUDE "M8535DEF.INC"
.ORG 0

RJMP MAIN
MAIN:
LDI R16, LOW(RAMEND)
OUT SPL, R16
LDI R16, HIGH(RAMEND)
OUT SPH, R16

//PORT A SEBAGAI OUTPUT
LDI R20, $FF
OUT DDRA, R20

//INISIALISASI COUNTER
AWAL:
LDI R16,0

//INISIALIASI POINTER Z
LDI ZL, LOW(2*DAT)
LDI ZH, HIGH(2*DAT)

//MULAI
L1:
LPM R20, Z+
OUT PORTA, R20
RCALL DELAY
INC R16
CPI R16, $10
BRLO L1
RJMP AWAL
//SUBRUTIN DELAY 1 DETIK
DELAY:
LDI R21, 11
L21:
LDI R22, 0
L22:
LDI R23, 0
L23:
DEC R23
BRNE L23
DEC R22
BRNE L22
DEC R21
BRNE L21
RET

.ORG 80
DAT:
.DB $3F, $06, $5B, $4F, $66, $6D, $7D, $07, $7F, $6F, $77, $7C, $39, $5E, $79, $71
```

Hasil:



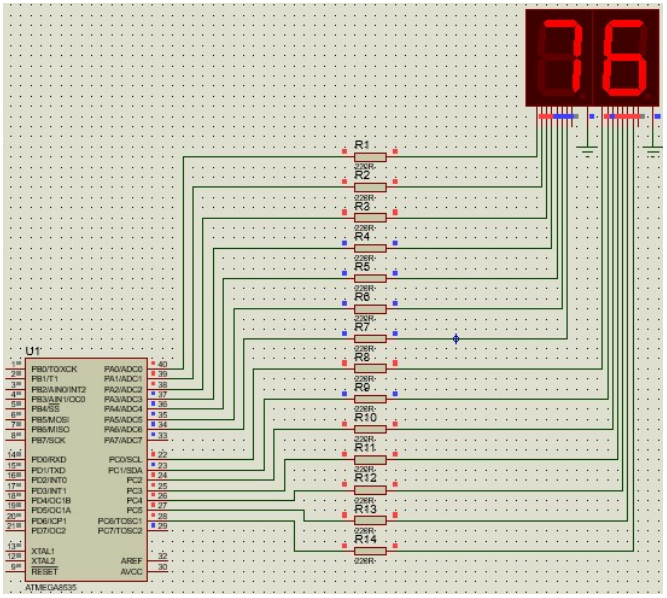
8. Program: Hasil:

```
.INCLUDE "M8535DEF.INC"
.ORG 0

RJMP MAIN
MAIN:
LDI R16, LOW(RAMEND)
OUT SPL, R16
LDI R16, HIGH(RAMEND)
OUT SPH, R16

//PORT A SEBAGAI OUTPUT
LDI R20, $FF
OUT DDRA, R20

//INISIALISASI COUNTER
AWAL:
LDI R16,0

//INISIALIASI POINTER Z
LDI ZL, LOW(2*DAT)
LDI ZH, HIGH(2*DAT)

//MULAI
L1:
LPM R20, Z+
OUT PORTA, R20
RCALL DELAY
INC R16
CPI R16, $8
BRLO L1
RJMP AWAL
//SUBRUTIN DELAY 1 DETIK
DELAY:
LDI R21, 11
L21:
LDI R22, 0
L22:
LDI R23, 0
L23:
DEC R23
BRNE L23
DEC R22
BRNE L22
DEC R21
BRNE L21
RET

.ORG 80
DAT:
.DB $07, $7D, $6D, $66, $4F, $5B, $06, $3F
```



9. Program: Hasil:

```
.include"m8535def.inc"
.org 0
rjmp main
main:   ldi r20, low(ramend)
        out spl, r20
        ldi r20, high(ramend)
        out sph, r20
        ldi r20,0x0ff
        out ddrc,r20
        ldi r20,0x0ff
        out ddra,r20
        ldi zl,low(kar*2)
        ldi zh,high(kar*2)
itu:    ldi r25,0
ini:    mov zl,r25
        lpm r17,z
        out porta,r17
        rcall tunda
        ldi r20,0
sini:   mov zl,r20
        lpm r24,z
        out portc,r24
        rcall tunda
        inc r20
        cpi r20,8
        brne sini
        inc r25
        cpi r25,8
        brne ini
        RJMP itu
akhir:  rjmp akhir
```



```
tunda : ldi r21,3
l21:    ldi r22,0
l22:    ldi r23,0
l23:    dec r23
        brne l23
        dec r22
        brne l22
        dec r21
        brne l21
    ret
;=================================
.org $80
KAR :
.db $3f,$06,$5b,$4f,$66,$6d,$7d,$07
```

V. ANALISIS

1. Pada program ini, perlu dibuat terlebih dahulu rangkaian hardware di simulator Proteus. Digunakan Atmega8535 serta LED dan resistor 330 ohm masing-masing 5 buah. Port yang akan digunakan sebagai port output untuk LED ini ialah port A, sehingga port A perlu dihubungkan ke anoda dari LED. Pada sisi program, port A akan diatur sebagai output dengan mengisi nilai DDRA dengan logika 1 atau FF.

   Setelah itu, R20 akan bertindak sebagai counter untuk membatasi nilai biner sampai 5-bit, dimana nilai maksimal dari 5-bit adalah 32. Nilai R20 ini akan diatur untuk pertama kali yaitu 0, kemudian diisikan menuju PORTA dengan instruksi OUT, nilai dari R20 ini akan terus meningkat dengan instruksi INC. Ketika nilai R20 ini masih dibawah 32, maka program akan terus mengulang ke label MULAI untuk menampilkan nilai R20 yang meningkat tadi ke port A. Akan tetapi ketika nilainya sudah menyentuh 32, maka program akan langsung menuju label AKHIR untuk mengakhiri program.

2. Program ini persis sama seperti sebelumnya, yang menjadi perbedaan ialah bahwa pada program ini, up-counter akan dilakukan secara berulang atau terus menerus. Hal ini dapat dilakukan dengan menghapus label AKHIR, serta diganti oleh perintah atau instruksi RJMP AWAL, sehingga ketika R20 telah menyentuh angka 32, program akan loncat ke label AWAL yang akan mereset R20, dan program mengulang kembali.

3. Program ini persis juga seperti sebelumnya, hanya saja jumlah bit nya menjadi 8-bit, sehingga perlu ditambahkan 3 LED pada setup Proteus. Selain itu, yang perlu diubah dari sisi program adalah batas perbandingan untuk R20. Dikarenakan 8-bit, maka nilai maksimalnya adalah 256, sehingga R20 perlu dibandingkan dengan 256 atau 100. Hal ini dapat disiasati dengan membandingkan R20 dengan 00, karena saat nilainya FF apabila ditingkatkan lagi akan menjadi 00 dan 1 akan disimpan di flag Carry.
   Selain itu, subrutin DELAY perlu sedikit perubahan dengan mengubah isi register R21 dengan nilai yang memungkinkan untuk penundaan selama 1/4 detik. Penulis mengisi R21 dengan 8, serta mengatur clock pada Atmega8535 di simulator proteus di angka 4MHz.

4. Pada soal ini, perlu dibuat dahulu rangkaian simulasi pada Proteus dengan menggunakan 7-segment common anode, dimana port A pada Atmega8535 tetap bertindak sebagai port output yang akan terhubung ke port input a – g pada 7-segment common anode. Pada sisi program, perlu disiapkan terlebih dahulu kombinasi

heksadesimal untuk menampilkan deretan angka desimal apda 7-segment CA, dimana 7-segment akan bernilai high bila diberi logika 0 dan low ketika diberi logika 1.

Sebagai contoh, untuk menampilkan angka 0 pada 7-segment, pin h dan g pada 7-segment akan bernilai 1, dan sisanya bernilai 0, sehingga apabila disusun dalam barisan heksadesimal akan menjadi 11000000 ($C0). Semua nilai kombinasi heksadesimal ini disimpan pada program area degan label DAT, mulai dari angka 0 hingga 9 (Modulo 10). Pointer Z akan digunakan sebagai penunjuk ke almaat program area, serta digunakan instruksi LPM untuk mengambil data dari program area. Register R20 akan bertindak untuk menyimpan nilai dari program area dan akan diisikan ke port A dengan perintah OUT, maka R20 ini nilainya akan meningkat terus menerus.

Register R16 akan bertindak sebagai counter untuk membatasi perubahan nilai pada port A sebanyak 10 kali, dimana perlu dibandingkan ketika nilai dari R20 belum menyentuh $A, maka program akan mengulang dengan loncat ke label L1, selain itu bila nilainya telah menyentuh $A, maka program akan kembali ke label AWAL untuk melakukan inisialisasi ulang mereset alamat pada pointer Z. Subrutin DELAY berfungsi untuk melakukan penundaan waktu selama 1 detik.

5. Program pada soal ini sama seperti pada soal nomor 4 sebelumnya, hanya saja 7-segment yang digunakan bertipe common cathode, sehingga perlu ada perubahan di sisi data yang dimasukkan pada program area. Seperti contoh, untuk menampilkan angka 0, apabila pada soal sebelumnya merupakan heksadesimal $C0, maka pada soal ini merupakan kebalikannya yaitu $3F, dikarenakan pada 7-segment bertipe common cathode, tiap segmen akan high ketika diberi logika 1, sebaliknya akan low ketika diberi logika 0.

6. Program pada soal ini sama seperti pada soal nomor 4 sebelumnya, hanya saja 7-segment akan menampilkan nilai heksadesimal dari F sampai 0, sehingga perlu ada perubahan di sisi data yang dimasukkan pada program area. Data pada program area disusun dari heksadesimal huruf F sampai ke 0 seperti ditunjukkan tabel berikut.

7SEG	P7	P6	P5	P4	P3	P2	P1	P0	HEX
	H	G	F	E	D	C	B	A	
0	1	1	0	0	0	0	0	0	C0
1	1	1	1	1	1	0	0	1	F9
2	1	0	1	0	0	1	0	0	A4
3	1	0	1	1	0	0	0	0	B0
4	1	0	0	1	1	0	0	1	99
5	1	0	0	1	0	0	1	0	92
6	1	0	0	0	0	0	1	0	82
7	1	1	1	1	1	0	0	0	F8
8	1	0	0	0	0	0	0	0	80
9	1	0	0	1	0	0	0	0	90
A	1	0	0	0	1	0	0	0	88
B	1	0	0	0	0	0	1	1	83
C	1	1	0	0	0	1	1	0	C6

D	1	0	1	0	0	0	0	1	A1
E	1	0	0	0	0	1	1	0	86
F	1	0	0	0	1	1	1	0	8E

7. Program pada soal ini sama seperti pada soal nomor 5 sebelumnya, hanya saja 7-segment akan menampilkan nilai heksadesimal dari 0 sampai F, sehingga perlu ada perubahan di sisi data yang dimasukkan pada program area. Data pada program area disusun dari heksadesimal huruf 0 sampai ke F pada 7-segment CC seperti ditunjukkan tabel berikut.

7SEG	P7	P6	P5	P4	P3	P2	P1	P0	HEX
	H	G	F	E	D	C	B	A	
0	0	0	1	1	1	1	1	1	3F
1	0	0	0	0	0	1	1	0	6
2	0	1	0	1	1	0	1	1	5B
3	0	1	0	0	1	1	1	1	4F
4	0	1	1	0	0	1	1	0	66
5	0	1	1	0	1	1	0	1	6D
6	0	1	1	1	1	1	0	1	7D
7	0	0	0	0	0	1	1	1	7
8	0	1	1	1	1	1	1	1	7F
9	0	1	1	0	1	1	1	1	6F
A	0	1	1	1	0	1	1	1	77
B	0	1	1	1	1	1	0	0	7C
C	0	0	1	1	1	0	0	1	39
D	0	1	0	1	1	1	1	0	5E
E	0	1	1	1	1	0	0	1	79
F	0	1	1	1	0	0	0	1	71

8. Program pada soal ini sama seperti pada soal nomor 7 sebelumnya, hanya saja 7-segment akan menampilkan nilai oktal dari 7 sampai 0, sehingga perlu ada perubahan di sisi data yang dimasukkan pada program area.

9. Pada program ini, port A akan bertindak sebagai output delapanan dan port c akan bertindak sebagai output satuan. Output pada port C akan terus meningkat dengan instruksi INC dimana dibaut perbandingan, ketika nilai dari port C telah menyentuh angka 7, maka port A akan menampilkan angka selanjutnya. Digit angka oktal berbentuk heksadesimal disimpan di program area dengan label KAR serta untuk dapat melakukan penunjukkan atau pointing pada alamatnya, digunakan pointer Z.

VI. KESIMPULAN

Dari praktikum ini, dapat diambil kesimpulan bahwa untuk dapat mengatur fungsi dari port baik untuk input maupun output, digunakan instruksi OUT DDRX, dimana X ini adalah abjad port yang akan diatur, logika 1 untuk output dan logika 0 untuk input. Selain itu juga,

untuk dapat mengoperasikan 7-segment baik common anode maupun common cathode, perlu diatur satu persatu digit yang akan ditampilkan dengan melakukan konversi ke bentuk heksadesimal untuk masing-masing segmen yang akan dinyalakan. Pointer Z memiliki peranan penting sebagai penunjuk menuju alamat program memori sebagai tempat disimpannya data yang akan ditampilkan.

VII. DAFTAR PUSTAKA

Yusrizal. 2016. *Mikrokontroler Atmega8535.* Yusrizal Weblog. (https://yusrizalandeslubs.wordpress.com/dasar-elektronika). Diakses 5 Oktober 2020