DIKTAT

SISTEM MIKROPROSESOR

DISUSUN
OLEH:

IRWAN KURNIAWAN, ST

POLITEKNIK JAMBI
2012

BAB I
Pengenalan Mikroprosesor

1. Perkembangan Mikroprosesor

Mikroprosessor dapat dikelompokkan menurut teknologi yang dipergunakan, menurut jumlah bit data, menurut struktur atau menurut kemampuan/karakteristik mikroprosessor dan menurut fungsi dari mikroprosessor itu sendiri. Berdasarkan jumlah bit data (Word Size) pada waktu ini telah terdapat banyak macam mirkoprosessor, mulai dari mikroprosessor 1bit, 4bit, 8 bit. 16 bit, 32 bit dan 64 bit. Selain itu mirkoprosessor dapat pula dikelompokan menurut fungsi dan integrasisnya adalah sebagai berikut:

- Mikroprosessor Monolotik (Chip Tunggal)
- Mikrokomputer chip tunggal (One – chip Mircocomputer)
- Mikroprosessor (atau Prosessor) Bit – Slice

1.1 Mikroprosessor 4 bit

Mikroprosessor intel 4004 merupakan mirkroprosessor pertama yang diperkenalkan pada tahun 1971. Mikroprosessor tersebut memperguanakan teknologi PMOS. Selain itu telah dikembangkan pula beberapa buah mirkoprosessor 4 dengan teknologi yang sama dan dengan teknologi yang lain. Alasan disebut mikroprosessor 4 bit adalah karena mikroprosessor ini hanya mampuh mengolah data 4 bit.

1.2 Mikroprosesor 8 bit

Mikroprosessor 8 bit merupakan mikroprosessor standard yang memperguanakan teknologi PMOS atau NMOS dan umumnya berupa mikroprosessor monolitik. Mikroprosessor ini diperkenalkan pada tahun 1975. sifat khusus mirkoprosessor ini adalah : Harga Murah, dengan kepadatan komponen sangat tinggi, daya yang cukup rendah tetapi dengan kecepatan yang relatif rendah pula. Beberapa contoh diantaranya adalah : Motorola 6800, 6809, Intel 8080, 8085 dan Zilog Z-80 yang semuanya mempergunakan Teknologi NMOS.

1.3 Mikroprosesor 16 bit

Pada mikroprosesor 16 bit, bagian ALU (Arithmetic Logic Unit), register dalam dan sebagian besar intruksinya dirancang untuk dapat berkeja dengan binary words sebesar 16. mikroprosessor ini makin populer dan terlihat mulai menggeser mikroprosessor 8 bit dalam kedudukannya sebagai mikroprosessor standard. Beberapa jenis mikroprosessor 16 bit yang cukup dikenal adalah : 8086, 8088, 80186, 80188, 80286, 80288 (intel), Motorola MC68000, Zilog Z8000 dan Texas Instruments 9900. Mikroprosessor 8086 mempunyai bus data 16 bit, sehingga dapat menulis atau membaca data ke/dari memori atau port input/output sebesar 16 bit atau 8 bit setiap saat, mikroprosessor ini mempunyai bus alamat 20 bit, sehingga dapat mengalamati sebanyak 2^{20} = 1,048,57626 lokasi memori.

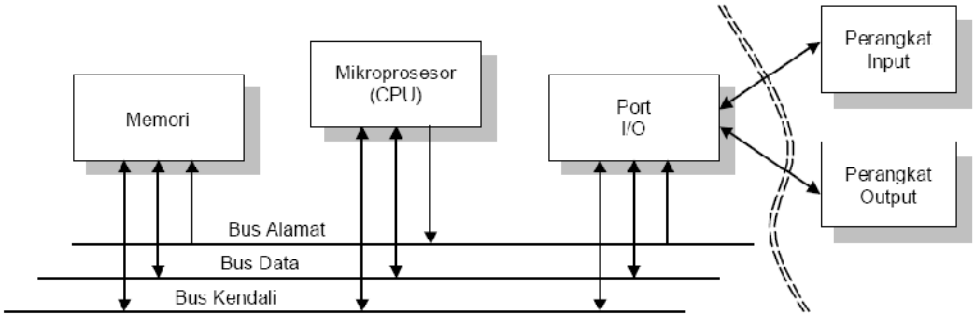
1.4 Mikroprosesor 32 bit

Dengan Perkembangan yang pesat dalam teknologi rangkaian terintegrasi jenis VLSI, maka teknologi mikroprosessor berkembang pula dalam arah jumlah bit data yang makin meningkat disamping peningkatan dalam segi kemampuan lainya. Pada sekitar tahun 1984/1985 telah diperkenalkan mikroprosessor 32 bit, yang dapar bekerja dengan memori dalam orde Mega byte. Contohnya NS32032, Motorola 68020, Western Electric WE32000.

1.5 Mikroprosesor 64 bit

Mikroprosessor ini diperkenalkan pada tahun 2002 pertamakali Oleh AMD. Dengan menggunakan mikroprosessor 64 bit dapat bekerja dengan memori dalam orde giga byte bahkan sampai tera byte. Mikroprosessor ini dikenal dengan sebutan prosessor64. Kemampuan mikroprosessor 64 bit dapat mengolah gambar dalam bentuk tiga dimensi.

2. Pengertian Umum



Gambar 1.1 Diagram Skematik Arsitektur Komputer

2.1. Mikroprosessor

 Mikroprosessor adalah Central Processing Unit dari sebuah computer, tanpa memori, I/O unit, dan peripheral yang dibutuhkan oleh suatu system lengkap. Sebagai contoh 8088 dan 80x86 adalah suatu mikroprosessor yang membutuhkan perangkat pendukung berupa RAM, ROM dan I/O unit.

 Bila sebuah mikroprosessor dikombinasikan (CPU) dengan memori dan I/O unit dapat juga dilakukan dalam level chip yang menghasilkan single chip mikrokomputer (SCM) untuk membedakannya dengan mikrokomputer. Untuk selanjutnya karena fungsinya SCM dapat disebut mikrokontroller.

- Pada sistem komputer mikroprosesor memiliki 3 tugas utama:
  1. Mentransfer data antara mikrorprosesor sendiri dan memori atau sistem I/O
  2. Menjalankan operasi aritmatika atau operasi logika
  3. Menentukan aliran program melelaui keputusan sederhana

2.2. CPU (Central Prossesing Unit)

 Unit Pengelola pusat (CPU) terdiri atas dua bagian, yaitu unit pengendali (Control Unit) serta aritmatika dan logika (ALU). Fungsi utama unit pengendali adalah mengambil, mengkode, dan melaksanakan urutan intruksi sebuah program yang tersimpan dalam memori. Unit pengendali mengatur urutan operasi seluruh system. Unit juga menghasilkan dan mengatur sinyal pengendali yang diperlukan untuk menyerempakkan operasi, juga aliran dan intruksi program.

 Unit Pengontrol mengendalikan aliran informasi pada bus data dan bus alamat, dilanjutkan dengan menafsirkan dan mengatur sinyal yang terdapat pada bus pengendali. Unit aritmatika dan logika melaksanakan pengelohan data secara aritmatika (aljabar) dan secara logika (fungsi OR, NOT, AND, dan XOR).

2.3. Bus

Bus adalah kumpulan jalur yang menghubungkan ketiga komponen di atas. Bus dapat dianalogikan sebagai jalan umum di muka rumah kita yang dapat kita lewati jika hendak menuju rumah tetangga, kantor, dsb. Bedanya, di jalan umum pada suatu waktu bisa terdapat banyak orang atau kendaraan yang melewatinya; sedangkan untuk bus, pada suatu saat hanya bisa ada satu keadaan (biner) untuk setiap jalurnya. Dengan kata lain, ada banyak komponen yang terhubung ke bus, tapi hanya sebuah komponen yang akan mengisi bus tersebut pada suatu saat. Bus dalam sistem komputer dibagi menjadi 3 kelompok:

Bus alamat (a*ddress bus*), yang digunakan oleh mikroprosesor untuk mengirim informasi alamat memori atau port I/O yang akan dihubungi olehnya. Ukuran bus alamat menentukan berapa kapasitas memori yang ada, misalnya ukuran bus alamat 16 bit (16 jalur alamat) akan mampu mengalamati 2^{16} atau 65536 (64 kb) lokasi memori. Perhatikan arah panah ke dan dari bus alamat pada Gambar 1.1

Bus data (d*ata bus*), yang digunakan untuk lewatnya data dari dan ke masing-masing komponen di atas. Bus data mempunyai ukuran tertentu misalnya 8, 16, atau 32 jalur. Ukuran ini tidak harus sama dengan ukuran data pada setiap lokasi memori. Misalnya apabila berukuran memori adalah 8 bit, maka dengan bus data 32 bit akan dapat memindahkan 4 data (menulis/membaca 4 lokasi memori) sekaligus.

Bus kendali (*control bus*), yang berisi jalur-jalur untuk keperluan pengiriman sinyal kendali antar komponen, misalnya sinyal yang menandakan isyarat untuk membaca, atau menulis, pemilihan memori atau port, interupsi, dll. Isyarat-isyarat ini yang kemudian menentukan aksi apa yang harus dilakukan oleh masing-masing komponen.

2.4. Memori

- Suatu sistem mikroprossesor/mikrokontroller maupun komputer memerlukan memori untuk tempat menyimpan program/data.
- Ada beberapa tingkatan memori, diantaranya adalah register internal, memori utama, dan memori massal, register internal adalah memori di dalam ALU. Waktu akses register sangat cepat umumnya kurang dari 100ns. Memori utama adalah memori suatu system. Ukurannya berkisar antara 4 Kbyte sampai 64Kbyte. Waktu akses lebih lambat dibandingkan register internal, yaitu antara 200 sampai 1000ns. Memori massal dipakai untuk menyimpan berkapasitas tinggi, biasanya berbentuk disket, pita magnetic, atauk kaset.

2.5. RAM

- RAM (Random Acces Memory) adalah memori yang dapat dibaca atau ditulis. Data dalam RAM akan terhapus (bersifat Volatile) bila catu daya dimatikan. Oleh karena sifat RAM yang volalite ini, maka program mikroprossesor/mikrokontroller tidak tersimpan dalam RAM. RAM hanya digunakan untuk menyimpan data sementara, yaitu data yang tidak begitu vital bila hilang akibat aliran daya terputus.
- Ada dua teknologi yang dipakai untuk membuat RAM, yaitu RAM static dan RAM dinamik. Dalam RAM static, satu bit infromasi disimpan dalam sebuah flip-flop. RAM static tidak memerlukan penyegar dan penangannanya tidak terlalu rumit. Isi dari RAM tetap tersimpan selama daya diberikan. Dua contoh RAM static adalah

6116 dan 6264 yang masing-masing berkapasitas 2 kb dan 8 kb.

 RAM dinamik menyimpan bit informasi sebagai muatan. Sel memori elemeter dibuat dari kapasitansi gerbang substrat transistor MOS. Keuntungan RAM dinamik adalah sel-sel memori lebih kecil sehingga memerlukan tempat yang lebih sempit, sehingga RAM dinamik menjadi lebih kecil dibandingkan dengan RAM static. Contoh RAM dinamik adalah 4116 yang berkapasitas 16384 x 1 bit.

 Kerugian menggunakan RAM dinamik adalah bertambahnya kerumitan pada papan memori, karena diperlukan rangkaian untuk proses penyegaran (refresh). Proses penyegaran untuk kapasitor ini dilakukan setiap 1 atau 2 mili detik.

2.6. ROM

 ROM (Read Only Memory) merupakan memori yang hanya dapat dibaca. Data dalam ROM tidak akan terhapus meskipun catu daya diputuskan (bersifat nonvolalite) karena sifatnya yang demikian, ROM digunakan untuk menyimpan program.

 Ada beberapa tipe ROM, diantaranya adalah ROM, PROM, EPROM, dan EEPROM, ROM adalah memori yang sudah deprogram oleh pabrik, PROM (Programable Read Only Memori) dapat diprogram oleh pemakai, tetapi hanya sekali program saja atau yang disebut OTP (One Time Programmable), alternative lain adalah menggunakan EPROM (Eraseable Programmable Read Only Memory), yaitu PROM yang dapat diprogram ulang. Isi EPROM dihapus dengan menggunakan sinar Ultra Violer. Isi EPROM setelah dihapus akan berlogika 1. pemograman EPROM adalah mengubah logika 1 menjadi 0. EEPROM (Electrical Eraseable Programmable Read Only Memory) yaitu PROM yang dapat diprogram ulang. Isi program dihapus menggunakan suatu tegangan listrik.

2.7. Input/Output

 Untuk melakukan hubungan dengan peranti diluar sistem, dibutuhkan alat I/O (input/output). Sesuai dengan namanya, alat I/O dapat menerima data dari mikroprosessor/mikrokontroller.

 Ada dua macam perantara I/O yang dipakai, yaitu peranti untuk hubungan serial (UART) dan piranti untuk hubungan parallel (PIO). Pada mikrokontroller MCS`51 kedua macam I/O tersebut sudah tersedia didalamnya.

 UART adalah perantara serial universal. UART (Universal Asynchronous receiver transmitter) yang merupakan pengirim penerima tak serempak universal. Kerja UART adalah mengubah masukan parallel menjadi keluaran serial. UART adalah mengubah masukan serial menjadi keluaran parallel dan mengubah masukan parallel menjadi serial.

 PIO (Paralel Input Output) merupakan perantara untuk hubungan data dalam format parallel. PIO adalah alat yang dapat deprogram dan menyediakan perantara masukan dan keluaran dasar untuk data parallel 8 bit.

BAB II

Sistem Bilangan dan Gerbang Logika

Sistem bilangan desimal atau denary, yaitu sistem bilangan dengan basis 10, yang mempunyai 10 buah simbol yaitu 0,1,2,...,9. tetapi sistem ini tidak selalu merupakan pilihan terbaik untuk setiap aplikasi. Sistem biner yang lebih sederhana pilihan lebih cocok digunakan pada elektronika digital. Sistem biner merupakan sistem bilangan berbasis 2 dan hanya mempunyai dua simbol yaitu 1 dan 0. sistem lain yang sering digunakan adalah sistem bilangan dengan basis 8 atau oktal dan sistem bilangan dengan basis 16 atau heksadesimal.

1. Sistem Bilangan Desimal dan Biner

Dalam sistem denary, yang lebih dikenal dengan sistem bilangan desimal, nilai yang terdapat pada kolom ketiga pada tabel dibawah yaitu A disebut satuan, kolom kedua yaitu B disebut puluhan, C disebut ratusan dan seterusnya. Kolom A,B,C menunjukkan kenaikan pada eksponen dengan basis 10 yaitu $10^0 = 1$, $10^1 = 10$, $10^2 = 100$.

Tabel 2.1 Tabel Ekponensial Bilangan Desimal

C	***B***	***A***
10^2	10^1	10^0
Ratusan	Puluhan	Satuan

Setiap kolom pada sistem bilangan biner, yaitu sistem bilangan dengan basis 2, menunjukkan eksponen dengan basis 2 yaitu $2^0 = 1$, $2^1 = 2$, $2^2 = 4$ dan seterusnya. Setiap digit biner disebut bit, bit paling kanan disebut Least Significant Bit (LSB) dan bit paling kiri disebut Most Significant Bit (MSB).

Tabel. 2.2 Bilangan Biner

Desimal	Biner		
	C (4)	B (2)	A (1)
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

a. Konversi Bilangan Desimal ke Bilangan Biner

Konversi bilangan desimal ke biner dapat dilakukan dengan kombinasi intuis dan metode coba-coba (trial and error). Bilangan desimal yang diketahui dipisah-pisahkan kedalam sejumlah bilangan pangkat dengan basis 2.

Tabel 2.3 Konversi Bilangan Desimal ke Biner

Bilangan Desimal	Kolom Biner						***Bilangan Biner***
	2^5	2^4	2^3	2^2	2^1	2^0	
15	0	0	1	1	1	1	001111
22	0	1	0	1	1	0	010110
45	1	0	1	1	0	1	101101
52	1	1	0	1	0	0	110100

Sebagai Contoh 22_{10} nilainya lebih kecil dari 2^5(=32), maka bit 0 akan ditempatkan pada kolom tersebut. Tetapi 22 lebih besar dari 2^4 (=16), sehingga bit 1 ditempatkan pada kolom tersebut. Sisanya adalah 22 – 16 = 6 yang lebih besar dari 2^2 (= 4) sehingga bit 1 ditempatkan pada kolom tersebut, sehingga sisanya 6 – 4 = 2 sisa ini akan menghasilkan bit 1 yang harus dipasang pada kolom 2^1 (= 2) dan bit 0 ditempatkan pada kolom 2^0 (= 1) sehingga bilangan $22_{10} = 010110_2$.

Cara lain adalah dengan pembagian. Bilangan desimal yang akan diubah secara berturut-turut dibagi 2, dengan memperhatikan sisa pembagiannya. Sisa pembagian akan bernilai 0 atau 1 yang akan membentuk bilangan biner dengan sisa yang terakhir menunjukkan MSBnya, sebagai Contoh untuk mengubah 52_{10} menjadi bilangan biner, diperlukan langkah-langkah berikut :

52/2 = 26 sisa 0, LSB
26/2 = 13 sisa 0
13/2 = 6 sisa 1
6/2 = 3 sisa 0
3/2 = 1 sisa 1
1/2 = 0 sisa 1, MSB

b. Konversi Bilangan Biner ke Bilangan Desimal

Untuk mengubah bilangan biner ke dalam bilangan desimal yaitu dengan menggunakan subskrip contoh mengubah bilangan biner 1110_2 ke dalam bilangan desimal.

$1110_2 = 1 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^1$

$= 8 + 4 + 2 + 0 = 14_{10}$

2. Bilangan Heksadesimal

Bilangan heksadesimal, sering disingkat dengan hex adalah bilangan dengan basis 16 dan mempunyai 16 simbol yang berbeda.

Tabel 2.4 Bilangan Hexadesimal ke Desimal

Heksadesimal	***Desimal***
0	0
1	1
2	2
3	3

4	4
5	5
6	6
7	7
8	8
9	9
A	10
B	11
C	12
D	13
E	14
F	15

Bilangan yang lebih besar dari 15_{10} memerlukan lebih dari satu digit hex. Kolom heksadesimal menunjukkan eksponen dengan basis 16, yaitu 16^0 = 1, 16^1 = 16, 16^2 = 256 dan seterusnya. Sebagai contoh $152B_{16}$ = ...$_{10}$

$152B_{16} = (1 \times 16^3) + (5 \times 16^2) + (2 \times 16^1) + (11 \times 16^0)$
$= 1 \times 4096 + 5 \times 256 + 2 \times 16 + 11 \times 1$
$= 4096 + 1280 + 32 + 11$
$= 5419_{10}$

Sebaliknya untuk mengubah bilangan desimal menjadi bilangan heksadesimal, dapat dilakukan dengan cara membagi bilangan desimal tersebut dengan 16. sebagai contoh untuk mengubah bilangan 3409_{10} menjadi bilangan heksadesimal dilakukan dengan langkah sebagai berikut :

3409/16 = 213 sisa $1_{10} = 1_{16}$ LSB

213/16 = 13 sisa $5_{10} = 5_{16}$

13/6 = 0 sisa $13_{10} = D_{16}$, MSB

sehingga $3409_{10} = D51_{16}$

Konversi Bilangan Heksadesimal ke Biner

Setiap digit pada bilangan heksadesimal dapat disajikan dengan empat buah bit seperti terlihat pada tabel dibawah ini:

Tabel 2.5 Konversi bilangan heksadesimal ke bilangan biner

Heksadesimal	***Biner***
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100

D	1101
E	1110
F	1111

untuk mengubah bilangan heksadesimal menjadi bilangan biner, setiap digit dari bilangan heksadesimal diubah secara terpisah ke dalam empat bit bilangan biner. Sebagai contoh
$2A5C_{16}$ dapat diubah ke bilangan biner sebagai berikut :

2_{16} =0010, MSB
A_{16} = 1010
5_{16} = 0101
C_{16} = 1100, LSB

sehingga bilangan heksadesimal 2A5C akan diubah menjadi bilangan biner 0010 1010 0101 1100.

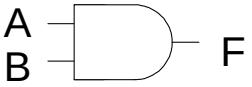
Sebaliknya bilangan biner dapat diubah menjadi bilangan heksadesimal dengan cara mengelompokkan setiap empat digit dari bilangan biner tersebut dimulai dari digit paling kanan. Sebagai contoh 0100111101011100_2 dapat dikelompokan menjadi 0100 1111 0101 1110 sehingga

$0100_2 = 4_{16}$ MSB
$1111_2 = F_{16}$
$0101_2 = 5_{16}$
$1110_2 = E_{16}$ LSB
dengan demikian bilangan $0100111101011100_2 = 4F5E_{16}$

3. Gerbang Logika

Gerbang-gerbang dasar logika merupakan elemen rangkaian digital dan rangkaian digital merupakan kesatuan dari gerbang-gerbang logika dasar yang membentuk fungsi pemrosesan sinyal digital. Gerbang dasar logika terdiri dari 3 gerbang utama, yaitu AND Gate, OR Gate, dan NOT Gate. Gerbang lainnya seperti NAND Gate, NOR Gate, EX-OR Gate dan EX-NOR Gate merupakan kombinasi dari 3 gerbang logika utama tersebut.

a. Gerbang AND
   Gerbang AND merupakan salah satu gerbang logika dasar yang memiliki 2 buah saluran masukan (input) atau lebih dan sebuah saluran keluaran (output). Suatu gerbang AND akan menghasilkan sebuah keluaran biner tergantung dari kondisi masukan dan fungsinya. Prinsip kerja dari gerbang AND adalah kondisi keluaran (output) akan berlogic 1 bila semua saluran masukan (input) berlogic 1. Selain itu output akan berlogic 0. Simbol gerbang logika AND 2 input :



dengan persamaan Boolean fungsi AND adalah F = A.B (dibaca F = A AND B).

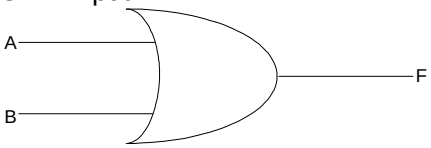
Tabel kebenaran:

input		Output
A	B	F
0	0	0
0	1	0
1	0	0
1	1	1

b. Gerbang OR

Gerbang OR merupakan salah satu gerbang logika dasar yang memiliki 2 buah saluran masukan (input) atau lebih dan sebuah saluran keluaran (output). Berapapun jumlah saluran masukan yang dimiliki oleh sebuah gerbang OR, maka tetap memiliki prinsip kerja yang sama dimana kondisi keluarannya akan berlogic 1 bila salah satu atau semua saluran masukannya berlogic 1. Selain itu output berlogic 0.

Simbol gerbang logika OR 2 input :



c.

dengan persamaan Boolean fungsi OR adalah F = A+B (dibaca F = A OR B).

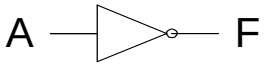
Tabel kebenaran:

input		Output
A	B	F
0	0	0
0	1	1
1	0	1
1	1	1

d. Gerbang NOT

Gerbang NOT sering disebut dengan gerbang inverter. Gerbang ini merupakan gerbang logika yang paling mudah diingat. Gerbang NOT memiliki 1 buah saluran masukan (input) dan 1 buah saluran keluaran (output). Gerbang NOT akan selalu menghasilkan nilai logika yang berlawanan dengan kondisi logika pada saluran masukannya. Bila pada saluran masukannya berlogic 1 maka pada saluran keluarannya akan berlogic 0 dan sebaliknya.

Simbol gerbang logika NOT :

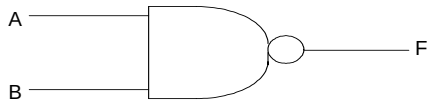


Tabel kebenaran:

Input (A)	Output (F)
0	1
1	0

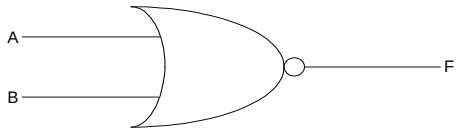
e. Gerbang NAND

Gerbang NAND merupakan kombinasi dari gerbang AND dengan gerbang NOT dimana keluaran gerbang AND dihubungkan ke saluran masukan dari gerbang NOT. Karena keluaran dari gerbang AND di"NOT"kan maka prinsip kerja dari gerbang NAND merupakan kebalikan dari gerbang AND. Outputnya merupakan komplemen atau kebalikan dari gerbang AND, yakni memberikan keadaan level logic 0 pada outputnya jika dan hanya jika keadaan semua inputnya berlogika 1. Simbol gerbang logika NAND 2 input :



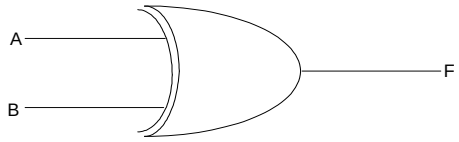
f. Gerbang NOR

Sama halnya dengan NAND Gate, gerbang NOR merupakan kombinasi dari gerbang OR dengan gerbang NOT dimana keluaran gerbang OR dihubungkan ke saluran masukan dari gerbang NOT. Karena keluaran dari gerbang OR di"NOT"kan maka prinsip kerja dari gerbang NOR merupakan kebalikan dari gerbang OR. Outputnya merupakan komplemen atau kebalikan dari gerbang OR, yakni memberikan keadaan level logic 0 pada outputnya jika salah satu atau lebih inputnya berlogika 1. Simbol gerbang logika NOR 2 input :



g. Gerbang EX-OR

EX-OR singkatan dari Exclusive OR dimana jika input berlogic sama maka output akan berlogic 0 dan sebaliknya jika input berlogic beda maka output akan berlogic 1. Simbol gerbang logika EX-OR 2 input :



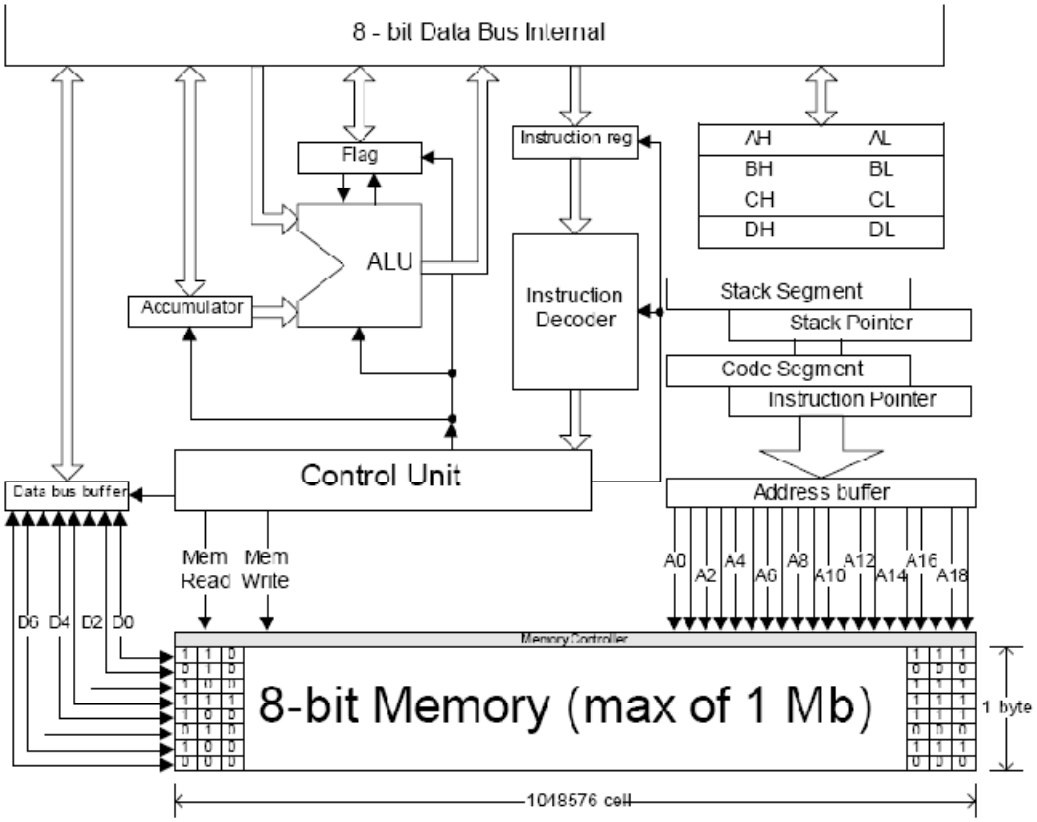
Tabel kebenaran:

input		Output
A	B	F
0	0	0
0	1	1
1	0	1
1	1	0

BAB III

ARSITEKTUR MIKROPROSESOR 8088

1. Bagan Dasar uP 8088



Gambar 3.1 Bagan Dasar uP 8088

Elemen didalam mikroprosesor adalah :

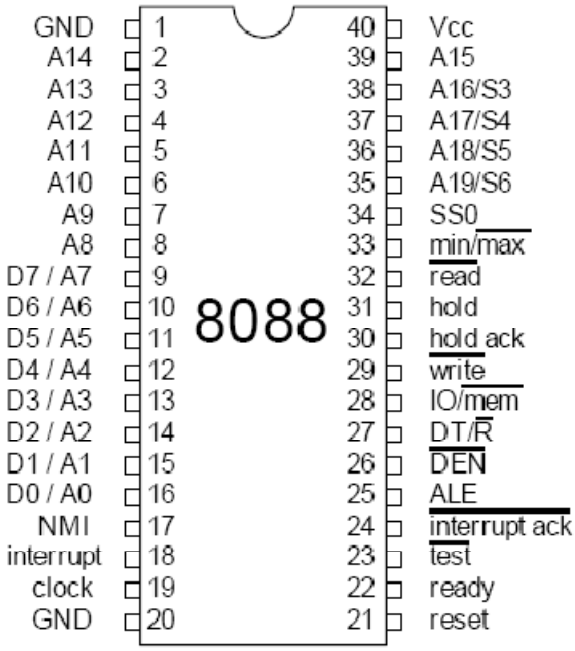
1. CU (**Control Unit**) adalah manajer dari semua unit. CU mengatur keselarasan kerja setiap unit. Apa yang harus dilakukan oleh suatu unit, semuanya diketahui oleh CU dengan bantuan microprogram yang ditanamkan padanya. Pengontrolan oleh CU dilakukan melalui Bus Kontrol (panah dari/ke Control Unit).

2. **Instruction Decoder** bertugas untuk menerjemahkan suatu instruksi dengan cara membandingkannya dengan tabel instruksi yang dimilikinya. Hasil dekoding diberikan ke CU, dan CU akan membangkitkan sinyal-sinyal kontrol yang diperlukan untuk melaksanakan instruksi tersebut.

3. **Register** adalah memori khusus di dalam uP. Untuk mengidentifikasikannya, register memiliki nama khusus yang mencerminkan fungsinya. Berdasarkan isinya, register dapat dibedakan menjadi :

 Register Data memiliki lebar 16 bit namun dapat diakses dalam format 2x8 bit: Accumulator : AX = AH+AL Base Register : BX = BH+BL Counter Register : CX = CH+CL Data Register : DX = DH+DL
m
 Register Alamat memiliki lebar 16 bit : Code Segment : CS, menyimpan alamat segment dari program Instruction Pointer : IP, menyimpan alamat offset dari program Data Segment : DS, menyimpan alamat segment dari data Index Register : BI (Base Index), SI (Source Index), DI (Destination Index), Pointer Register : BP (Base pointer), Stack Segment : SS, menyimpan alamat segment dari stack

2. Pin Out uP 8088 dan fungsinya



Gambar 3.2 Pin-Out mikroprosesor 8088

Hubungan Pin dan fungsinya

AD7- AD0 (D0/A0 – D7/A7)
Address Line (A0 – A7) + Data Line (D0-D7)
Jalu bus alamat/data 8088 terdiri dari bus data alamat yang dimultipleks pada 8088 dan berisi 8 bit paling kanan dari alamat memori atau nomor por I/O jika ALE logika 1 atau ALE logika 0

A15-A8
Bus alamat 8088 menyediakan bit-bit alamat memori paruh atas yand ada selama satu siklus bus.

A19/S6 – A16/S3
Bit bit bus alamat/status di-multipleks untuk memberikan sinyal alamat A19-A16 dan juga bit-bit status S6-S3 (mode maksimum).

$\overline{RD}$ (Read)
Jika Sinyal Baca berupa logika 0, bus data bisa menerima data dari memori atau alat I/O yang dihubungkan ke Sistem.

$\overline{WR}$ (Write)
Jalur write merupakan sinyal yang menunjukkan bahwa 8088 sedanga mengeluarkan data ke memori atau I/O. Selama WR berlogika 0, bus data berisi data yang valid untuk memori dan I/O.

INTR (Interrupt)
Interrupt request digunakan untuk meminta interrupt perangkat keras. Jika INTR dijaga tetap High ketia IF = 1, 8088 memasuki siklus interrupt acknowledge (INTA menjadi Aktif) setelah instruksi pada saat itu telah dijalankan sepenuhnya.

$\overline{INTA}$ (Interrupt ACK)
Sinyal Interrupt Ackhowledge merupakan tanggapan terhapad pit Input INTR. Pin $\overline{INTA}$ biasanya digunakan untuk memasukkan nomor vector interrupt ke bus data sebagai jawaban atas permintaan Interrupt.

ALE
Address Latch Enable menunjukkan bahwa bus alamat/data AD7-AD0 8088 berisikan informasi alamat. Alamat ini bisa merupakan alamat memori atau nomor port I/O.

DT/$\overline{R}$
Sinyal data transmit/receiver menunjukkan bahwa bus data 8088 sedang mengirim data (DT/R = 1) atau menerima data (DT/R =0). Sinyal ini digunakan untuk enable buffer bus data eksternal.

$\overline{DEN}$
Data Bus Enable mengaktifkan buffer bus data eksternal.

IO/$\overline{M}$ (IO/MEM)
Pin IO/$\overline{M}$ 8088 memilih memori ata I/O. Pin ini menunjukkan bahwa bus alamat mikroprosesor berisi alamat memori atau alamat port I/O.

READY
Merupakan Pin Input yang dikendalikan untuk menyisipkan status tunggu ke timing mikroprosesor. Jika Pin Ready di set logika 0, mikroprosesor memasuki status tunggu dan tidak bekerja.

RESET
Input Reset menyebabkan mikroprosesor mereset dirinya sendiri jika Pin ini tetap high (logika 1) selama empat periode clock. Jika 8088 di reset, akan mulai mengeksekusi instruksi pada lokasi memori FFFF0H dan mendisable interrupt berikutnya dengan meng-clearkan bit flag IF

VCC
Input catu daya ini menyediakan sinyal +5,0 Volt, ± 10 % ke mikroprosesor.

GND
Hubungan ground merupakan jalur kembali catu daya. 8088 memiliki 2 pin GND dan keduanya harus dihubungkan ke ground.

MN/$\overline{MX}$

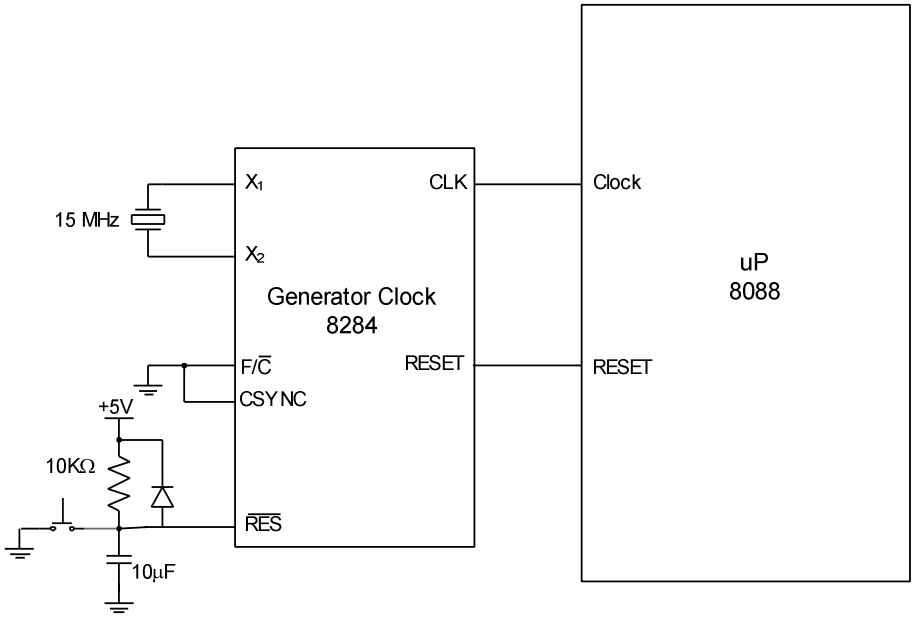
Pin Mode minimum/maksimum memilih operasi mode minimum atau maksimum untuk mikroprosesor. Jika dipilih mode minimum pin MN/MX harus langsung dihubungkan ke +5,0 V.

CLK

Pin clock menyediakan sinyal waktu (timing) dasar ke mikroprosesor. Sinyal clock harus memiliki siklus kerja 33 persen (high selama sepertiga clock dan low selama dua-pertiganya) untuk memberikan timing internal yang sesuai untuk 8088.

Generator Clock 8284

8284 merupakan komponen tambahan mikroprosesor 8088. Tanpa generator clock banyak rangkaian tambahan yang dibutuhkan untuk membangkitkan clock (CLK) pada system yang berbasiskan mikroprosesor 8088. 8284 menyediakan fungsi-fungsi atau sinyal-sinyal dasar berikut ini: pembangkitan clock, sikronisasi RESET. Gambar berikut merupakan generator clock yang dibentuk dari IC 8284.



Gambar 3.3 Rangkaian Clock pada Sistem Berbasis Mikroprosesor 8088

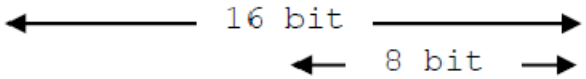
REGISTER

Pada bagan organisasi komputer, bagian memori diletakan terpisah dari mikroprosesor, jika bagan tersebut diimplementasikan, mikroprosesor harus mengakses memori setiap saat. Karena kecepatan memori jauh lebih lambat dari kecepatan mikroprosesor, maka kecepatan kerja mikroprosesor akan dipengaruhi oleh kecepatan memori. Untuk mempercepat pemrosesan data data di dalam mikroprosesor, selain CU dan ALU, mikroprosesor juga akan membutuhkan memori yang memiliki kecepatan yang sama dengan mikroprosesor. Untuk melakukannya, memori ini harus diimplementasikan di dalam mikroprosesor. Memori jenis ini disebut dengan register.

Berdasarkan jenis informasi yang disimpan mikroprosesor dapat dibagi menjadi:

1. Register Data
   Menyimpan data yang diperlukan untuk suatu operasi
   Terdiri dari :
   - AX (Accumulator)
   - BX (Base)
   - CX (Counter)
   - DX (Data)
2. Register Alamat
   Karena jumlah register data terbatas, maka sebagian besar data tetap diletakkan di memori. Untuk dapat mengakses memori, mikroprosesor membutuhkan alamat dari data tersebut pada memori. Alamat data tersebut di simpan di register yang disebut dengan register alamat.
   Terdiri dari :
   - SP (Stack Pointer)
   - BP (Base Pointer)
   - SI (Source Index)
   - DI (Destination Index)
   - DS (Data Segment)
   - ES (Extra Segment)
   - SS (Stack Segment)
   - CS (Code Segment)
   - IP (Instruction Pointer) dan
   - BX (Base) selain dapat digunakan sebagai register data dapat juga digunakan sebagai register alamat
3. Register Flag (flags)
   Digunakan untuk menyimpan status dari hasil operasi ALU
   Terdiri dari :
   OF (Overflow Flag), DF, IF, TF, SF (Sign Flag), ZF (Zero Flag), AF, PF, CF (Carry Flag)
4. Register Instruksi
   digunakan untuk menyimpan instruksi yang sedang dikerjakan.

Lebar semua register uP8088 adalah 16 bit = 2 byte. Sehingga setiap register dapat berharga 0000H s.d. FFFFH atau 216 = 65536 kombinasi harga. Setiap register akan diakses (dibaca/ditulis) dalam format 16 bit tersebut, kecuali register data dapat diakses dalam format 8 bit = 1 byte.



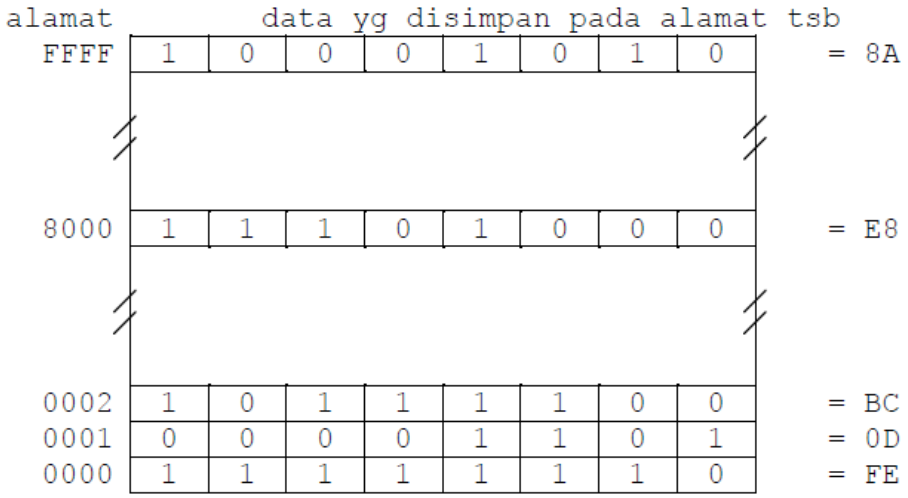
AH	AL
BH	BL
CH	CL
DH	DL

Struktur Memori pada uP8088

Memori pada sistem uP8088 memiliki dua ciri :

1. diakses dgn alamat selebar 16 bit (00000H s.d. FFFFFH) atau 2 byte
2. data yg diakses untuk setiap alamat adalah 8 bit atau 1 byte

Contoh :



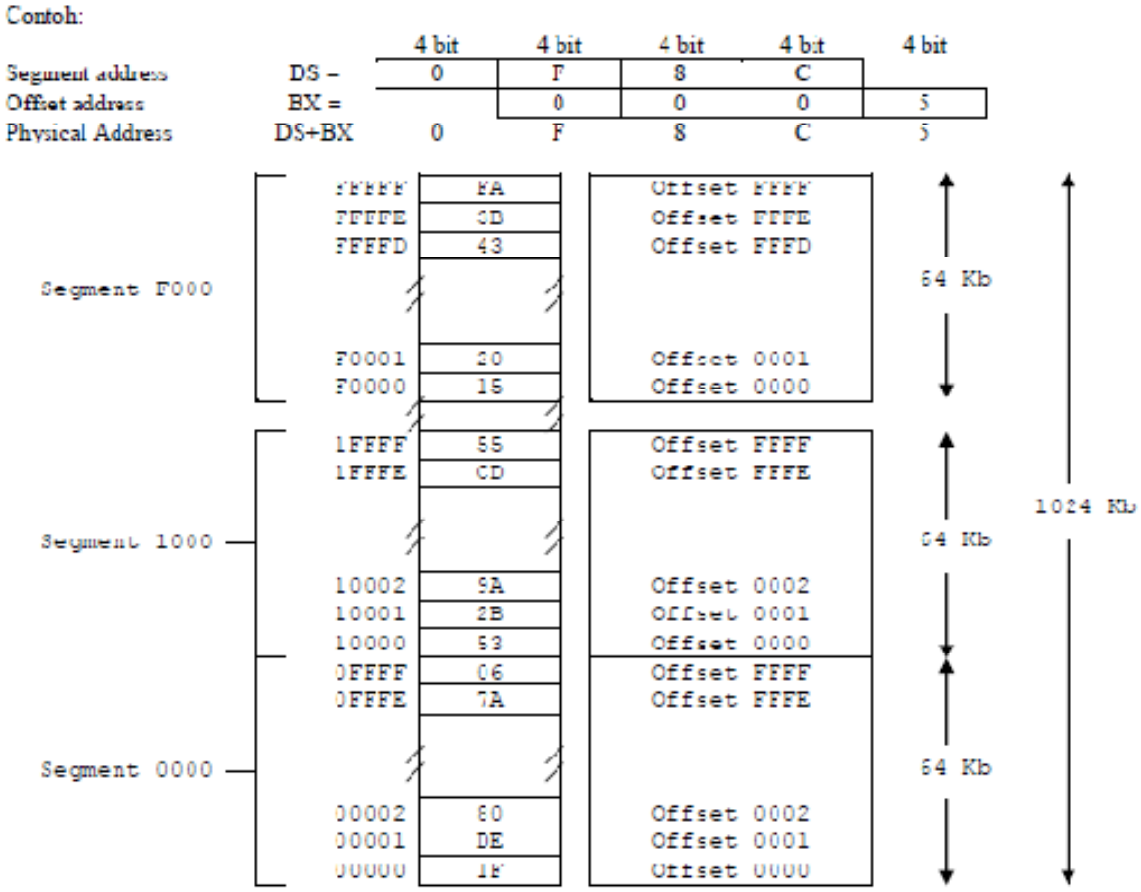
Alamat dari suatu cell memori direpresentasikan dalam format 2 byte (0000H – FFFFH) yg disimpan dalam register alamat (yg lebarnya juga 2 byte). Karena kapasitas register alamat adalah 2 byte, maka jumlah cell memori yg dapat disimpan alamatnya adalah 216 = 65536 cell memori.

Dan karena suatu cell memori menyimpan data 1 byte (00H – FFH) maka suatu register alamat uP8088 dapat mengakses (membaca/menulis) memori berkapasitas 65536 byte = 64 Kbyte.

Microprocessor harus dapat mengakses semua cell memori dari alamat terendah sampai alamat tertinggi. Alamat tersebut akan disimpan didalam register alamat. Secara fisik, uP8088 memiliki 20 buah jalur alamat (A0 – A19) untuk menyediakan informasi alamat selebar 20 bit dimana informasi alamat tersebut dapat berharga 00000H s.d FFFFFH. Ke-20 bit tersebut digunakan untuk mengakses memori dgn kapasitas 220 = 1048576 cell memori.

Namun masalah timbul pada lebar register alamat. uP8088 memiliki register alamat dengan lebar hanya 16 bit dari 20 yang dibutuhkan untuk mengakses semua memori.
Ini berarti jika suatu register alamat menyimpan alamat memori, dia hanya dapat digunakan untuk mengakses alamat 0000 sampai FFFF atau 64 Kb.
Untuk mengatasi kekurangan ini, Intel menggunakan 2 register untuk pengalamatan.

Satu register akan menyimpan alamat segment (suatu area memori seluas 64 Kbyte), dan Satu register akan menyimpan alamat offset (menentukan byte yang mana di dalam segment tersebut yg akan diakses).



uP8088 menyediakan 4 segment untuk menjalankan suatu program.

1. Segment untuk Program (Code/Instruksi)
   CS:IP CS (Code Segment) menyimpan alamat segment (64 Kb of memory) dari program. IP (Instruction Pointer) menyimpan alamat offset dari program yang akan menentukan instruksi mana di dalam 64 Kb tadi yang akan dieksekusi

2. Segment untuk Data
   DS:BX DS (Data Segment) menyimpan alamat segment (64 Kb of memory) dari data. BX (Base Register) menyimpan alamat offset dari data yang akan menentukan data mana di dalam 64 Kb tadi yang akan diambil

3. Segment untuk Stack
   SS:SP SS (Stack Segment) menyimpan alamat segment (64 Kb of memory) dari stack. SP (Stack Pointer) menyimpan alamat offset dari top of the stack yang akan menentukan tumpukan (stack) mana di dalam 64 Kb tadi yang akan diambil (POP)

4. Extra Segment

Instruksi Mesin

Dilihat dari fungsi yg dilakukannya, instruksi mesin dapat dibedakan menjadi
1. **Data transfer** digunakan untuk (1) memindahkan data dari suatu elemen memori ke elemen memory lainnya atau (2) mengisi register data dengan suatu data contoh: MOV, PUSH, POP
2. **Aritmetika dan Logika** digunakan untuk mengkalkulasi suatu perhitungan aritmetika (contoh: ADD, SUB) dan logika (AND, OR, SHL)
3. **Kontrol** digunakan untuk memindahkan kontrol instruksi ke suatu lokasi baru (tidak lagi secara sekuensial) contoh: JMP, JZ

Instruksi MOV

1. Register - Data (contoh: MOV AX,1234 ; mengisi AX dgn data 1234)
2. Register - Register (contoh: MOV AX,BX ; memindahkan isi BX ke AX)
3. Register - Memory (contoh: MOV AX,[BX] ; memindahkan isi memori ke AX, dimana alamat dari datanya ada di BX)
4. Memory - Register (contoh: MOV [BX],AX ; memindahkan isi AX ke memori, dimana datanya akan ditulis di memori pada alamat yg ada di BX)

```
Contoh instruksi MOV untuk transfer antar register dan pengisian langsung
-a
0F6C:0100 mov bx,abcd    → mengisi reg. AX dgn data ABCD_H
0F6C:0103 mov ah,56      → mengisi reg. AH dgn data 56_H
0F6C:0105 mov bl,ah      → mengcopy isi reg. AH ke reg. BL
0F6C:0107 mov ax,bx      → mengcopy isi reg. BX ke reg. AX
0F6C:0109
-r
AX=0000  BX=0000  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0F6C  ES=0F6C  SS=0F6C  CS=0F6C  IP=0100   NV UP EI PL NZ NA PO NC
0F6C:0100 BBCDAB        MOV     BX,ABCD
-t
AX=0000  BX=ABCD  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0F6C  ES=0F6C  SS=0F6C  CS=0F6C  IP=0103   NV UP EI PL NZ NA PO NC
0F6C:0103 B456          MOV     AH,56
```

```
-t
AX=5600  BX=ABCD  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0F6C  ES=0F6C  SS=0F6C  CS=0F6C  IP=0105   NV UP EI PL NZ NA PO NC
0F6C:0105 88E3          MOV     BL,AH
-t
AX=5600  BX=AB56  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0F6C  ES=0F6C  SS=0F6C  CS=0F6C  IP=0107   NV UP EI PL NZ NA PO NC
0F6C:0107 89D8          MOV     AX,BX
-t
AX=AB56  BX=AB56  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0F6C  ES=0F6C  SS=0F6C  CS=0F6C  IP=0109   NV UP EI PL NZ NA PO NC
```

```
catatan:
Instruksi berikut akan menimbulkan Error :
mov ch,5678 :     tidak bisa karena CH = 1 byte dan datanya 2 byte
mov dl,ax   :     tidak bisa karena AX = 2 byte dan DL = 1 byte
mov dx,al   :     tidak bisa karena AL = 1 byte dan DX = 2 byte
```

```
Contoh instruksi MOV untuk :
1. transfer Register ← Memory (membaca data dari memori)
2. transfer Memory ← Register (menulis data ke memori)
-a
0F6C:0100 mov bx,0002    → mengisi reg. BX dgn data 0002H
0F6C:0103 mov ah,[bx]    → membaca memori pada alamat BX sebanyak 1 byte (AH)
0F6C:0105 mov ax,[bx]    → membaca memori pada alamat BX sebanyak 2 byte (AX)
0F6C:0107 mov ax,[bx+1] → membaca memori pada alamat BX+1 sebanyak 2 byte (AX)
0F6C:010A mov [bx],ax    → menulis isi reg. AX ke memori pada alamat BX
0F6C:010C
-d
0F6C:0000  00 01 02 03 04 05 06 07-08 09 0A 0B 0C 0D 0E 0F  ................
0F6C:0010  10 11 12 13 14 15 16 17-18 19 1A 1B 1C 1D 1E 1F  ................
0F6C:0020  20 21 22 23 24 25 26 27-28 29 2A 2B 2C 2D 2E 2F   !"#$%&'()*+,-./
0F6C:0030  30 31 32 33 34 35 36 37-38 39 3A 3B 3C 3D 3E 3F  0123456789:;<=>?
0F6C:0040  40 41 42 43 44 45 46 47-48 49 4A 4B 4C 4D 4E 4F  @ABCDEFGHIJKLMNO
0F6C:0050  50 51 52 53 54 55 56 57-58 59 5A 5B 5C 5D 5E 5F  PQRSTUVWXYZ[\]^_
0F6C:0060  60 61 62 63 64 65 66 67-68 69 6A 6B 6C 6D 6E 6F  `abcdefghijklmno
0F6C:0070  70 71 72 73 74 75 76 77-78 79 7A 7B 7C 7D 7E 7F  pqrstuvwxyz{|}~.
-r
AX=0000  BX=0000  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0F6C  ES=0F6C  SS=0F6C  CS=0F6C  IP=0100   NV UP EI PL NZ NA PO NC
0F6C:0100 BB0200        MOV     BX,0002
-t
AX=0000  BX=0002  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0F6C  ES=0F6C  SS=0F6C  CS=0F6C  IP=0103   NV UP EI PL NZ NA PO NC
0F6C:0103 8A27          MOV     AH,[BX]                         DS:0002=02
-t
AX=0200  BX=0002  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0F6C  ES=0F6C  SS=0F6C  CS=0F6C  IP=0105   NV UP EI PL NZ NA PO NC
0F6C:0105 8B07          MOV     AX,[BX]                         DS:0002=0302
-t
AX=0302  BX=0002  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0F6C  ES=0F6C  SS=0F6C  CS=0F6C  IP=0107   NV UP EI PL NZ NA PO NC
0F6C:0107 8B4701        MOV     AX,[BX+01]                      DS:0003=0403
-t
AX=0403  BX=0002  CX=0000  DX=0000  SP=FFEE  BP=0000  SI=0000  DI=0000
DS=0F6C  ES=0F6C  SS=0F6C  CS=0F6C  IP=010A   NV UP EI PL NZ NA PO NC
0F6C:010A 8907          MOV     [BX],AX                         DS:0002=0302
```

```
catatan:
Instruksi berikut akan menimbulkan Error :
mov [bx],[bx+1]   :     transfer data dari memori ke memori secara langsung
mov [bx],12       :     transfer data langsung ke memori
```

kesimpulan : semua transfer yg melibatkan memori harus via register

```
mov ah,[bl]       :     register alamat harus digunakan dalam format 2 byte
mov [ax],bx       :     reg. AX bukan register alamat
mov [cx],bx       :     reg. CX bukan register alamat
mov [dx],bx       :     reg. DX bukan register alamat
```

DATA TRANSFER

MOV – Move

1. Reg/Mem to/from Reg	1000 10dw	modregr/m	
2. Immediate to Register	1011 wreg	data	data (w=1)
3. Memory to Accumulator	1010 000w	addr-low	addr-high
4. Accumulator to Memory	1010 001w	addr-low	addr-high

```
contoh 1:
MOV AL,BL    = 88D8 (10001000 11011000): d=0,w=0,mod=11,reg=011(BL),r/m=000(AL)
MOV AX,BX    = 89D8 (10001001 11011000): d=0,w=1,mod=11,reg=011(BX),r/m=000(AX)
MOV [BX],AL  = 8807 (10001000 00000111): d=0,w=0,mod=00,reg=000(AL),r/m=111([BX])
MOV [BX],AX  = 8907 (10001001 00000111): d=0,w=1,mod=00,reg=000(AX),r/m=111([BX])
MOV AL,[BX]  = 8A07 (10001010 00000111): d=1,w=0,mod=00,reg=000(AL),r/m=111([BX])
MOV AX,[BX]  = 8B07 (10001011 00000111): d=1,w=1,mod=00,reg=000(AX),r/m=111([BX])
contoh 2:
MOV AX,1234  = B83412(10111000 34H 12H): w=1,reg=000(AX),Low-data=34H,Hi-data=12H
MOV AL,78    = B078   (10110000 78H    ): w=0,reg=000(AL),data=78H
contoh 3:
MOV AX,[1234] = A13412 (10100001 34H 12H)): w=1,addr-low=34H,addr-high=12H
contoh 4:
MOV [1234],AX = A33412 (10100011 34H 12H)): w=1,addr-low=34H,addr-high=12H
```

BAB IV

Multipleksing Alamat/Data, Buffering dan Timing Diagram Sistem Bus

Sebelum 8088 dapat digunakan dengan memori atau antarmuka I/O, bus yang dimultipleknya harus didemultipleks. Bab ini memberikan perincian yang dibutuhkan untuk demultipleks bus dan mengilustrasikan bagaiamana bus dibuffer untuk system yang sangat besar.

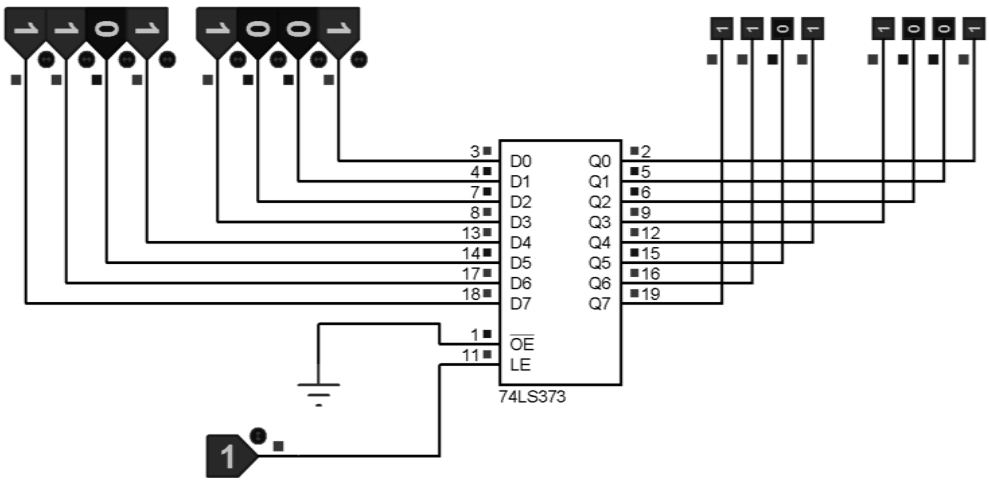
SISTEM BUS

- **Demultipleks Bus Alamat A19-A0 (20 bit)**

Bus alamat/data pada 8088 dimultipleks (dipakai bersama) untuk memperkecil jumlah pin yang dibutuhkan IC mikroprosesor 8088. Sayangnya, hal ini membebani perancang dengan tugas untuk menyaring atau demultipleks informasi dari pin yang dimultpleks ini. Memori dan I/O mensyaratkan bahwa alamat tetap valid dan stabil selama siklus baca (read) atau tulis (write). Jika bus dimultipleks, tidak dapat diketahui alamat memori atau alamat I/O yang berada pada bus, yang bisa menyebabkan pembacaan dan penulisan data dilakukan pada alamat yang salah.

Semua sistem komputer memiliki tiga bus: (1) bus alamat yang memberikan alamat memori dan nomor port (alamat) I/O ke memori dan I/O, (2) bus data yang mentransfer data antara mikroprosesor dan memori atau I/O, (3) bus kontrol yang menyediakan sinyal kontrol ke memori dan I/O. Bus-bus ini harus ada untuk pengaturan antarmuka (interface) ke memori dan I/O.

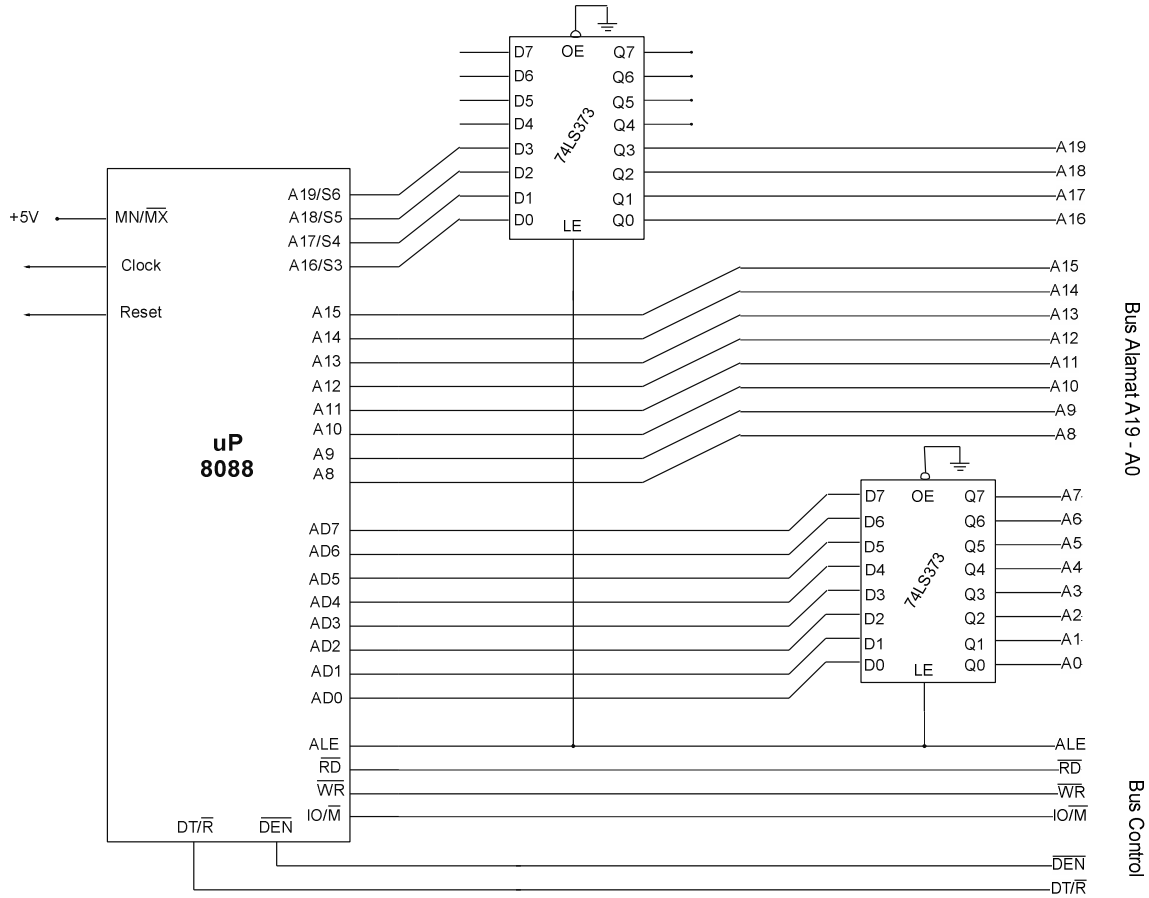
Untuk melakukan demultipleks bus alamat digunakan komponen Latch Transparan 74LS373. Latch Transparan ini berlaku seperti sistem kancing yaitu jika pin LE (Latch Enable) berlogika 1, maka IC akan melewatkan data dari Pin Input (D7-D0) ke Pin Output (Q7-Q0). Ketika pin LE kembali ke logika 0, menyebabkan latch menahan/menyangga data output tidak berubah selama kondisi tersebut, yang artinya ketika LE berlogika 0 nilai output (Q7-Q0) tidak akan berubah. Kaki $\overline{OE}$ dihubungkan ke ground untuk mengaktifkan fungsi output enable($\overline{OE}$). Seperti yang diilustrasikan pada gambar berikut :



Gambar 4.1 prinsip kerja komponen Latch 74LS373

Gambar 4.2 mengilustrasikan mikroprosesor 8088 dan komponen-komponen yang diperlukan untuk demultipleks busnya. Dalam hal ini, dua latch transparan 74LS373 digunakan untuk demultipleks hubungan bus alamat/data AD7-AD0 dan hubungan alamat/status A19/S6 – A16/S3 yang termultipleks.

Pin LE pada komponen Latch 74LS373 dihubungkan dengan Pin ALE (Address Latch Enable) dari 8088. Jika Pin ALE berlogika 1, melewatkan input ke output. Setelah selang waktu yang singkat, ALE kembali ke kondisi logika 0-nya, yang menyebabkan latch mengingat input yang dilewatkan tadi pada saat perubahan ke logika 0. Dalam hal ini, A7-A0 disimpan pada lacth bawah dan A19-A16 disimpan pada latch atas. Hal ini menghasilkan bus alamat yang terpisah dengan hubungan A19-A0. Hubungan alamat ini memungkinkan mikropsesor 8088 mengalamatkan 1M byte ruang memori. Fakta bahwa bus data terpisah memungkinnya untuk dihubungkan ke alat periferal atau komponen memori 8 bit apa saja.



Gambar 4.2 Demultipleks alamat pada uP 8088 dengan menggunakan dua komponen Latch 74LS373 sebagai Buffering dan Latching Bus Alamat AD7-AD0 dan A19/S6 – A16/S3.

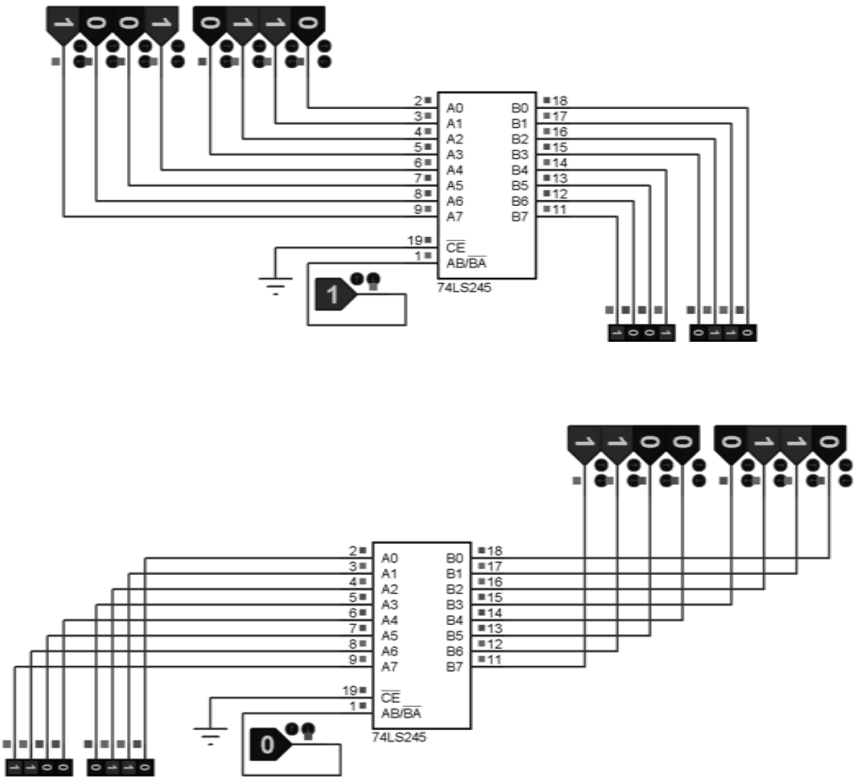
- **Demultipleks Bus Data 8 Bit**

Pada bagian sebelumnya kita telah mempelajari bagaimana proses demultipleks bus alamat A19-A0 dari mikroprosesor 8088. Ada bagian penting lainnya yang belum dibahas yaitu demultipleks bus data. Seperti yang telah diketahui, bus alamat digunakan untuk mengirimkan informasi alamat/lokasi yang menunjukkan dimana data akan ditempatkan(write) atau diambil(read) dari memori/IO atau ke memori/IO. Bus alamat dikirimkan oleh mikroprosesor yang menunjukkan komunikasi satu arah saja yaitu dari mikroprosesor ke memori/IO.

Sedangkan Bus data berisikan informasi data yang akan ditempatkan ke memori/IO atau diambil dari memori/IO. Hal ini menunjukkan bahwa aliran komunikasi data pada bus data adalah komunikasi dua arah (biderksional) .Komunikasi data dua arah ini ditunjukkan dari aliran data yang terjadi pada bus data yaitu aliran data dari mikroprosesor ke memori/IO atau aliran data dari memori/IO ke mikroprosesor.

Untuk melakukan demultipleks (menyaring informasi data) dari alamat dan data yang termultipleks (digunakan bersama) pada pin AD7-AD0 mikroprosesor 8088, sekaligus untuk menangani komunikasi data bidereksional digunakan buffer bidireksional oktal (8 bit) 74LS245. Sebelum kita membahas penggunaan 74LS245 pada sistem bus data 8088, terlebih dahulu kita pelajari prinsip kerja dari buffer bidireksional 74LS245.

Perhatikan gambar berikut:



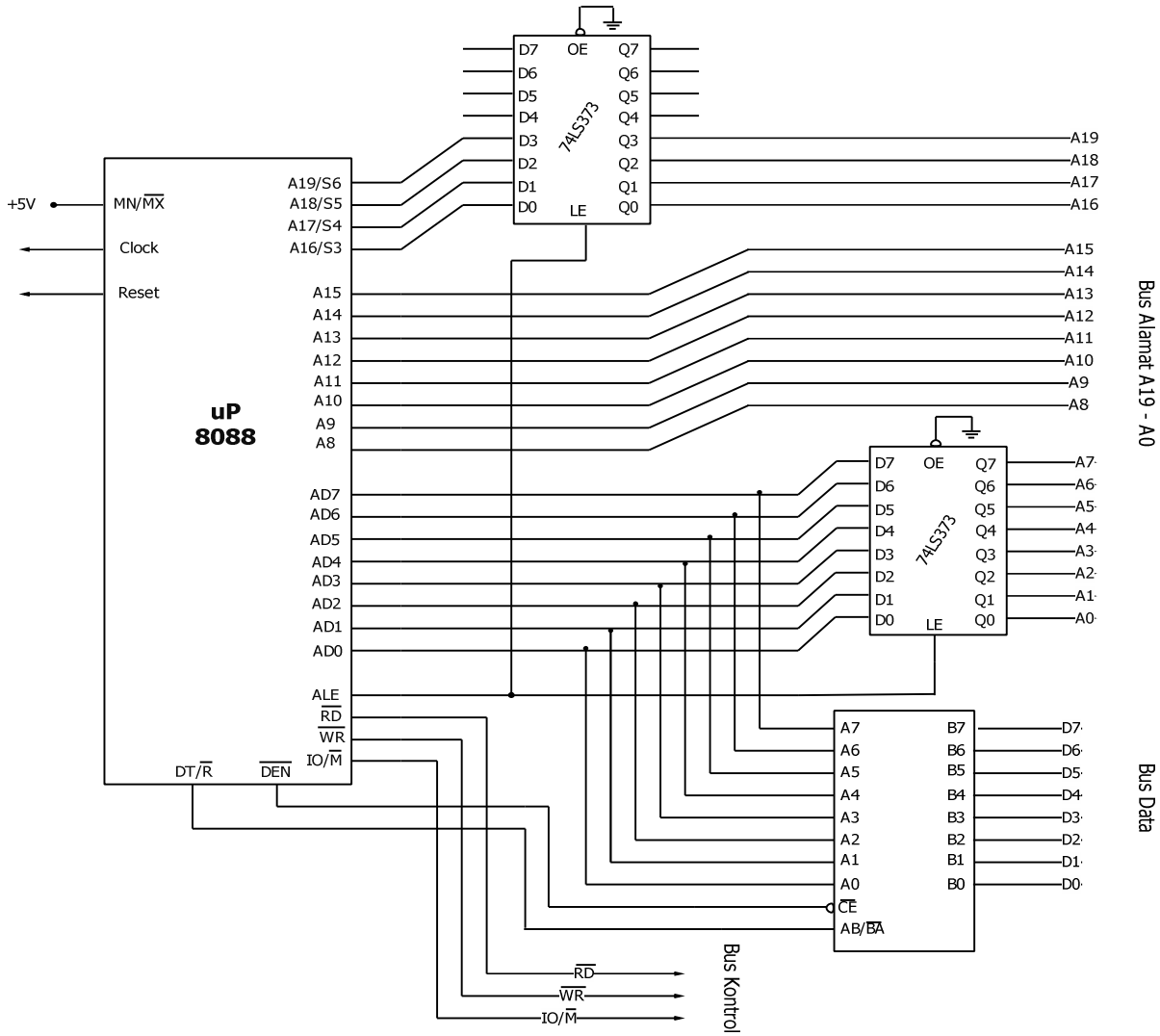
Gambar 4.4 Prinsip Kerja Buffer Bus Direksional Oktal 74LS245

Ketika Pin AB/$\overline{BA}$ diberi logika “1”, fuffer (Gambar 74LS245 Atas) akan melewatkan data dari A(A7-A0) ke B(B7-B0) yang menunjukkan data aliran data keluar (B sebagai Output). Sedangkan pada gambar 74LS245 bawah, dimana pin AB/$\overline{BA}$ diberi logika “0” maka buffer akan melewatkan data dari B (B7-B0) ke A(A7-A0) yang menunjukkan aliran data masuk (B sebagai input). Sedangkan kaki $\overline{CE}$ dihubungkan ke ground guna meng-aktifkan fungsi chip enable ($\overline{CE}$).

Dari prinsip kerja buffer bidireksional 74LS245 kita dapat menggunakannya untuk proses demultipleks aliran data pada pin AD7-AD0 (alamat/data termultipleks). Kaki $\overline{CE}$ pada buffer 74LS245 dihubungkan dengan pin $\overline{DEN}$ dari mikroprosesor. Ketika mikroprosesor 8088 mengeluarkan ataupun mengambil data (bukan inforamasi alamat) melalui pin AD7-AD0, Pin $\overline{DEN}$ akan berlogika "0" sehingga fungsi Chip Enable($\overline{CE}$) buffer 74LS245 akan diaktifkan. Ketika mikroprosesor 8088 mengeluarkan informasi alamat (bukan data) yang melalui pin AD7-AD0 maka Pin $\overline{DEN}$ akan diberi logika "1" sehingga buffer Chip Enable ($\overline{CE}$) 74LS245 tidak diaktifkan.

Sementara itu pin AB/$\overline{BA}$ pada 74LS245 dihubungkan ke pin DT/$\overline{R}$ pada mikroprosesor 8088. Pin DT/$\overline{R}$ akan berlogika "1" ketika mikroprosesor 8088 mengirimkan data ke memori/IO (melalui AD0-AD7) sebaliknya Pin DT/$\overline{R}$ akan berlogika "0" ketika mikroprosesor 8088 membaca data dari memori/IO (melalui AD0-AD7) ke mikroprosesor 8088. Dengan dihubungkannya pin AB/$\overline{BA}$ dari 74LS245 ke Pin DT/$\overline{R}$ dari 8088, maka komunikasi dua arah pada bus data dapat dilakukan.

Penggunaan bufer 74LS245 untuk demultipleks bus data pada sistem minimum mikroprosesor 8088 dapat dilihat pada gambar berikut ini:



Gambar 4.5 Demultipleks Bus Alamat dan Bus Data

TIMING BUS

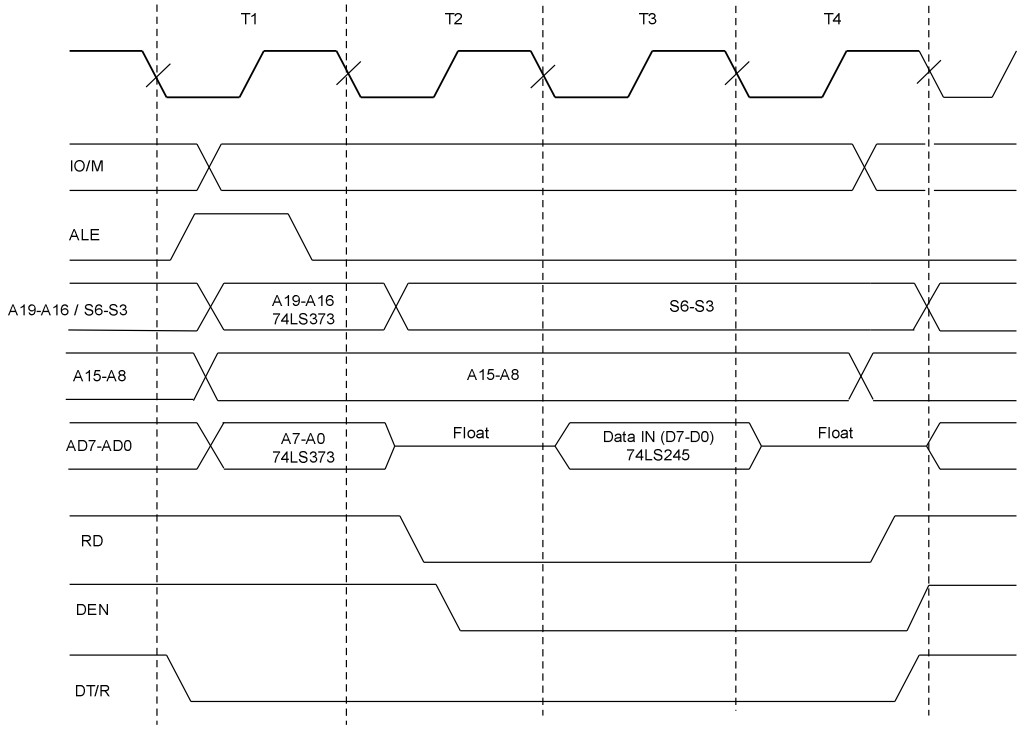
Adalah penting untuk memahami timing bus sistem sebelum memilih memori atau alat I/O untuk pengantarmukaa dengan mikroprosesor 8088. Pada pokok bahasan kali ini memberikan pengertian mengenai operasi sinyal bus, dan timing baca/tulis dasar pada 8088.

Tiga bus pada 8088 : alamat, data dan kontrol berfungsi dengan cara yang tepat sama dengan mikroprosesor lain. Jika data ditulis ke memori, mikroprosesor mengeluarkan alamat memori ke bus alamat, mengeluarkan data yang akan ditulis ke memori pada bus data, dan mengeluarkan sinyal write ($\overline{WR}$) ke memori dan IO/$\overline{M}$ di set pada logika “0”. Jika data dibaca dari memori, mikroprosesor mengeluarkan alamat memori ke bus alamat, mengeluarkan sinyal memori read ($\overline{RD}$), dan menerima data melalui bus data.

Mikroprosesor menggunakan memori dan I/O dalam periode yang disebut siklus bus. Setiap siklus bus sama dengan empat periode clocking sistem (level T). Selama periode pertama pada siklus bus, yang disebut T1, banyak hal yang terjadi. Alamat memori atau lokasi I/O dikirim melalui bus alamat dan hubungan bus alamat/data (termultipleks). Selama Siklus T1 sinyal –sinyal kontrol ALE, DT/$\overline{R}$ dan IO/$\overline{M}$ (8088) juga dikeluarkan. Sinyal IO/$\overline{M}$ atau menunjukan apakah bus berisi alamat memori atau nomor alat I/O (alamat Port I/O).

Pada waktu T2, mikroprosesor 8088 mengeluarkan sinyal $\overline{RD}$ atau $\overline{WR}$, $\overline{DEN}$ dan pada kasus write, data yang akan ditulis muncul pada bus data. Peristiwa-peristiwa ini menyebabkan memori atau perangkat I/O mulai melakukan baca/tulis. Sinyal $\overline{DEN}$ mengaktifkan buffer bus data, sehingga memori atau I/O dapat menerima data yang akan ditulis., atau sehingga mikroprosesor dapat menerima data yang dibaca dari memori atau I/O untuk operaso baca.

- **Timing Baca**

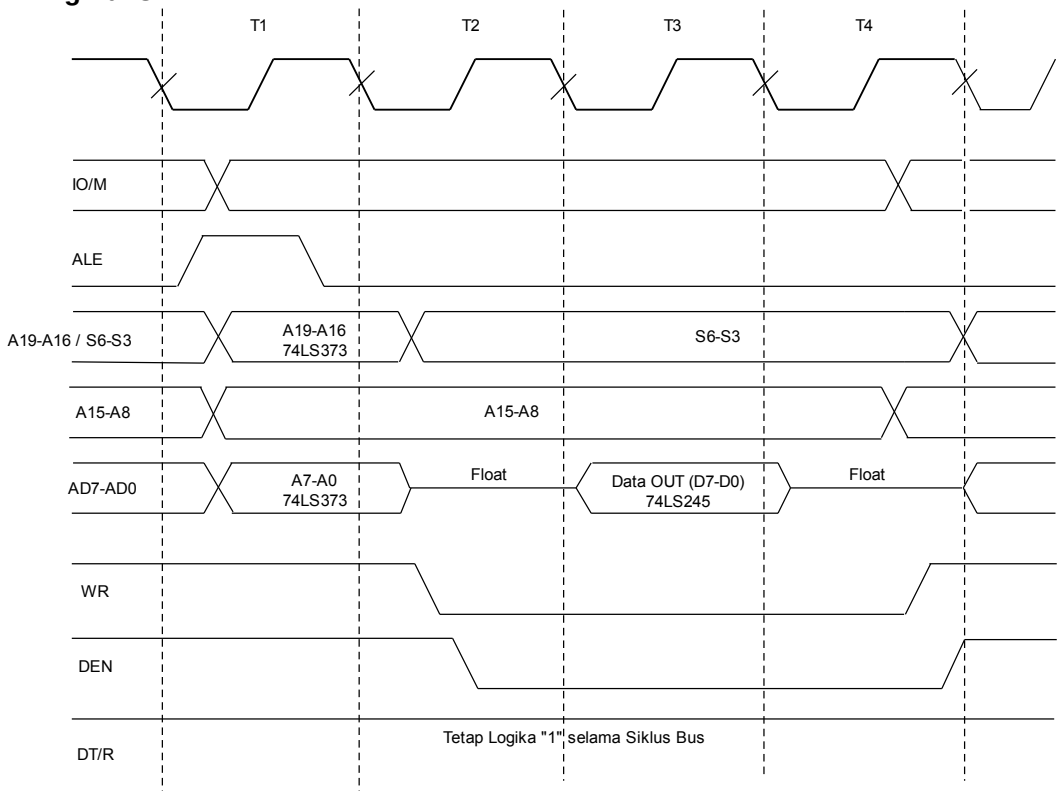


Selama T1
- Mikroprosesor mengeluarkan sinyal ALE mengaktifkan Latch Enable dari 74LS373
- Data Alamat A19-A0 dikeluarkan melalui Bus Alamat
- DT/R di set pada logika “0” agar data input (dari memori/IO) dapat diambil dari buffer 74LS245

Selama T2
- Sinyal RD (Read) di kirimkan ke Memori atau I/O
- DEN di set pada logika “0” sebagai sinyal enable buffer 74LS245

Selama T3
- Data Input dibaca dari memori melalui bus data

- **Timing Tulis**



Selama T1
- Mikroprosesor mengeluarkan sinyal ALE mengaktifkan Latch Enable dari 74LS373
- Data Alamat A19-A0 dikeluarkan melalui Bus Alamat
- DT/R di set pada logika “1” agar data out (ke memori/IO) dapat dilewatkan melaluiri buffer 74LS245

Selama T2
- Sinyal WR (Write) di kirimkan ke Memori atau I/O
- DEN di set pada logika “0” sebagai sinyal enable buffer 74LS245

Selama T3
- Data Out ditulis ke memori melalui bus data

BAB V

PENDEKODEAN ALAMAT MEMORI dan I/O

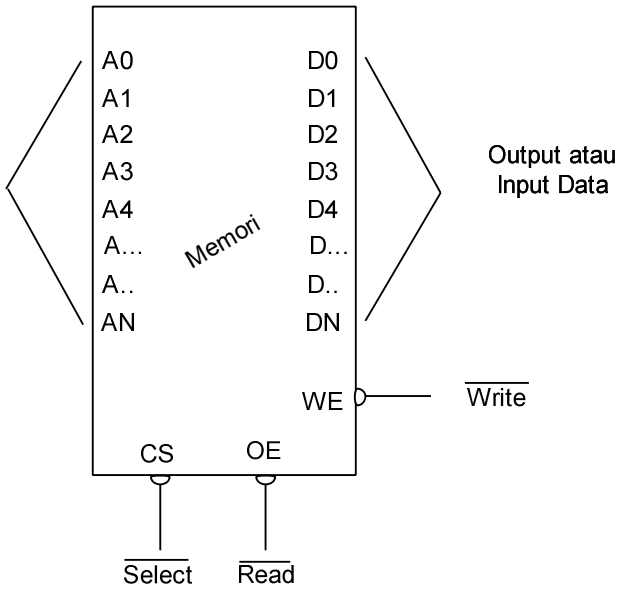
PERANTI MEMORI

Sebelum mengantarmukakan memori ke mikroprosesor, adalah penting untuk benar-benar mengerti operasi komponen-komponen memori. Pada pembahasan kali ini akan dijelaskan fungsi dari empat tipe umum memori:

- Memori hanya baca (ROM – Read Only Memory)
- Memori flash (EEPROM)
- Static random acces memory (SRAM)
- Dinamik random acces memory (DRAM)

Hubungan Pin Memori

Hubungan pin yang umum untuk semua piranti memori adalah masukan (input) alamat, masukan/keluaran (input/Output) data, beberapa tipe masukan seleksi dan paling tidak satu input kontrol yang digunakan untuk menyeleksi operaso baca dan tulis. Seperti yang terlihat pada gambar berikut ini:



Gambar 5.1 Hubungan Pin Memory (secara umum)

A0- AN digunakan sebagai masukan alamat
D0-DN digunakan sebagai masukan/keluaran data
$\overline{\text{Select}}$ digunakan untuk mengaktifkan fungsi chip select (CS)
$\overline{\text{Read}}$ digunakan untuk mengaktikan fungsi Output Enable (OE) untuk proses baca data dari memori
$\overline{\text{Write}}$ digunakan untuk mengaktikan fungsi write untuk proses tulis ke memori

Hubungan Alamat

Semua piranti memori memiliki input alamat yang memilih lokasi memori di dalam peranti memori. Input alamat hampir selalu diberi label dari A_0, yang paling keil dari input sampai A_n dimana n bisa nilai apapun selalu diberi label sebagai kurang satu dari jumlah pin (n = jumlah pin alamat – 1). Sebagai contoh sebuah memori yang memiliki 10 pin alamat memiliki pin yang diberi label A0 – A9.

Jumlah dari lokasi memori dapat diperkirakan dari jumlah pin alamat. Sebagai contoh, peranti memori yang memiliki 10 (A0-A9) pin alamat memiliki lokasi memori 2^{10} = 1024 lokasi atau 1K (1024) lokasi memori begitu juga seterusnya. Sebagai contoh sebuah memori dengan jumlah bit data yang digunakan sebanyak 8 bit (1 Byte) dan jumlah lokasi memori 1024 (1K), dapat dinyatakan kapasitas memori sebesar 1024 x 8 bit = 1K-Byte atau 1KB.

400H mewakili bagian 1K-byte dari sistem memori. Jika peranti memori dikodekan untuk digunakan / dipasang pada sistem mikroprosesor dengan alamat awal memori 10000H, maka lokasi terakhir adalah pada lamat 103FFH (satu lokasi kurang dari 400H).

Hubungan data

Semua peranti memori memiliki kumpulan data output/input. Peranti yang dilustrasikan pada gambar 5.1 memiliki kumpulan umum dari hubungan input/output (I/O). Hubungan data adalah titik dimana data dimasukkan untuk penyimpanan atau dikeluarkan untuk pembacaan. Pin data pada peranti memori diberi label D0-D7 untuk memori dengan lebar data 8 bit.

Hubungan seleksi

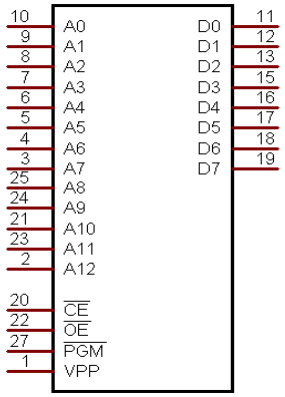
Setiap peranti memori memiliki sebuah input – kadang-kadang lebih dari satu yang memilih atau mengenable peranti memori. Input jenis ini paling sering disebut masukan sebagai suatu pemilih chip ($\overline{CS}$ chip select), enable chip ($\overline{CE}$ chip enable), atau hanya pilih ($\overline{S}$ select). Memori RAM umumnya memiliki paling tidak satu input $\overline{CS}$ atau $\overline{S}$, dan ROM palign tidak satu $\overline{CE}$. Jika input $\overline{CE}$, $\overline{CS}$ atau $\overline{S}$ aktif (sebuah logika 0 karena garis diatas), peranti memori memberikan sebuah pembacaan atau penulisan. Jika pin ini tidak aktif (sebuh logika 1), peranti tidak dapat melakukan sebuah pembacaan atau penulisan data karena dimatikan atau didisabel. Jika ada lebih dari satu penghubung $\overline{CS}$, maka semuanya harus diaktifkan untuk membaca atau menulis data.

Hubugan kontrol

Semua peranti memori memilki beberapa beuntuk input kontrol atau banyak input kontrol. Sebuah ROM biasanya memiliki satu input kontrol, dimana RAM sering memiliki satu atau dua input kontrol.

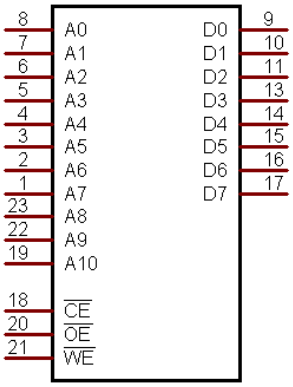
Input kontrol yang sering ditemukan pada ROM adalah output enable ($\overline{OE}$) atau hubungan gerbang ($\overline{G}$) gate, yang memungkinkan data mengalir keluar dari pin output data dari ROM. Jika $\overline{OE}$ tidak aktif, output tidak diaktifkan sehingga pembacaan data tidak dapat dilakukan.

Input kontrol yang sering ditemukan pada peranti RAM memiliki satu maupun dua input kontrol. Jika ada satu input kontrol, sering disebut sebagai R/$\overline{W}$. Pin ini memilih mode operasi pembacaan atau penulisan hanya jika peranti input $\overline{CS}$ diaktifkan. Jika RAM memiliki dua input kontrol, biasanya diberi label $\overline{WE}$ (write enable) dan $\overline{OE}$ (output enable) atau $\overline{G}$. disini $\overline{WE}$ difungsikan untuk mengaktifkan mode penulisan ke memori, sedangkan $\overline{OE}$ difungsikan untuk mengaktifkan mode pembacaan data dari memori. Kedua pin tidak boleh aktif bersamaan.



$\overline{CE}$ => Chip Enable
$\overline{OE}$ => Output Enable
$\overline{PGM}$ => Chip Programmer
VPP => Catu Daya Mode Program

Gambar 5.2 Contoh Peranti memori EPROM 8KB 2764 (13 bit alamat) dan fungsi pin



$\overline{CE}$ => Chip Enable
$\overline{OE}$ => Output Enable (Read Enable)
$\overline{WE}$=> Write Enable

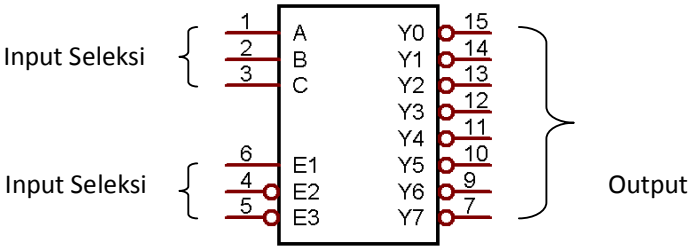
Gambar 5.3 Contoh Peranti memori RAM 2KB 6116 (11 bit alamat) dan fungsi pin

Pendekodean Alamat Memori

Agar dapat menyambungkan peranti memori ke mikroprosesor, diperlukan pendekodean alamat dari mikroprosesor untuk membuat fungsi memori pada bagian yang unik atau partisi dari peta memori. Tanpa pendekode alamat, hanya satu peralatan memori yang dapat dihubungkan ke mikroprosesor. Jika mikroprosesor 8088 dibandingkan dengan EPROM 2716 terjadi perbedaan jumlah hubungan alamat. EPROM 2716 memiliki 11 hubungan alamat sedangkan 8088 memiliki 20 hubungan alamat. Ini berarti bahwa mikroprosesor mengirimkan 20 bit setiap kali membaca atau menulis data. Karena EPROM hanya memiliki 11 input alamat, ada ketidakcocokan yang dikoreksi. Dekoder mengoreksi ketidakcocokan dengan mendekodekan alamat pin yang tidak terhubung ke komponen memori.

DEKODER 74LS138

Pendekodean dapat dilakukan dengan menggunakan sebuah IC dekoder 74LS138. Prinsip kerja IC dekoder ini dapat dilihat pada gambar berikut:



Inputs					Outputs							
Enable		Select										
G1	G2 (Note 8)	C	B	A	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
X	H	X	X	X	H	H	H	H	H	H	H	H
L	X	X	X	X	H	H	H	H	H	H	H	H
H	L	L	L	L	L	H	H	H	H	H	H	H
H	I	I	I	H	H	I	H	H	H	H	H	H
H	I	I	H	I	H	H	I	H	H	H	H	H
H	L	L	H	H	H	H	H	L	H	H	H	H
H	L	H	L	L	H	H	H	H	L	H	H	H
H	L	H	L	H	H	H	H	H	H	L	H	H
H	L	H	H	L	H	H	H	H	H	H	L	H
H	L	H	H	H	H	H	H	H	H	H	H	L

H = High Level, L = Low Level, X = Don't Care

Keterangan:
G1=E1 , G2=E2 G3= E3

Contoh 1:

Sebuah EPROM 8K-byte dihubungkan dengan mikroprsesor 8088 pada dengan alamat awal FE000H, pertama kita mengilustrasikan logika 0 untuk 13 bit alamat pada EPROM sebagai alamat terendah, kemudian logika 1 pada 13 bit alamat sebagai alamat tertinggi.

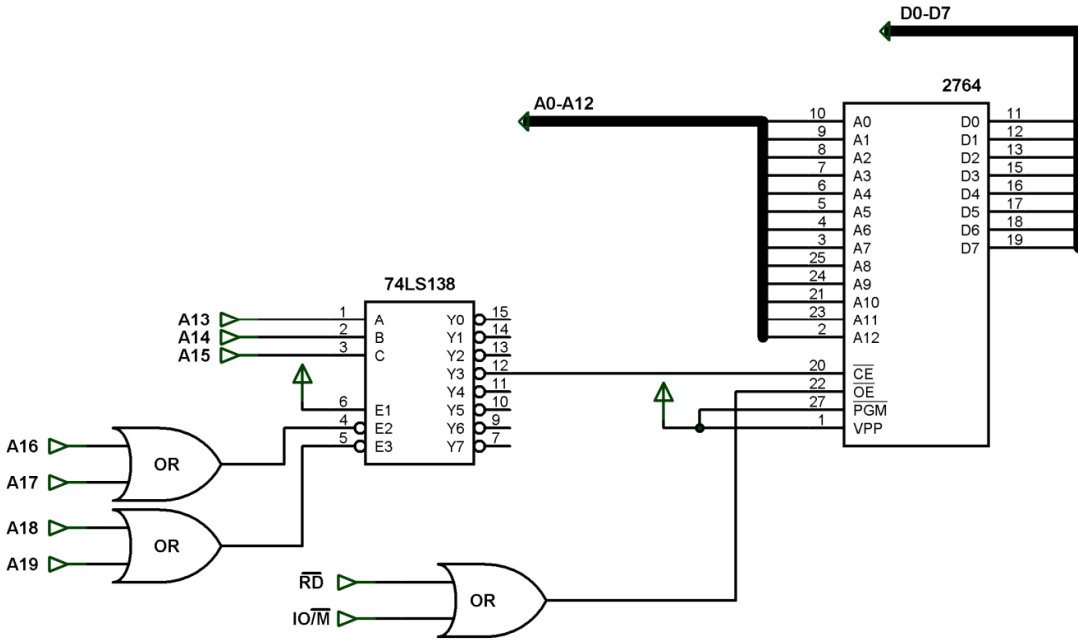
EPROM 8KB ~ 13 bit ($A_{12} – A_0$)

$A_{19} – A_{16}$	$A_{15} – A_{12}$	$A_{11} – A_8$	$A_7 – A_4$	$A_3 – A_0$	
0 0 0 0	0 1 1 0	0 0 0 0	0 0 0 0	0 0 0 0	~ 06000
0 0 0 0	0 1 1 1	1 1 1 1	1 1 1 1	1 1 1 1	~ 07FFF

Mendekodekan alamat pin yang tidak terhubung dengan komponen memori

A_{19}	A_{18}	A_{17}	A_{16}	A_{15}	A_{14}	A_{13}
0	0	0	0	0	1	1

Proses pendekodean dapat menggunakan 2 buah gerbang OR dan 1 buah dekoder 74LS138 seperti yang terlihat pada gambar berikut:



Gambar 5.4 Pendekodean Alamat EPROM 2764 8K-Byte dengan alamat 6000H – 7FFFH

Perhatikan bahwa sinyal kontrol IO/$\overline{M}$ digunakan untuk mengisolasi pengaksesan antara perangkat memori dan perangkat I/O.

Contoh 2:

Buatlah rangkaian dekoder yang digunakan untuk mengakses 1 buah ROM 2 KB dan 1 buah RAM 1KB dan 1 buah RAM 2Kbyte. Dengan kondisi:
- Alamat awal EPROM 2 KB 1800H
- Alamat awal RAM-1KB 300H
- Alamat awal RAM-2KB berdekatan dengan alamat akhir RAM-1KB

Solusi:
Pemetaan Memori
1KB ~ 10 bit alamat A0-A9 ~> lebar alamat 400H
2KB ~ 11 bit alamat A0-A10 ~>lebar alamat 800H
urutkan partisi dari alamat terendah sampai alamat tertinggi

RAM-1KB	400H
	7FFH
RAM-2KB	800H
	FFFH
ROM 2KB	1800H
	1FFFH

RAM-1KB (alamat yang digunakan A_0 - A_9)

A_{19} - A_{16}	A_{15} - A_{12}	A_{11} - A_8	A_7 - A_4	A_3 - A_2	
0 0 0 0	0 0 0 0	0 1 0 0	0 0 0 0	0 0 0 0	~ 00400H
0 0 0 0	0 0 0 0	0 1 1 1	1 1 1 1	1 1 1 1	~ 007FFH

RAM-2KB (alamat yang digunakan A_0 – A_{10})

A_{19} - A_{16}	A_{15} - A_{12}	A_{11} - A_8	A_7 - A_4	A_3 - A_2	
0 0 0 0	0 0 0 0	1 0 0 0	0 0 0 0	0 0 0 0	~ 00800H
0 0 0 0	0 0 0 0	1 1 1 1	1 1 1 1	1 1 1 1	~ 00FFFH

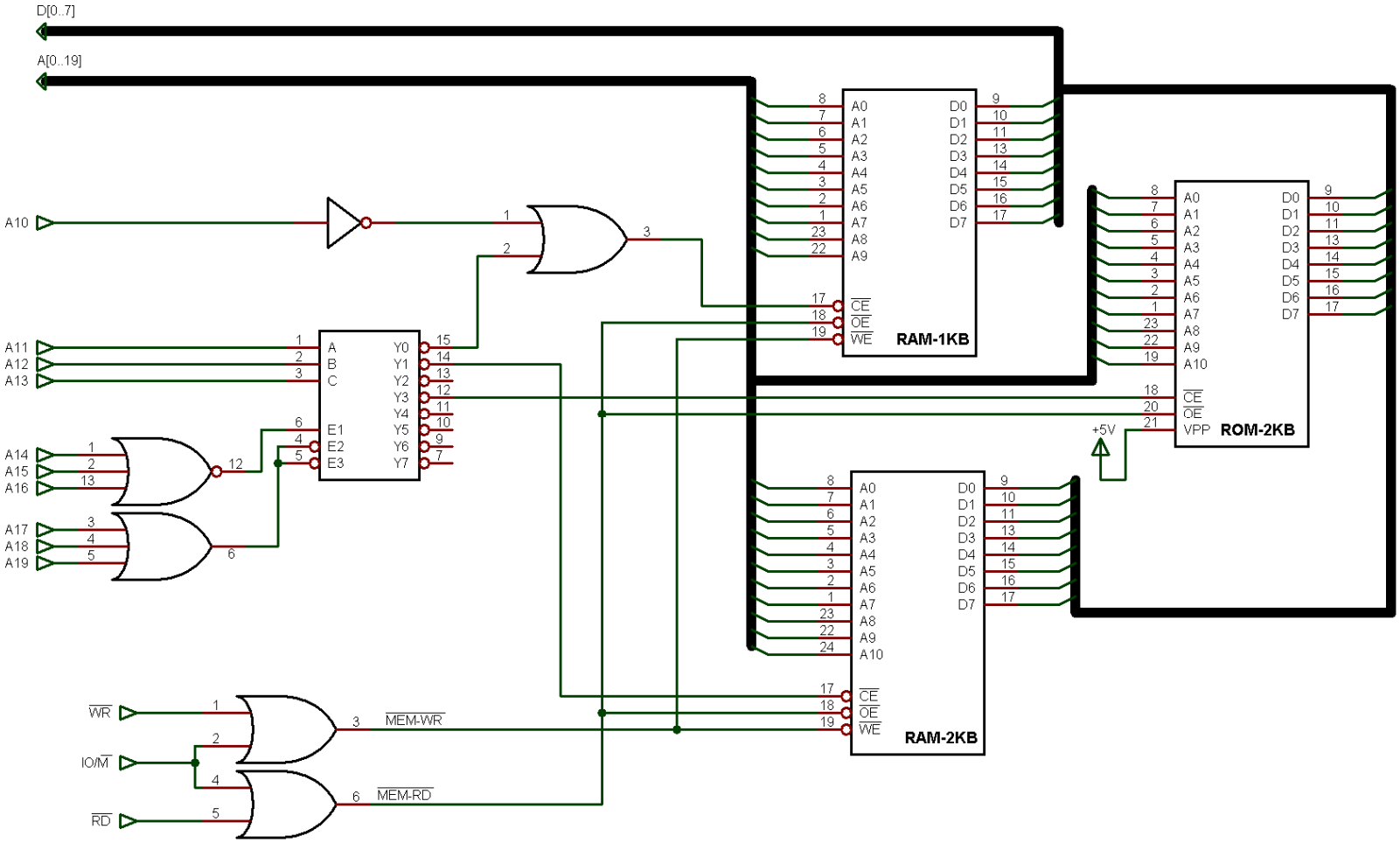
ROM 2KB (alamat yang digunakan A_0 – A_{10})

A_{19} - A_{16}	A_{15} - A_{12}	A_{11} - A_8	A_7 - A_4	A_3 - A_2	
0 0 0 0	0 0 0 1	1 0 0 0	0 0 0 0	0 0 0 0	~ 01800H
0 0 0 0	0 0 0 1	1 1 1 1	1 1 1 1	1 1 1 1	~ 01FFFH

Alamat yang didekodekan:

A_{19}	A_{18}	A_{17}	A_{16}	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	KET
0	0	0	0	0	0	**0**	**0**	**0**	**1**	RAM-1KB
0	0	0	0	0	0	**0**	**0**	**1**	X	RAM-2KB
0	0	0	0	0	0	**0**	**1**	**1**	X	ROM-2KB

Dari tabel alamat diatas dapat didekodekan dengan menggunakan dekoder 74LS138 , gerbang NOR dan Gerbang OR. Bentuk rangkaian selengkapnya dapat dilihat pada gambar 5.5.



Gambar 5.5 Rangkaian pendekodean alamat memori menggunakan dekoder 74LS138 dan gerbang logika

Contoh instruksi pembacaan data memori pada alamat 1045H (ROM2KB)

```
MOV  BX, 1045H
MOV AL, [BX];
```

Jika alamat memori yang digunakan lebih besar dari 64 KB maka instruk pemindahan data dilakukan dengan bantuan register DS sebagai Alamat Segmen Data dan BX sebagai Alamat Offset.

Contoh instruksi pembacaan data memori pada alamat 18903H (Alamat Fisik)

```
MOV DS, 1800H
MOV BX, 0903H
MOV AL,[BX]

;DS = 1800H;BX = 0903H;DS + BX = 18903H
```

Contoh instruksi penulisan data 20H ke memori pada alamat 678H

```
MOV AL, 20H
MOV BX,0678H
MOV [BX], AL
;DS ~> 000H
```

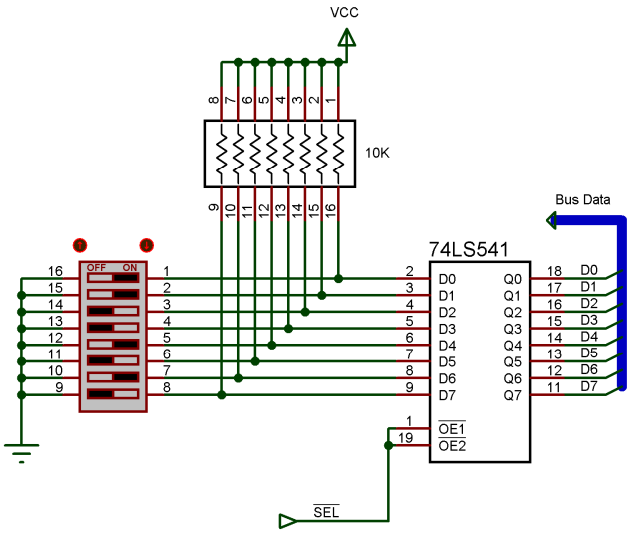
PENDEKODEAN ALAMAT I/O

Piranti masukan dasar adalah buffer 3-state. Piranti keluaran dasar adalah kumpulan lacth data. IN adalah perintah untuk memindahkan data dari piranti I/O ke mikroprosesor dan OUT adalah perintah untuk memindahkan data keluar dari mikroprosesor ke piranti I/O.

Antarmuka Masukan Dasar

Buffer 3-state digunakan untuk membuat port masukan 8-pin yang digambarkan dalam gambar 5.6. Perhatikan bahwa data TTL luar (toggle switch sederhana dalam contoh ini) dihubungkan ke masukan buffer. Keluaran buffer dihubungkan ke bus data.

Ketika mikroprosesor mengeksekusi instruksi IN, port I/O didekode untuk menghasilkan logika “0” pada SEL. Logika “0” ditempatkan pada pada output dari kendali input (OE1 dan OE2) buffer 74LS541 yang mengakibatkan koneksi input data (D) disambungkan ke koneksi data output (Q). jika Logika “1” diletakkan pada output dari kendali input buffer dari 74LS541 piranti tersebut akan masuk ke mde 3 keadaan dengan impendasi tinggi yang seccara efektif memutuskan koneksi saklar dari bus.



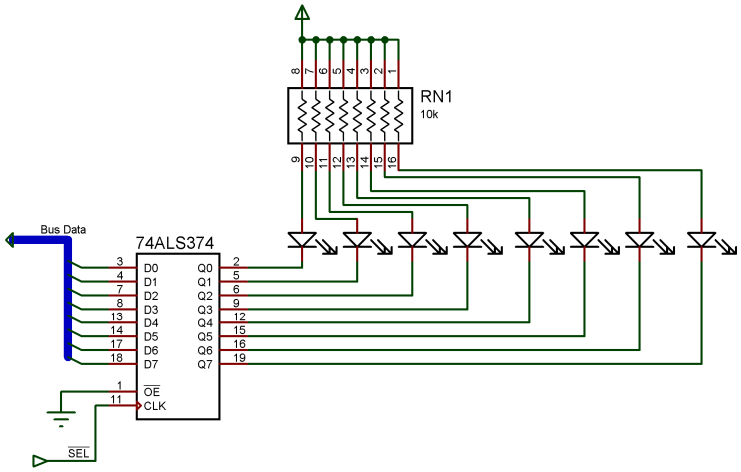
5.6 Antarmuka masukan dasar yang mengilustrasikan koneksi 8 buah switch.

Antarmuka Keluaran Dasar

Antarmuka keluaran dasar menerima data dari miroprosesor dan lazimnya menggunakan penahan data untuk berhubungan dengan piranti luarnya. Penahan data ini disebut dengan latch atau flip-flop. Latch seperti buffer yang dipakai pada piranti masukan, yang sering pula dipakai dalam piranti I/O.

Pada gambar 5.7 adalah suatu contoh yang sederhana bagaimana membangun suatu rangkaian keluaran untuk keperluar I/O. Latch dibutuhkan untuk menahan data, karena ketika mikroprosesor mengeksekusi instruksi OUT, data hanya terdapat pada bus data dengan waktu penahan yang sangat singkat. Tanpa latch ini kita tidak akan pernah melihat LED bercahaya.

Ketika instruksi OUT dilakukan data dari AL dipindahkan ke latch melalui bus data. Disini, masukan D pada lacth octal 74ALS374 dihubungkan ke bus data untuk menangkap data keluaran , dan keluaran Q dari latch disambungkan ke LED. Pada saat keluaran Q berada pada logika “0”, LED bercahaya. Setiap waktu instruksi OUT dilakukan sinyal SEL ke latch aktif, menangkap keluaran data ke latch dati setiap bagian 8-bit bus data. Data disimpan sampai instruksi OUT berikutnya dilaksanakan. Dengan demikian kapanpun instruksi keluaran dilakukan dalam rangkaian ini, data dari register AL tampak pada LED.



Gambar 5.7 Antarmuka keluaran dasar yang dihubungkan ke 8 peraga LED

Pendekodean alamat dari gerbang I/O adalah sama dengan pendekodean alamat memori. Hanya saja pada sinyal kontrol untuk melakukan Read dan Write yang dikombinasikan dengan sinyal IO/M=1 (IO/M=0 untuk akses memori).

Perbedaan lainnya yang paling utama adalah antara pendekodean memori dan I/O tersiolasi memori ada pada jumlah pin-pin alamat yang dihubungkan dengan dekoder. Kita mendekodekan A19-A0 untuk memori dan A15-A10 untuk gerbang I/O.

Pendekodean Alamat I/O 8 bit (00h-FFh)

Seperti yang telah dijelaskan, perintah tetap I/O menggunakan 8-bit dari alamat gerbang I/O yang terlihat pada A15-A0 sebagai 0000H – 00FFH. Jika sistem tidak memiliki lebih dari 256 peranti I/O, kita sering hanya mendekodekan alamat yang menghubungkan A7-A0 sebagai gerbang pengalamatan 8-bit. Maka , kita dapat mengabaikan alamat yang menghubungkan A15-A8. Harap dicatat bahwa register DX selalu dapat mengalamatkan gerbang I/O sebagai 00H-FFH.

Pendekodean Alamat I/O 16 bit (0000H – FFFFH)

Kita juga dapat mendekodekan alamat I/O 16 bit, khususnya didalam sistem komputer personal (PC). Perbedaan yang paling utama antara pendekodean alamat I/O 8-bit dengan pendekodean alamat I/O 16-bit adalaha adanya jalur alamat tambahan (A15-A8) yang harus didekodekan.

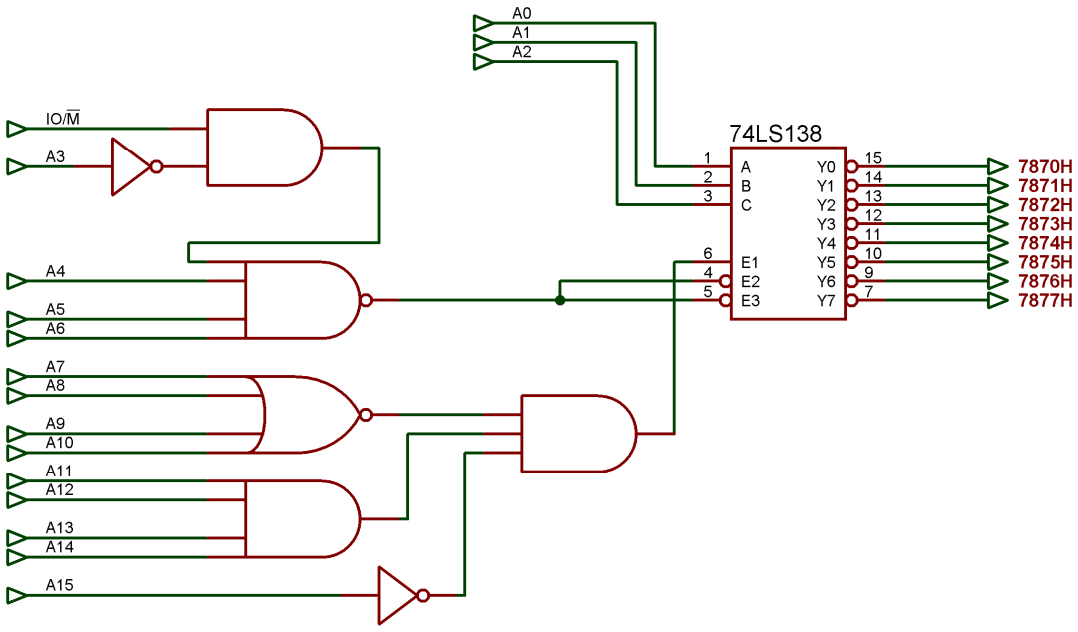
Contoh Kasus:
Buatlah rangkaian dekoder alamat gerbang I/O 16-bit pada lokasi 7870H-7877H

A_{15} - A_{12}	A_{11} - A_8	A_7 - A_4	A_3 - A_2	
0 1 1 1	1 0 0 0	0 1 1 1	0 0 0 0	~ 7870H
0 1 1 1	1 0 0 0	0 1 1 1	0 0 0 1	~ 7871H
0 1 1 1	1 0 0 0	0 1 1 1	0 0 1 0	~ 7872H
0 1 1 1	1 0 0 0	0 1 1 1	0 0 1 1	~ 7873H
0 1 1 1	1 0 0 0	0 1 1 1	0 1 0 0	~ 7874H
0 1 1 1	1 0 0 0	0 1 1 1	0 1 0 1	~ 7875H
0 1 1 1	1 0 0 0	0 1 1 1	0 1 1 0	~ 7876H
0 1 1 1	1 0 0 0	0 1 1 1	0 1 1 1	~ 7877H

Alamat yang didekodekan A15 – A3

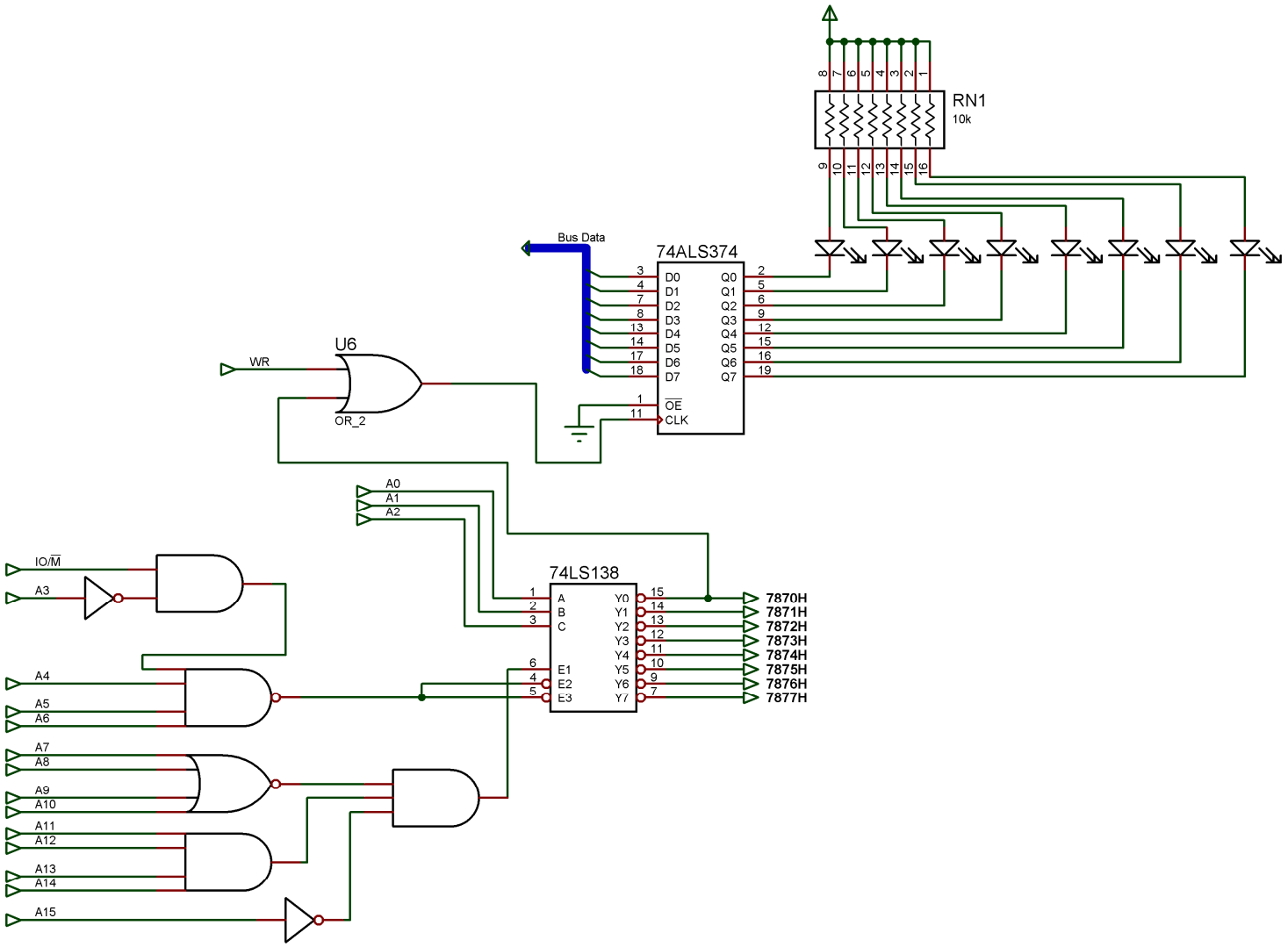
A_{15} - A_{12}	A_{11} - A_8	A_7 - A_4	A_3
0 1 1 1	1 0 0 0	0 1 1 1	0

Sinyal kontrol IO/M dikombinasikan dengan alamat yang didekodekan untuk membedakan antara pengaksesan memori dan pengaksesan gerbang I/O (I/O terisolasi memori)



Gambar 5.8 rangkaian dekoder gerbang I/O 7870H-7877H

Gambar dibawah ini adalah contoh aplikasi I/O dengan menggunakan alamat gerbang I/O 7870H yang terhubung dengan rangkaian antarmuka keluaran LED. Gerbang OR dengan Input sinyal kontrol WR digunakan untuk membedakan proses tulis (perintah OUT) dengan proses baca (perintah IN)

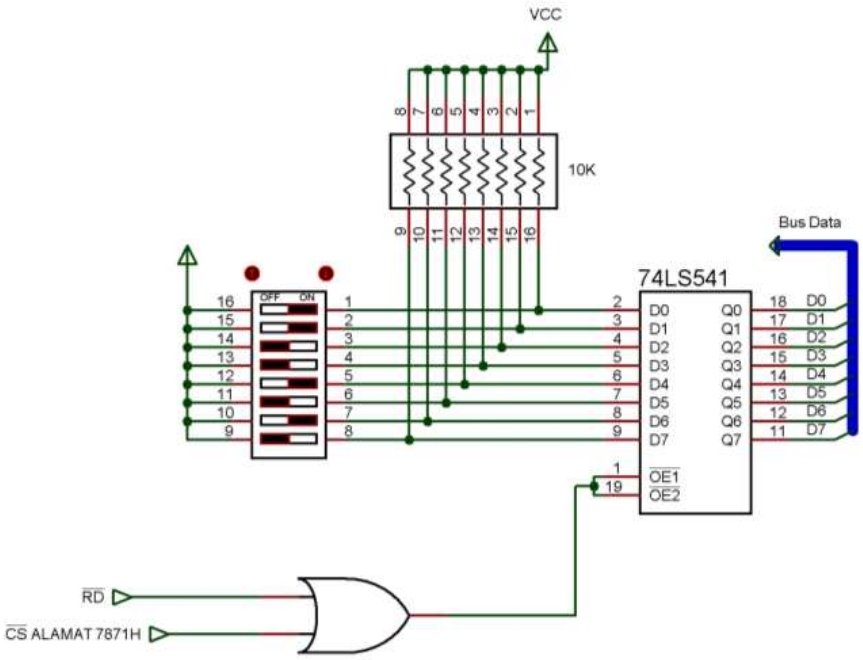

RN1
10k
Bus Data
74ALS374
D0
D1
D2
D3
D4
D5
D6
D7
Q0
Q1
Q2
Q3
Q4
Q5
Q6
Q7
OE
CLK
U6
WR
OR_2
A0
A1
A2
74LS138
A
B
C
E1
E2
E3
Y0
Y1
Y2
Y3
Y4
Y5
Y6
Y7
7870H
7871H
7872H
7873H
7874H
7875H
7876H
7877H
IO/M
A3
A4
A5
A6
A7
A8
A9
A10
A11
A12
A13
A14
A15

Contoh intruksi pada rangkaian aplikasi diatas:

MOV AL, 45H
MOV DX, 7870H
OUT DX, AL

Perhatikan bahwa alamat gerbang I/O ditempatkan terlebih dahulu ke register DX, kemudian instruksi OUT DX, AL dieksekusi. Pada saat instruksi OUT dieksekusi sinyal keluaran dari pin IO/M dari mikroprosesor di set pada logika “1” dan sinyal WR di set pada logika “0” sedangkan sinyal RD di set pada logika “1”. Alamat I/O ditempatkan pada Bus Alamat. Sehingga sinyal latch pada IC 74ALS374 diaktifkan yang menyebabkan keluaran Q akan menahan data yang dikeluarkan / dipindahkan dari register AL melalui bus Data.

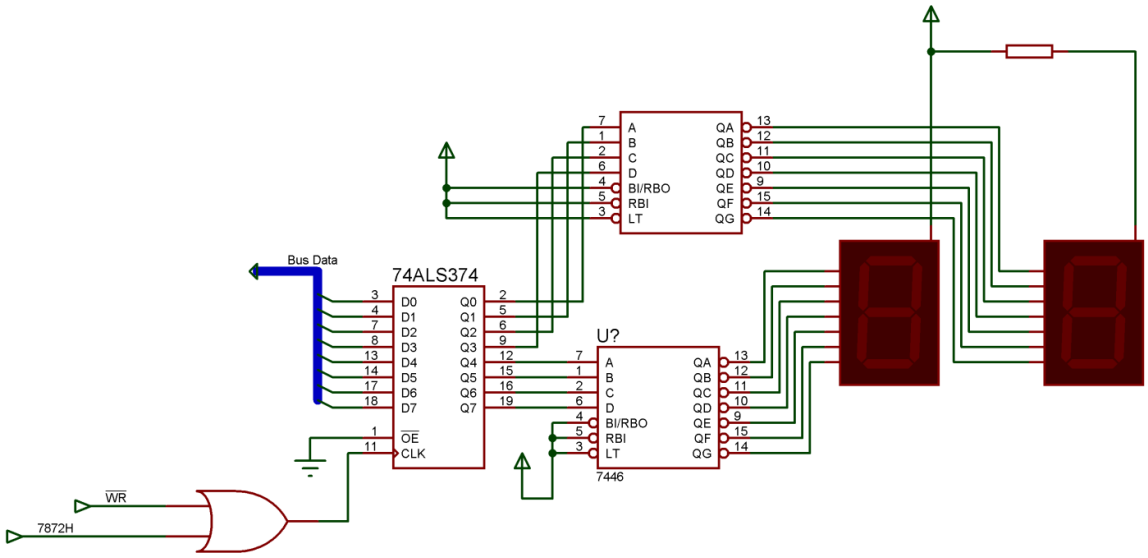
Proses yang sama juga akan terjadi pada instruksi IN, dimana sinyal WR diset pada logika”1”, sedangkan sinyal RD diset pada logika “0”. Untuk rangkaian aplikasi pembacaan input dari gerbang I/O dapat mengabungkan rangkaian dekoder dengan rangkaian pada gambar 7.9, dimana gerbang OE1 dan OE2 (dari 74LS541) dihubungkan gerbang OR yang diinputkan dari sinyal RD dan keluaran chip-select pada alamat 7871H seperti yang terlihat pada gambar 5.10 berikut ini:



Gambar 5.10 Aplikasi Input Switch pada alamat 7871H

Contoh instruksi membaca data dari inputsaklar

MOV DX, 7871H
IN AL, DX



Gambar 5.11 Rangkaian Aplikasi 2 Digit 7 Segmen pada alamat I/O 7872H

Contoh instruksi menampilkan angka 45 pada 7 segmen

MOV AL, 45H
MOV DX, 7872H
OUT DX, AL