Mikroprosessor Z80

PERTEMUAN / MATERI KE 9

Antarmuka dan Kamputer

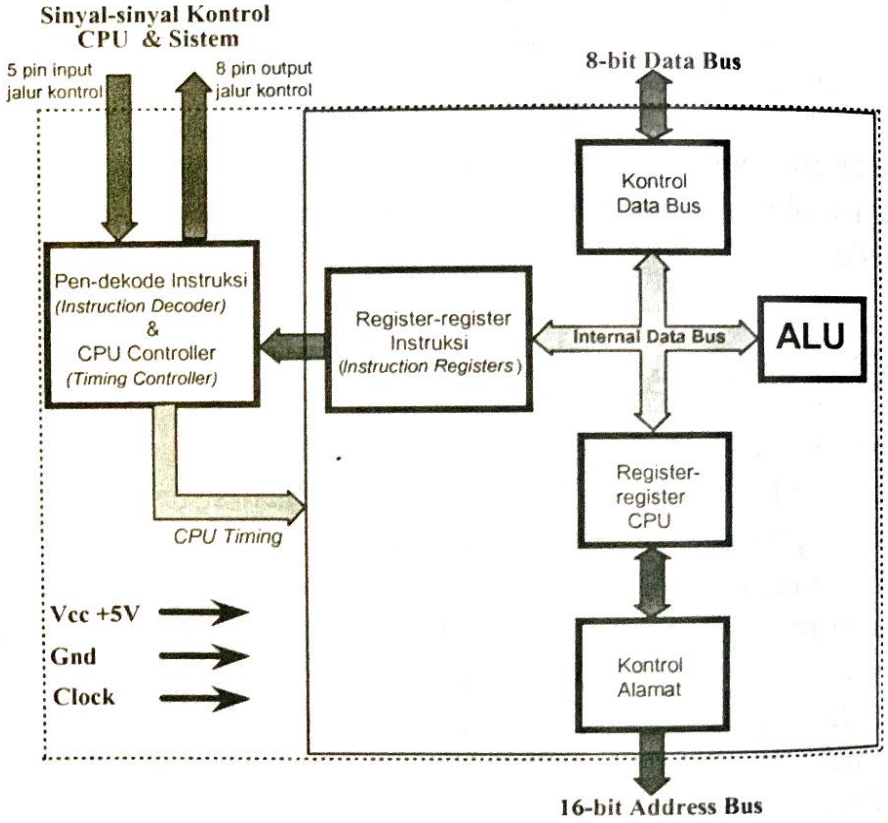
Arsitektur CPU Z80

Asitektur Z80

Arsitektur Z80 didesain dengan mengaplikasikan metode standard bus, yaitu pengelompokan jalur-jalur dalam tiga macam bus:

- ✓ Address Bus
- ✓Data Bus
- ✓Control Bus

Asitektur Z80



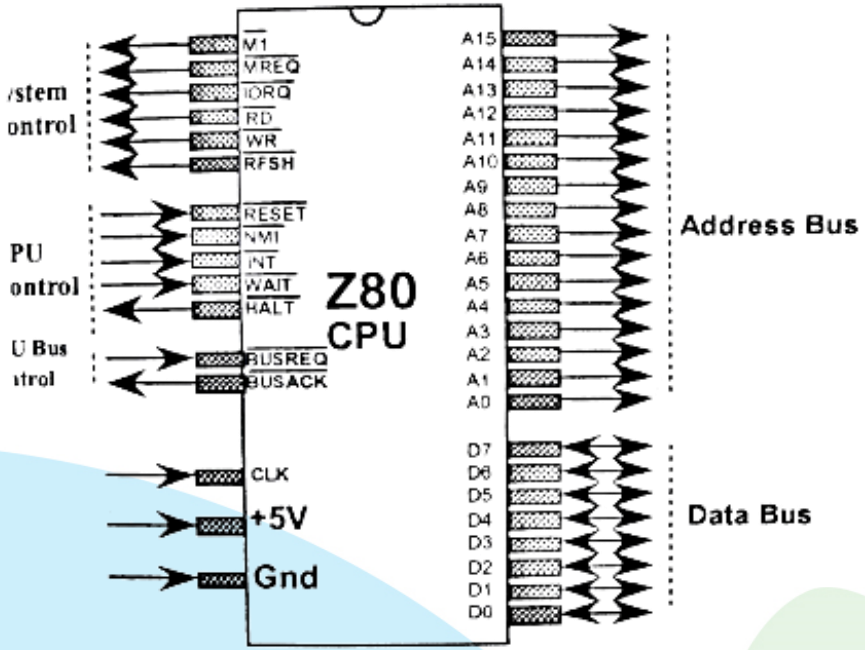
Gambar 9.1 Arsitektur Z80

Gambar 5.1 memperlihatkan arsitektur Z80, baik untuk tipe CMOS maupun tipe *non* CMOS .

CPU Z80 mengaplikasikan metode *standar bus* (*address bus, data bus* dan *control bus*).

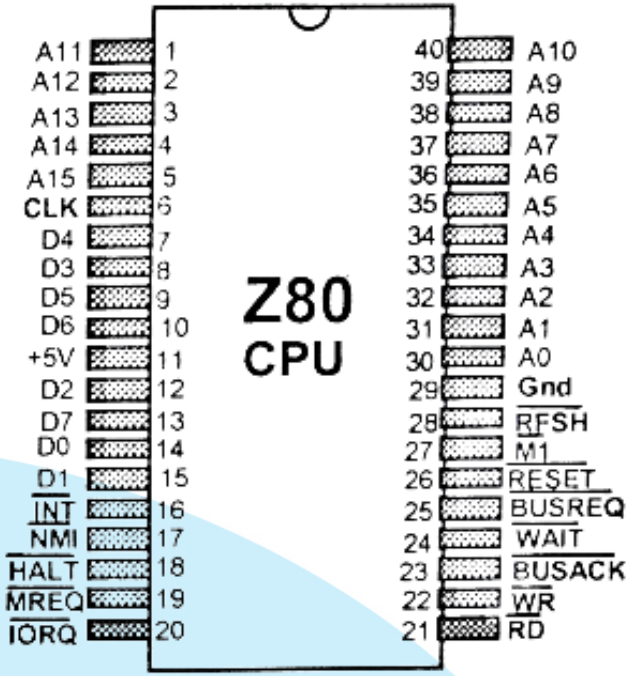
Z80 bekerja pada tegangan +5 Volt, memiliki kemampuan kerja 2 MHz, internal register 208 bit, 6 general-pupos regiter yang dapat dipisah (tiap 8 bit), tersedia instruksi exchange yang menunjang fungsi respon interrupt yang cepat.

KONFIGURASI PIN BERDASARKAN INPUT/OUTPUT



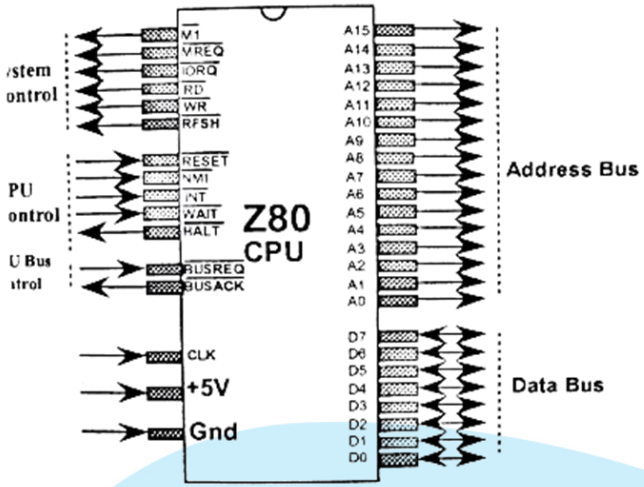
Gambar 9.2 Konfigurasi pin Z80 berdasarkan fungsi Input-Output

KONFIGURASI PIN BERDASARKAN INPUT/OUTPUT



Gambar 9.3 Konfigurasi dalam DIL (Dual In Line Package) 40 pin

Fungsi Pin Z80



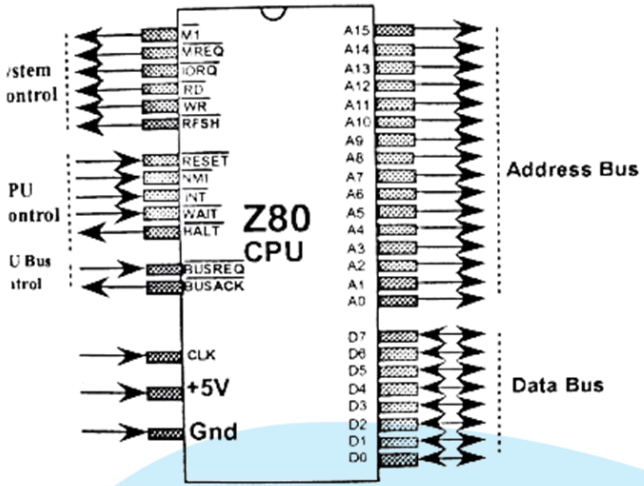
AO - A15 Address bus, berfungsi sebagai output, dapat memiliki 3 kondisi (high, low dan tri-state). A0 sampai A15 mem bentuk 16-bit address bus dengan posisi A0 sebagai L SB dan A15 sebagai MSB. Kapasitas memori dengan pengalamatan langsung adalah 64KB.

D0 - D7 Data bus, dapat berfungsi sebagai input atau output, bi sa dalam 3 kondisi (low, high dan tri-state). D0 - D7 ad alah 8-bit bidirectional address bus, digunakan untuk p ertukaran data antara CPU dan memori atau rangkaian Input/Output.

CLK Clock, berfungsi sebagai input. Pada jalur ini, harus di berikan pulsa clock dari rangkaian oscillator agar siste m dapat bekerja.

$\overline{RD}$ Read, berfungsi sebagai output, active low dan dapat memiliki 3 kondisi (low, high dan tri-state). Adanya siny al aktif menunjukkan bahwa CPU menginginkan suatu proses pembacaan data dari memori atau peralatan In put.

Fungsi Pin Z80



$\overline{WR}$

Write, berfungsi sebagai *output, active low* dan dapat me miliki 3 kondisi (*low, high* dan *tri-state*). menunjukkan bah wa CPU menginginkan suatu proses penulisan data ke memori atau peralatan *Output*.

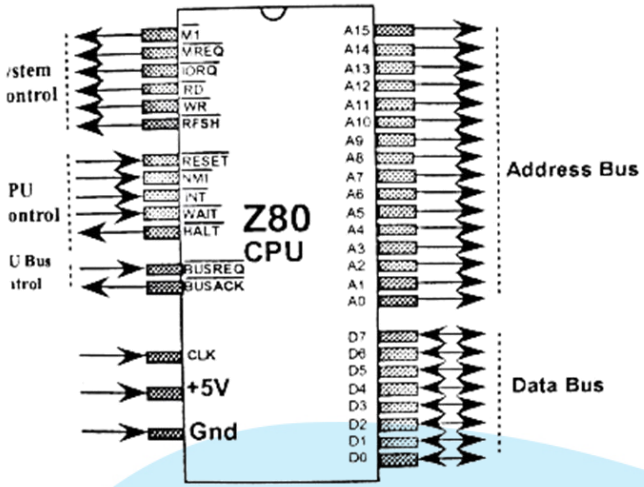
$\overline{MREQ}$

Memory Request, berfungsi sebagai output, active low, d apat memiliki 3 kondisi (low, high dan tri-state). Adanya si nyal aktif menunjukkan bahwa address bus sedang berisi alamat yang valid untuk suatu proses read/write memori.

$\overline{IORQ}$

Input/Output Request, berfungsi sebagai *input/output, act ive low* dan dapat memiliki 3 kondisi (*low, high dan tri-sta te*). Adanya sinyal aktif menunjukkan bahwa A7 - A0 dan *address bus* sedang berisi alamat I/O yang valid untuk m elakukan operasi *Read* atau *Write* pada peranti I/O terse but. Sinyal juga dihasilkan secara bersamaan dengan sin yal pada saat suatu pemintaan *interrupt* telah diterima ol eh CPU untuk menunjukkan bahwa instruksi berikut yang merupakan *interrupt vector* boleh ditempatkan di *address bus*.

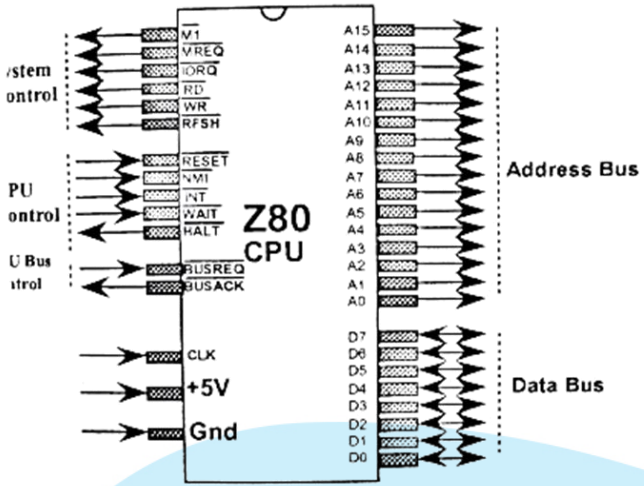
Fungsi Pin Z80



$\overline{\text{BUSREQ}}$ *Bus request*, berfungsi sebagai input *active low*. Bila rang kaian luar menginginkan agar suatu saat CPU bus berad a dalam keadaan *tri-state* dengan maksud akan menggun akan bus tersebut untuk suatu keperluan, maka *bus requ est* dapat digunakan. Dengan mengaturnya dalam keada an *low* maka CPU bus. yaitu *address bus*, *data bus*, $\overline{\text{RD}}$, $\overline{\text{WR}}$, $\overline{\text{WREQ}}$ dan $\overline{\text{IORQ}}$ akan *tri-state*. Bila telah *tri-state*, CP U akan memberikan sinyal $\overline{\text{BUSACK}}$

$\overline{\text{BUSACK}}$ *Bus acknowledge*, berfungsi sebagai *output active low*. *B us acknowledge* berfungsi memberitahu kepada rangkaia n tertentu yang sebelumnya meminta kepada CPU untuk melakukan sesuatu bahwa CPU *address bus*, CPU *addr ess bus* dan sinyal-sinyal kontrol $\overline{\text{RD}}$. $\overline{\text{WR}}$ $\overline{\text{MREQ}}$ dan $\overline{\text{IORQ}}$ telah dalam keadaan *high-impedance* (*tri-state*), untuk sel anjutnya rangkaian tertentu tersebut dapat mengontrol bu s-bus untuk digunakan.

Fungsi Pin Z80



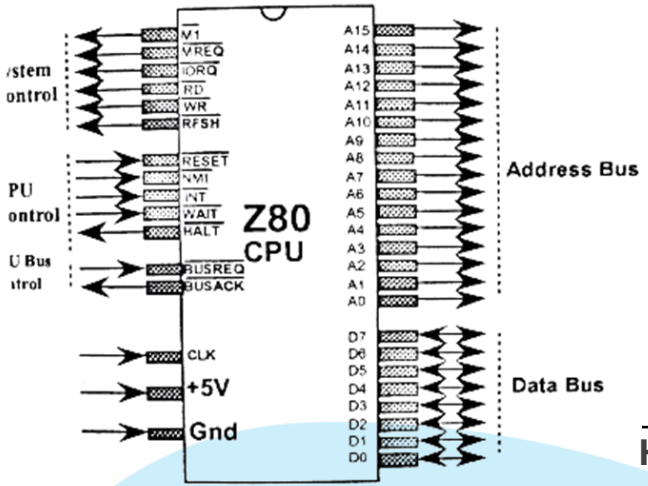
$\overline{\text{INT}}$

Interrupt Request, berfungsi sebagai input, *active low*. Bila peralatan I/O membutuhkan suatu fasilitas agar pada suatu saat dapat melakukan suatu interrupt kepada CPU (interupsi terhadap proses yang sedang dikerjakan), maka ia dapat menggunakan kontrol $\overline{\text{INT}}$ ini. Bila $\overline{\text{INT}}$ diaktifkan maka CPU akan melakukan interupsi terhadap program yang sedang dijalankan dan akan menuju ke suatu program yang telah didefinisikan sesuai dengan pemberian instruksi $\overline{\text{INT}}$. Alamat awal program rutin *Interrupt* ini disebut sebagai *Interrupt Vector.*

$\overline{\text{NMI}}$

Non-Maskable Interrupt. berfungsi sebagai input, negative edge-triggered. $\overline{\text{NMI}}$ memiliki prioritas yang lebih tinggi dari pada $\overline{\text{INT}}$. Maksudnya bila pertama diberikan $\overline{\text{INT}}$ kemudian $\overline{\text{NMI}}$, maka CPU akan memperhatikan $\overline{\text{NMI}}$ meskipun proses yang dikehendaki oleh $\overline{\text{INT}}$ belum selesai. Namun, bila diberikan $\overline{\text{NMI}}$ terlebih dahulu dan kemudian $\overline{\text{INT}}$, maka $\overline{\text{INT}}$ harus menunggu proses yang dikehendaki $\overline{\text{NMI}}$ selesai terlebih dahulu baru $\overline{\text{INT}}$ akan diperhatikan. Pada setiap pemberian sinyal $\overline{\text{NMI}}$, CPU akan melakukan proses jump ke alamat 0066H.

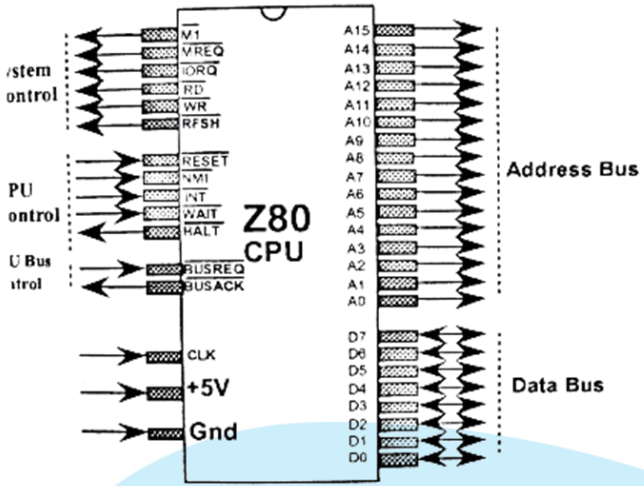
Fungsi Pin Z80



RESET *Reset*, berfungsi sebagai input, *active low*. Mengaktifkan RESET berarti menginisialisasi CPU kembali ke posisi: PC diset 0000H, semua register termasuk I dan R di-*clear*-kan dan *interrupt status* diset dalam mode 0. Selama *reset* terjadi, *address bus* dan *data bus* berada dalam kondisi *high-impedance* (*tri-state*), sedang semua sinyal kontrol *output* berada dalam keadaan tidak aktif (*inactive*). Untuk mendapatkan hasil *reset* yang lengkap, sinyal RESET harus diberikan minimal selama 3 siklus *clock* secara penuh.

HALT *Halt State,* berfungsi sebagai *output, active low.* Sinyal HALT yang muncul menunjukkan bahwa CPU telah mendapat perintah HALT dari program untuk menghentikan proses kerjanya sementara, sampai ia mendapat sinyal *interrupt,* baik NMI maupun INT . Selama menunggu sinyal *interrupt*, CPU menjalankan instruksi NOP (*no operation*) secara berulang untuk menjaga agar memori *(dynamic* RAM, jika memakai) tetap valid.

Fungsi Pin Z80



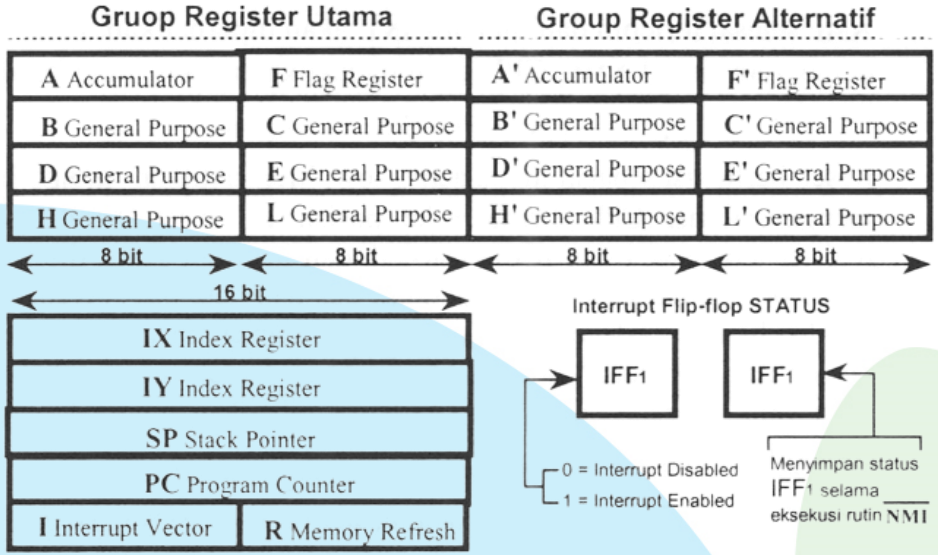
$\overline{\text{MI}}$ *Machine Cycle One*, berfungsi sebagai *output, active low*. bersama sama dengan $\overline{\text{RFSH}}$ menunjukkan bahwa siklus mesin (operasi) yang sedang berjalan pada saat itu adal ah *opCode fetch cycle* (siklus pengambilan Opcode) dari eksekusi sebuah instuksi.

$\overline{\text{RFSH}}$ *Refresh,* berfungsi sebagai *output, active low*. $\overline{\text{RFSH}}$ dan bersama sama menunjukkan bahwa konfigurasi A7—A0 pada *address bus* dapat digunakan sebagai *refresh addr ess* ke sistem memori dinamik.

$\overline{\text{WAIT}}$ *Wait,* berfungsi sebagai *input, active low.* Bila aktif berarti CPU sedang dlberitahu bahwa memori atau peralatan I/O yang dialamati oleh CPU belum siap memberikan datany a untuk ditransfer. Bila ini muncul, CPU akan menunda p ekerjaannya, menunggu sampai kontrol tidak aktif..

Register Z80

CPU Z80 memiliki Read/Write memori yang disebut register sebanyak 280 bit, atau 26 x 8 bit. Memori di dalam CPU ini sangat berguna bagi programmer da lam pembuatan program. Internal memori ini dibagi dalam 18 buah register 8 -bit dan 4 buah register 16 bit, seperti ditunjukan pada Gambar 5.4.



Gambar 9.4 Register-register CPU Z80

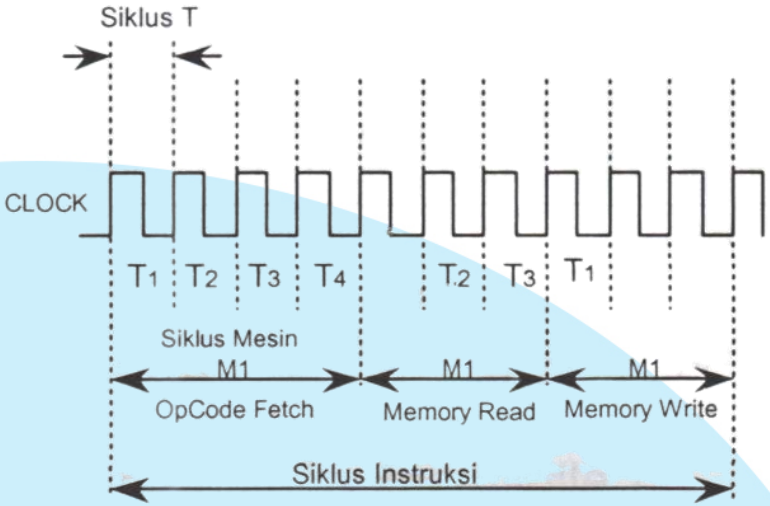
Pewaktuan Read/Write Memori Mikroprosessor Z80

PERTEMUAN / MATERI KE 10

Antarmuka dan Kamputer

SISTEM TIMING CPU Z80

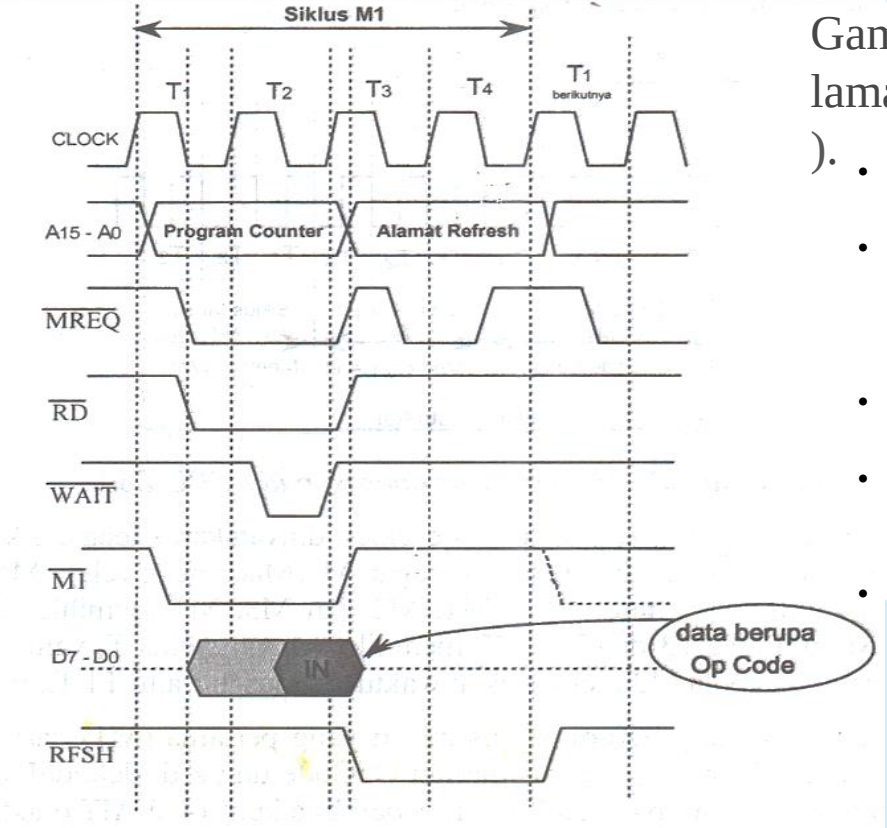
Seperti ditunjukan pada Gambar 10.1., sebuah periode *clock* dinyatakan sebag ai siklus T, sedangkan operasi dasar dinyatakan sebagai M (*Machine Cycle*). *M achine cycle* atau siklus mesin terdiri dari M1, M2 dan M3. M1 memiliki 4 bua h siklus T, yaitu T1, T2, T3 dan T4. M2 memiliki 3 buah siklus T, yaitu T1, T2 da n T3. Demikian pula M3 memerlukan waktu 3 siklus T, yaitu T1, T2 dan T3.



Gambar 10.1 Timing dasar operasi pada CPU Z80

Dalam operasi program, siklus mesin ya ng pertama (M1) dari setiap instruksi a dalah *fetch cycle* (pengarnbilan OpCode untuk di-dekode. Siklus ini membutuhk an 4 sampai 6 siklus, terkecuali bila kon trol $\overline{\text{WAIT}}$ diaktifkan, siklus dapat diperp anjang.

PEWAKTUAN OPCODE FETCH

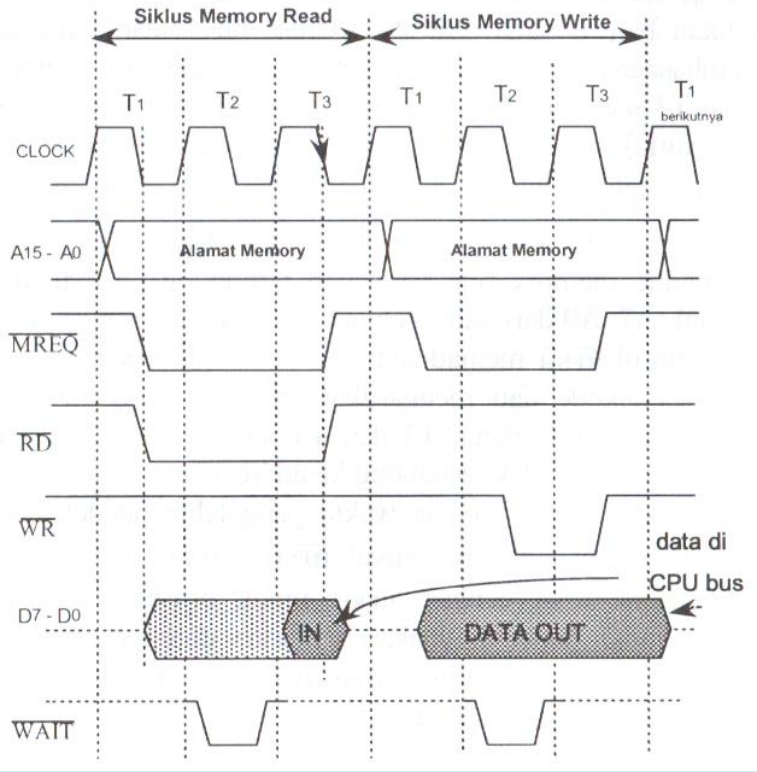


Gambar 10.2 Pewaktuan siklus *OpCode fetch*

Gambar 10.2 menunjukkan diagram pewaktuan selama siklus mesin MI (*Operation Code fetch cycle*).

- Pertama, siklus MI address bus berisi data yang tersimpan di Program Counter (PC).
- Setengah *clock cycle* berikutnya, sinyal MREQ menjadi aktif. Pada saat ini, langsung sebagai kontrol chip *enable* pada memori. Kontrol RD juga menjadi aktif yang digunakan untuk proses pembacaan data memori. CPU mengambil data dari memori
- Pada saat *rising edge* T3 dan pada waktu yang bersamaan pula CPU meng-inactivekan sinyal MREQ dan RD .
- Clock state T3 dan T4 digunakan untuk merefresh *dynamic memory*. Pada State ini, Å7—A0 dari *address bus* berisi alamat memori yang di-*refresh* dan sinyal kontrol RFSH menjadi åktif.
- Sinyal RD tidak aktif selama T3 dan T4 menjaga agar data dari segmen memori yang berbeda tidak terhubung ke *address bus*. Selama *refresh*, sinyal MREQ juga aktif dengan selang waktu yang lebih pendek dan berada di tengah-tengah Waktu aktif dari sinyal MREQ ini biasanya juga di gunakan bersama-sama dengan RFSH.

PEWAKTUAN MEMBACA MEMORI

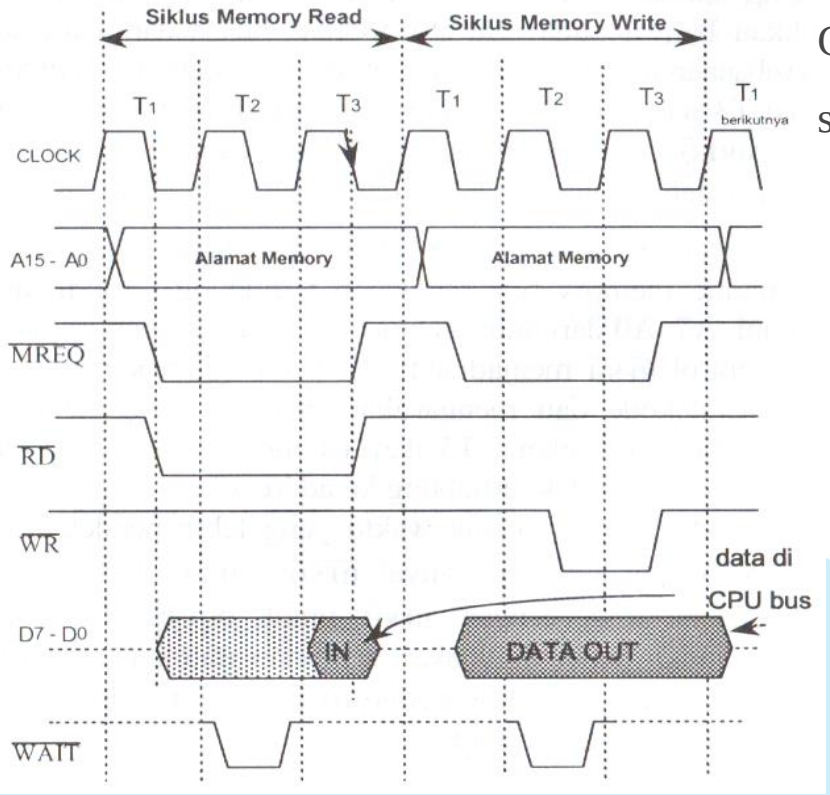


Gambar 10.3 Pewaktuan siklus *OpCode fetch*

Gambar 10.3 menunjukkan diagram pewaktuan dari siklus Read pada CPU Z80. Pada umumnya terdiri dari 3 buah si klus T, Kecuali WAIT diaktifkan.

- Saat *Write cycle* T1, T2, T3 alamat memori yang akan di tulis, mulai dilet akkan di *address bus.* Begitu rising edge T1 terjadi, alamat memori yang akan dituju diletakkan di *address bus.*
- Begitu T1 mengalami rising edge, $\overline{\text{MREQ}}$ dan $\overline{\text{RD}}$ bersama-sama aktif be gitu T1 menuju ke 0 pada Sisi *falling edge*. *Wait state* dapat ditambahkan dengan mengaktifkan $\overline{\text{WAIT}}$ pada *falling edge* pulsa clock berikutnya, da lam hal ini adalah T2.
- Data yang akan dibaca berada di address bus pada saat siklus berada di T 3. Data ini diambil oleh CPU berdasarkan falling edge dari T3. Berdasark an *falling edge* ini pula, sesaat kemudian $\overline{\text{MREQ}}$ dan $\overline{\text{RD}}$ menjadi *inactive* kembali.

PEWAKTUAN MENULIS MEMORI

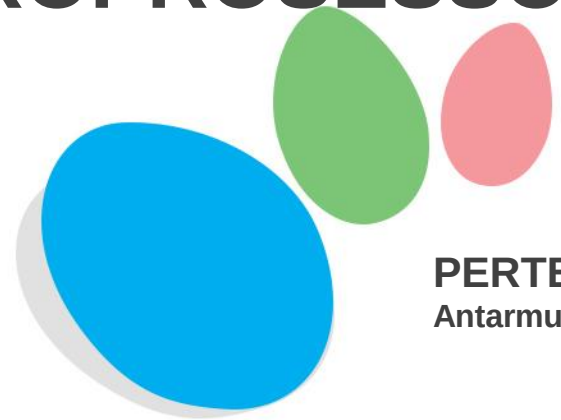


Gambar 10.4 Pewaktuan siklus *OpCode fetch*

Gambar 10.4 menunjukkan diagram pewaktuan dari siklus Write pada CPU Z80.

- Pada *Write cycle* memiliki T1, T2 dan T3. Alamat memori yang akan di tulis mulai diletakkan di address bus begitu *rising edge* T1 terjadi.
- Berdasarkan *falling edge* T1 $\overline{\text{MREQ}}$ menjadi aktif. Begitu $\overline{\text{MREQ}}$ aktif, data bus akan diisi data oleh CPU. Data ini adalah data yang akan dituliskan oleh CPU ke memori.
- Wait state dapat ditambahkan dengan mengaktifkan pada saat *rising edge* T2 telah stabil. Sinyal aktif begitu *falling edge* T2 terjadi. Pada saat inilah rangkaian memori mengambil data yang telah ada di *address bus*. Berikumya, berdasarkan *falling edge* T3, sinyal $\overline{\text{MREQ}}$ dan $\overline{\text{WR}}$ menjadi *inactive* kembali.

PEWAKTUAN
INPUT/OUTPUT dan READ/WRITE
MIKROPROSESSOR Z80



PERTEMUAN / MATERI KE 11

Antarmuka dan Kamputer

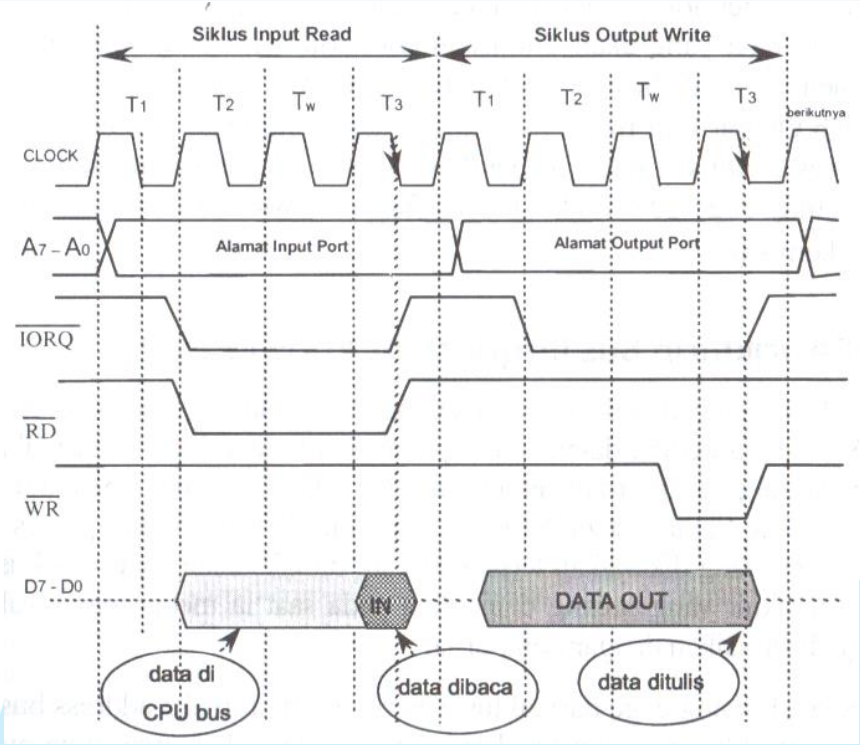
PEWAKTUAN INPUT/OUTPUT READ/WRITE

Pada operasi Input read maupun *output write*, secara otomatis sebuah *wait state* (Tw) ditambahkan setelah T2.

Alasannya adalah :

Waktu yang tersedia dari sinyal $\overline{\text{IORQ}}$, $\overline{\text{RD}}$ atau $\overline{\text{WR}}$ (aktifnya sinyal $\overline{\text{IORQ}}$ sampai *falling edge* T3) untuk menjadi inactive kembali, tidak cukup untuk membuat data yang ditransfer melalui *address bus* menjadi se mpurna atau siap untuk ditransfer.

PEWAKTUAN MEMBACA MASUKAN

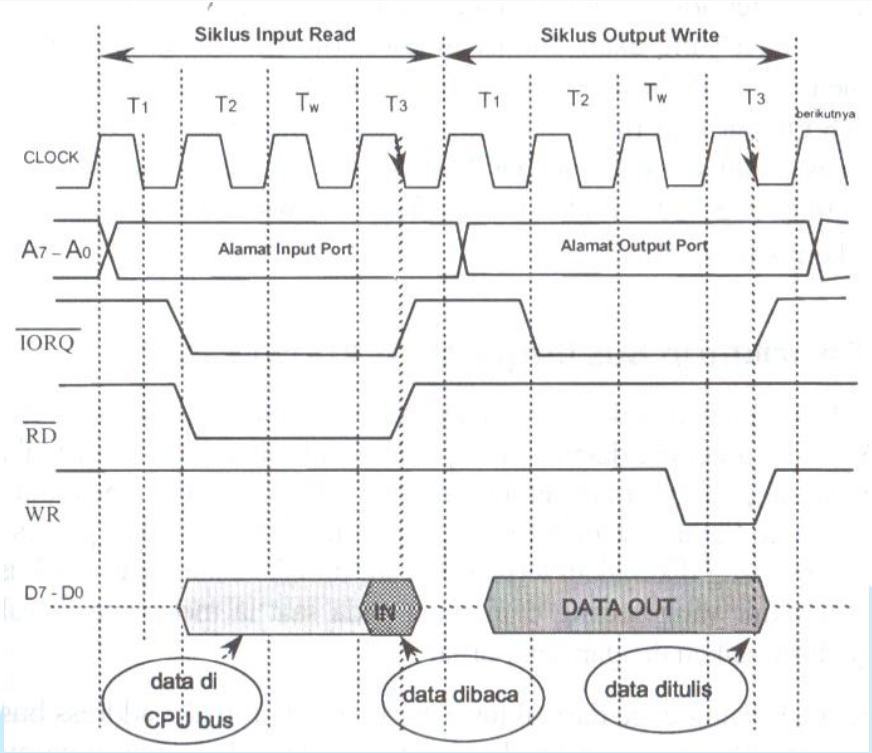


Gambar 11.1 Pewaktuan siklus *OpCode fetch*

Gambar 11.1 Menunjukkan pewaktuan dari operas i *Input Read* dan *Output Write*

- Pada Tl , begitu *rising edge* terjadi, *address bus* diisi dengan alam at port (terminal) yang dituju untuk dibaca datanya. Tl hanya dig unakan untuk mengaktifkan *address bus*.
- Berdasarkan *rising edge* T2, sinyal $\overline{\text{IORQ}}$ dan $\overline{\text{RD}}$ bersama-sama menjadi aktif untuk mengaktifkan kontrol-kontrol peripheral I/O, misalnya semacam (*Chip Select*).
- Data yang akan dibaca oleh CPU diletakkan di data bus pada saat *rising edge* T2, berdasarkan *falling edge* T3, data tersebut diambi l oleh CPU.
- Sesaat kemudian, sinyal $\overline{\text{IORQ}}$ dan $\overline{\text{RD}}$ menjadi *inactive* kembali.

PEWAKTUAN PENULISAN PADA UNIT KELUARAN



Gambar 11.2 Pewaktuan siklus *OpCode fetch*

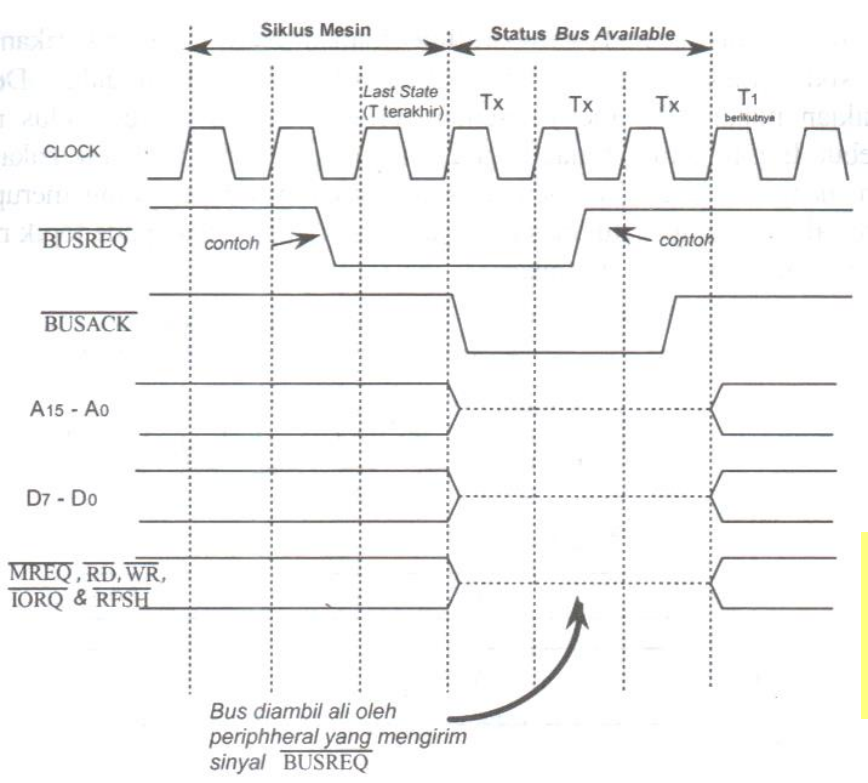
Gambar 11.2 menunjukkan pewaktuan dari operasi *I nput Read* dan *Output Write* (lajutan)

- Proses *Output write* dimulai dengan diisinya address bus dengan alamat *Output Port* yang akan dituju, untuk diisi dengan suatu da ta begitu *rising edge* Tl terjadi.
- Setelah Tl mengalami *falling edge*, CPU mengisi data bus denga n data yang akan dituliskan.
- Berdasarkan *rising edge* T2, sinyal $\overline{\text{IORQ}}$ menjadi aktif. Sesaat k emudian, sinyal $\overline{\text{WR}}$ juga menjadi aktif. pada saat inilah *Output Port* mulai mengambil data yang telah tersedia di *address bus*.
- Proses penulisan data ini berakhir begitu T3 mengalami *falling e dge*. Begitu *falling edge* T3 terjadi, sinyal $\overline{\text{IORQ}}$ dan $\overline{\text{WR}}$ bersam a-sama menjadi *inactive* kembali.

PEWAKTUAN BUS REQUEST/ACKNOWLEDGE

Bila Z80 sedang berada dalam suatu proses kemudian diaktifkan maka Z80 akan mela ksanakan permintaan untuk membuat CPU bus menjadi *high impedance* atau *tri-state* setelah *rising edge* siklus T terakhir dari proses siklus mesin yang sedang dikerjakan t erjadi. Hal ini dimaksudkan agar eksekusi instruksi terakhir yang sedang dijalankan p ada saat ia menerima sinyal aktif $\overline{\text{BUSREQ}}$ diselesaikan dengan sempurna.

PEWAKTUAN BUS REQUEST/ACKNOWLEDGE



Gambar 11.3 Pewaktuan siklus $\overline{\text{BUSREQ}}$ (*Bus Reques*) dan $\overline{\text{BUSACK}}$ (*Bus Acknowledge*)

Gambar 11.3 memperlihatkan pewaktuan dari siklus *b us request/Aclmowledge*.

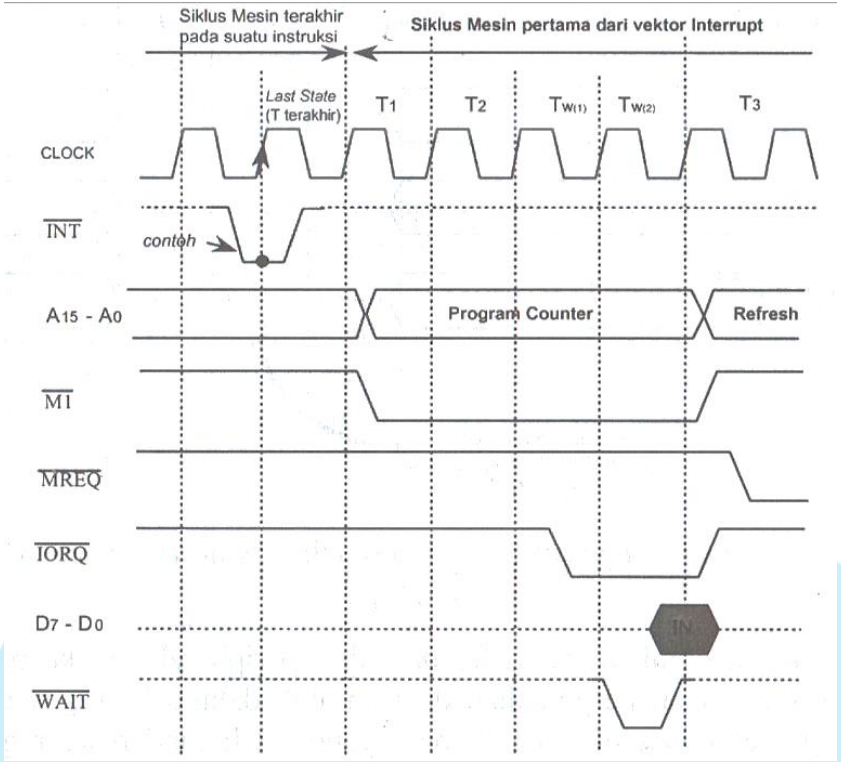
Setelah rising edge dafi siklus T berikut terjadi maka address bus, data bus dan control bus yang terdiri dari kontrol-kontrol bersama-sama menjadi tri-state. Sebagai tanda dari CPU ke p eripheral yang meminta (meminjam) bus (sinyal $\overline{\text{BUSACK}}$ menjadi aktif). Pada Saat ini, bus dapat dipakai oleh rangkaia n luar untuk proses kerja yang dikehendaki.

Umumya, fasilitas seperti ini dimanfaatkan oleh rangkaian luar untuk me lakukan proses transfer data secara langsung antar-peripheral tanpa mem inta pertolongan CPU.

PEWAKTUAN SIKLUS INTERRUPT REQUEST/ACKNOWLEDGE

Seperti pada pengaktifkan sinyal $\overline{\text{BUSREQ}}$, permintaan akan di laksanakan oleh CPU, setelah siklus T dari siklus mesin yang se dang dijalankan berakhir.

PEWAKTUAN SIKLUS INTERRUPT REQUEST/ ACKNOWLEDGE



Gambar 11.4 Pewaktuan siklus *Interrupt* (INT)

Gambar 11.4 memperlihatkan pewaktuan bila sinyal kontrol $\overline{\text{INT}}$ diaktifkan.

- Diperlihatkan bahwa $\overline{\text{INT}}$ diberikan pada saat siklus mesi n berada di permulaan siklus T yang terakhir, dengan de mikian CPU menyelesaikan terlebih dahulu proses siklus mesin tersebut.
- Begitu berakhir (*rising edge* T1), maka CPU akan mengisi *address bus* dengan suatu program counter yang merup akan vektor alamat.

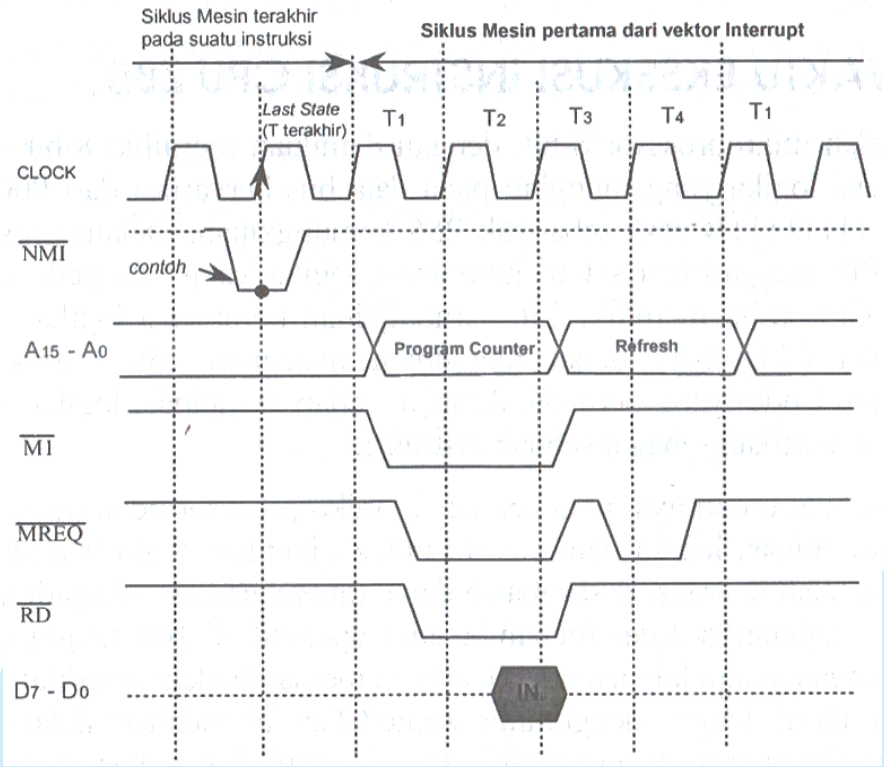
PEWAKTUAN SIKLUS NON MASKABLE INTERRUPT (NMI)

Sinyal kontrol $\overline{NMI}$ ini memunyai prioritas yang lebih tinggi da ri pada sinyal kontrol, tetapi masih lebih rendah satu tingkat dari sinyal kontrol RESET.

Catatan:

- Permintaan $\overline{NMI}$ akan dilaksanakan terlebih dahulu sampai tuntas, baru kemudian $\overline{INT}$.
- Suatu contoh jika $\overline{INT}$ diberikan terlebih dahulu, kemudian diikuti pemberian $\overline{NMI}$, maka meskipun CPU belum menyelesaikan tugas yang diminta oleh $\overline{INT}$. CPU akan menunda pe kerjaannya dan melaksanakan tugas yang diminta oleh $\overline{NMI}$

PEWAKTUAN SIKLUS NON MASKABLE INTERRUPT (NMI)



Gambar 11.5 Pewaktuan siklus *Non-Maskable Interrupt* (NMI)

Gambar 11.5 memperlihatkan pewaktuan dari proses *request/aclmowledge cycle*, untuk sinyal interrupt ya ng berasal dari $\overline{\text{NMI}}$.

- $\overline{\text{NMI}}$ diberikan pada saat yang sama seperti contoh pada pemberian sinyal.
- Satu siklus T berikutnya, CPU baru melaksanakan permintaan $\overline{\text{NMI}}$ dengan mengisi address bus dengan alamat 0066H, atau PC diset 00 66H. Bersamaan itu pula, sinyal $\overline{\text{NMI}}$ menjadi aktif.
- Begitu Tl mengalami falling edge maka sinyal $\overline{\text{MREQ}}$ dan $\overline{\text{RD}}$ menj adi aktif. Pada saat inilah data yang berada di alamat memori yang dihubungi dengan alamat 0066H, diambil oleh CPU untuk didekode dan dieksekusi.
- Selama T3 dan T4, CPU mengaktifkan $\overline{\text{MREQ}}$ dan $\overline{\text{RFSH}}$, untuk me lakukan memori *refresh*.


UNIVERSITAS PGRI SEMARANG
TERIMA KASIH