8. DE RAM-I/O-TIMER CHIP 8155

Evenals de 8355 is ook de RAM-I/O-timer chip 8155 ontwikkeld voor toepassing in 8085-systemen. De 8155 is nl. volledig aangepast aan de gemultiplexte data/adresbus. Verder werkt de 8155 met dezelfde status- en besturingssignalen als de 8085.

Vraag 15: In de 8155 bevinden zich ..... bytes RAM, ... I/O-poorten met samen ..... I/O-lijnen en een ....-bits timer.

In de 8155 bevindt zich een RAM-gedeelte met een capaciteit van 256 bytes, 2 I/O-poorten met elk 8 I/O-lijnen, een I/O-poort met 6 I/O-lijnen en een 14-bits timer.
Al deze mogelijkheden en het feit, dat vrijwel geen extra logica nodig is om de 8155 op een 8085-bussysteem aan te sluiten, maken de 8155 bijzonder geschikt voor toepassing in kleine systemen.

In fig.7 zijn de aansluitmogelijkheden van deze chip weergegeven.

8155

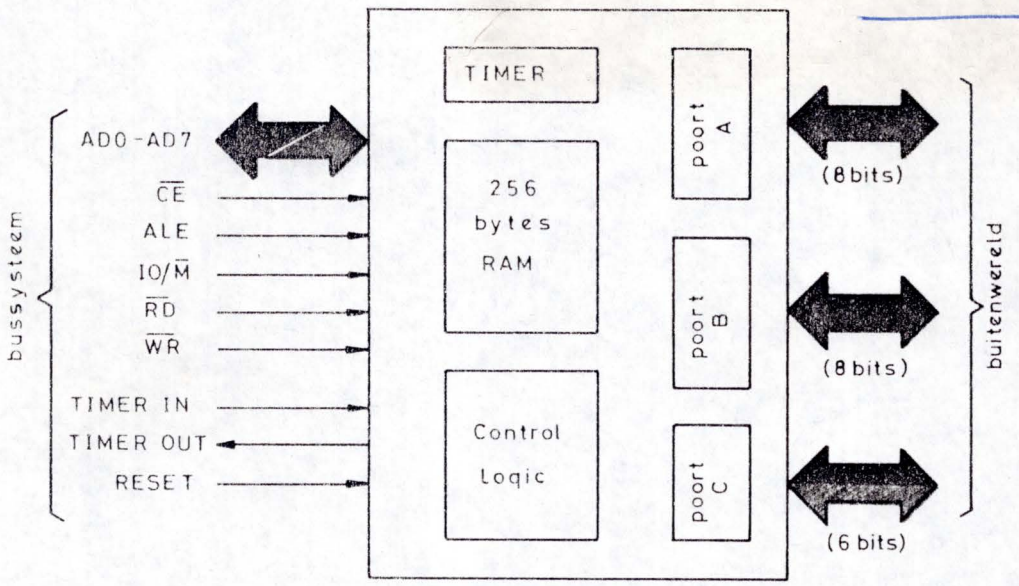

fig.7

We bespreken nu achtereenvolgens
a. Control logic (paragraaf 9)
b. RAM-gedeelte (paragraaf 10)
c. I/O-gedeelte (paragraaf 11)
d. Timer-gedeelte (paragraaf 12).

Daarna bespreken we op welke wijze de basic 8155 (paragraaf 13) en de expansion 8155 (paragraaf 14) in de SDK 85 zijn opgenomen.

 Antw.15: 256; 3; 22; 14.

9. CONTROL LOGIC (8155)

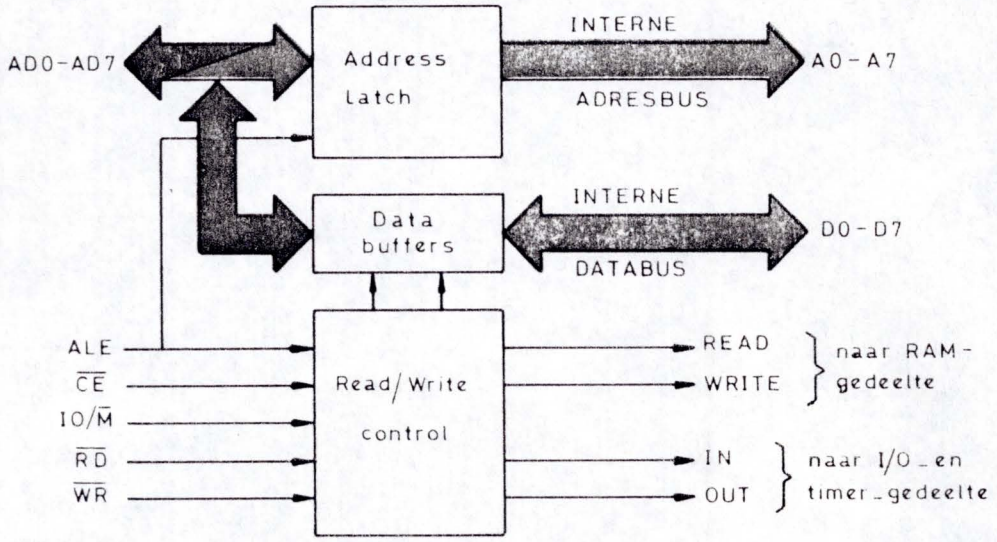

fig.8

In fig.8 is een gedetailleerder blokschema van de control logic weergegeven.

De address latch en de databuffers functioneren op dezelfde wijze als in de 8355. De interne adresbus is nu echter 8 bits breed.

Merk op, dat de 8155 geen READY-signaal afgeeft. Gewoonlijk is dit geen bezwaar, omdat de snelheid van de 8185 op die van de 8085 (waarvoor de chip is ontworpen) is aangepast.

De read/write control geeft de besturingssignalen voor de overige delen van de 8155 af. Hierbij spelen vooral de signalen op $\overline{CE}$, IO/$\overline{M}$, $\overline{RD}$ en $\overline{WR}$ een rol (zie tabel 3).

$\overline{CE}$	IO/$\overline{M}$	$\overline{RD}$ of $\overline{WR}$	afgegeven
1	X	X	-
0	0	$\overline{RD}$	READ
0	0	$\overline{WR}$	WRITE
0	1	$\overline{RD}$	IN
0	1	$\overline{WR}$	OUT
X = don't care. - = geen actief signaal aanwezig.			

Tabel 3

De 8155 wordt alleen geactiveerd als geldt $\overline{CE}$ = 0. Omdat er geen aparte ingangen zijn voor input- en output-signalen (zoals $\overline{IOR}$ en $\overline{IOW}$ bij de 8355) kunnen het I/O-gedeelte en de timer alleen volgens I/O mapped I/O worden geadresseerd.

10. RAM-GEDEELTE (8155)

Vraag 16: Het RAM-gedeelte heeft een capaciteit van ..... bytes.
Hiervoor zijn ..... adreslijnen noodzakelijk.

Het RAM-gedeelte van de 8155 bevat 256 geheugenwoorden van 8 bits. Om deze 256 locaties te kunnen adresseren zijn 8 adreslijnen nodig ($2^8 = 256$). Dit zijn de lijnen A0 t/m A7 van de interne databus.

Omdat er geen READY-signaal wordt afgegeven, moet de access-tijd van de 8155 aangepast zijn aan de snelheid van de CPU. Als de access-tijd voor een bepaalde CPU te lang duurt, kunnen we één van de volgende oplossingen kiezen:

a. De klokfrequentie van de CPU verlagen. Dit houdt wel in, dat elke state langer gaat duren, zodat de gehele programma-uitvoering wordt vertraagd.
b. D.m.v. een extra schakeling, die steeds wanneer de 8155 wordt geselecteerd, na een bepaalde tijd (minimaal de access-tijd) een READY-signaal genereert.

SAMENVATTING 5

18. De RAM-I/O-timer chip 8155 bevat
    a. 256 bytes RAM.
    b. 2 8-bits I/O-poorten.
    c. 1 6-bits I/O-poort.
    d. 1 14-bits timer.
    e. control logic.

19. De control logic bestaat uit
    a. een demultiplexer latch voor de data/adresbus.
    b. bi-directionele tri-state buffers t.b.v. de databus.
    c. read/write logic voor het opwekken van de interne besturingssignalen.

20. De 8155 geeft geen READY-signaal af.

11. I/O-GEDEELTE (8155)

Binnen het I/O-gedeelte van de 8155 komen 4 locaties voor, die door de gebruiker kunnen worden geadresseerd.
In fig.9 zijn de adresdecoder en I/O-control weergegeven. Deze twee schakelingen zorgen voor de juiste besturingssignalen voor de 4 locaties (tabel 4).

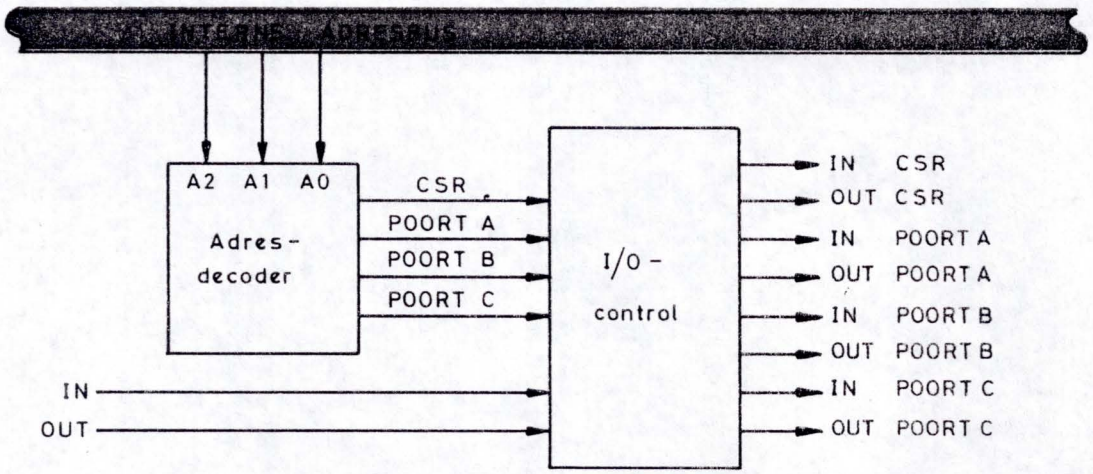

fig.9

A2	A1	A0	IN of OUT	afgegeven
0	0	0	IN	IN CSR
0	0	0	OUT	OUT CSR
0	0	1	IN	IN POORT A
0	0	1	OUT	OUT POORT A
0	1	0	IN	IN POORT B
0	1	0	OUT	OUT POORT B
0	1	1	IN	IN POORT C
0	1	1	OUT	OUT POORT C
1	X	X	X	-
X	X	X	-	-

X = don't care.
\- = geen actief signaal aanwezig.

Tabel 4

Vraag 17: Er wordt alleen een besturingssignaal afgegeven als A2 = 0/1.

Uit tabel 4 blijkt, dat er alleen een besturingssignaal wordt afgegeven als A2 = 0. Wanneer er zich nl. I/O-adressen met A2 = 1 op de databus bevinden, dan wordt een locatie binnen het timer-gedeelte geselecteerd.

Antw.17: 0.

In fig.10 is het blokschema van het I/O-gedeelte weergegeven.
T.b.v. de overzichtelijkheid zijn de besturingslijnen tussen I/O-control, command status register en de 3 I/O-poorten niet alle getekend.

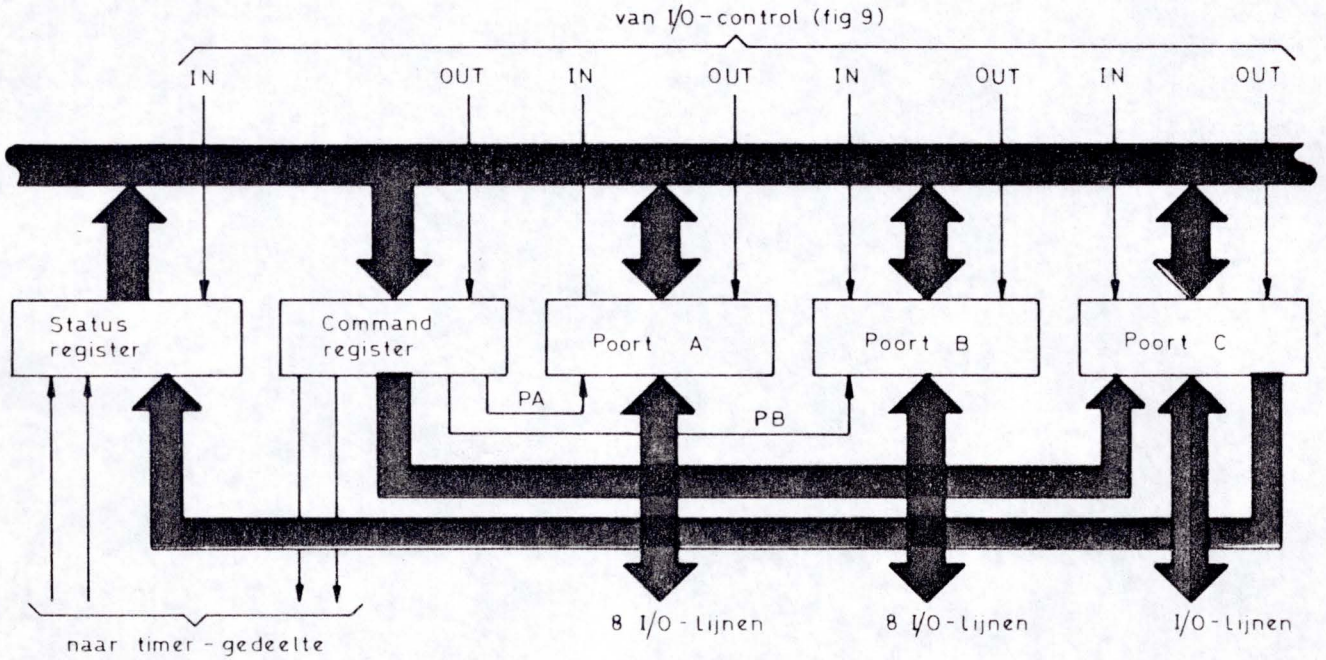

fig.10

We zullen nu achtereenvolgens behandelen
a. Command status register.
b. Strobed I/O.
c. Poorten A en B.
d. Poort C.

a. Command status register

Vraag 18: Om het command status register te selecteren, moet gelden CE = 0/1; IO/$\overline{M}$ = 0/1; A2 = 0/1; A1 = 0/1 en A0 = 0/1.

Het command status register wordt m.b.v. input- en output-instructies aangesproken. Als we het juiste I/O-adres op de adresbus plaatsen, geldt $\overline{CE}$ = 0 (de 8155 wordt dus geactiveerd) en IO/$\overline{M}$ = 1 (het I/O-gedeelte binnen de 8155 wordt geselecteerd). Het juiste I/O-adres is XXXXX000 (zie tabel 4). Hierin zijn A7 t/m A3 don't cares (zie fig.9).

Het command status register is in twee delen te splitsen, nl. het command register en het status register.

Als we het CSR d.m.v. een output-instructie aanspreken, dan wordt het command register (fig.11) geselecteerd. Dit command register is alleen

Antw.18: 0; 1; 0; 0; 0.

met een bepaalde bitcombinatie te vullen (is dus niet uit te lezen) en dient voor het programmeren van het I/O-gedeelte en de timer.

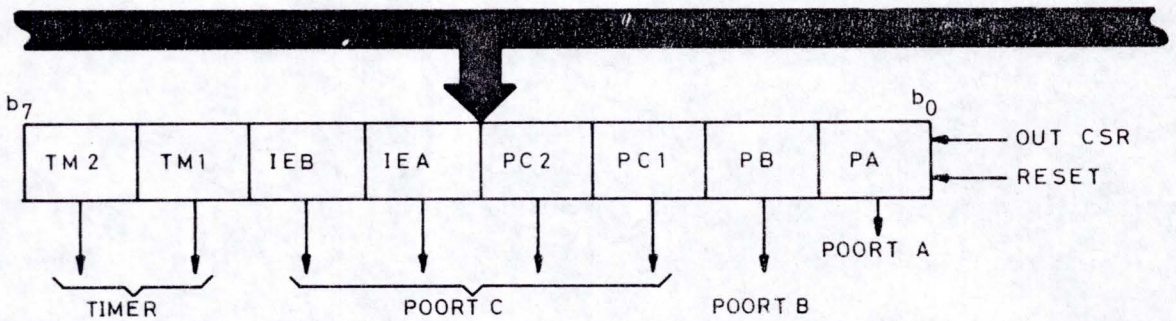

fig.11

Met b_0 en b_1 kunnen we de I/O-poorten A en B als volgt initialiseren.
PA = 0: poort A is input.
PA = 1: poort A is output.
PB = 0: poort B is input.
PB = 1: poort B is output.

Met b_2 t/m b_5 programmeren we poort C. Hierop gaan we in paragraaf 11d dieper in.

Met b_6 en b_7 laten we de timer starten en stoppen. Dit wordt in paragraaf 12 behandeld.

Stel, we willen poort A als input-poort en poort B als output-poort initialiseren.
b_2 t/m b_7 van het command register moeten 0 zijn.

Vraag 19: Het command register moet dan worden gevuld met $\ldots\ldots_{16}$.

Er moet dan gelden PA = 0 en PB = 1. Het command register moet dan worden gevuld met $00000010_2 = 02_{16}$.

Door een impuls op de RESET-ingang van de 8155 wordt het command register gevuld met 01000000_2.

Vraag 20: Na een RESET-impuls geldt poort A is input/output en poort B is input/output.

Na een RESET-impuls zijn PA en PB beide 0, de poorten A en B zijn dan beide als input-poort geïnitialiseerd.

Als we het CSR d.m.v. een input-instructie aanspreken, dan wordt het status register (fig.12) geselecteerd. Dit status register kan alleen worden uitgelezen en dient om de status van het I/O-gedeelte en de timer naar de CPU over te brengen.

Antw.19: 02. Antw.20: input; input.

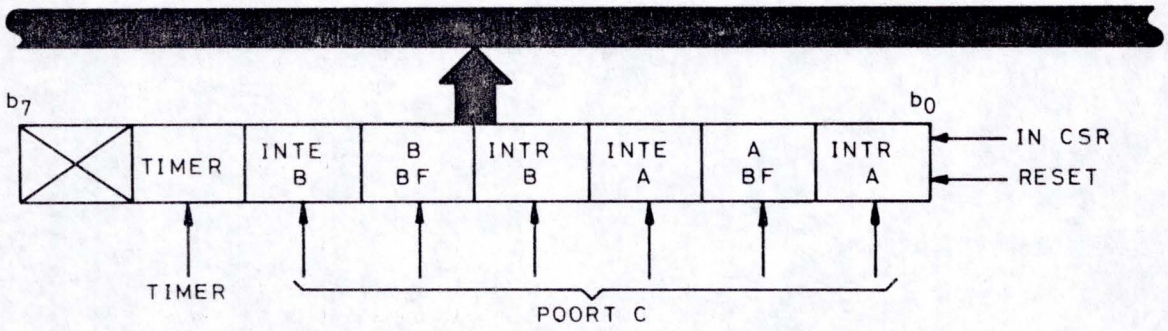

fig.12

b_0 t/m b_5 bevatten steeds informatie over de status van de I/O-poorten. De betekenis van deze bits wordt in paragraaf 11d besproken.

b_6 wordt 1 op het moment dat de timer in de 8155 tot nul is teruggeteld. b_6 wordt 0 (gereset) als het status register wordt uitgelezen of door een impuls op de RESET-ingang van de 8155.

Het uitlezen van het status register wordt voornamelijk toegepast bij geprogrammeerde I/O. De CPU behoeft nu niet een aantal afzonderlijke statuslijnen te testen, maar kan d.m.v. één enkele input-instructie de status van het gehele I/O-gedeelte naar binnen halen.

SAMENVATTING 6

21. In het I/O-gedeelte van de 8155 bevinden zich:
    adresdecoder,
    I/O-control,
    2 8-bits I/O-poorten
    1 6-bits I/O-poort
    command status register.

22. De adresdecoder en de I/O-control zorgen voor de juiste IN- en OUT-commando's voor de 3 I/O-poorten en het command status register.

23. Het command status register is te splitsen in command register en status register.

24. Het command register dient om het I/O-gedeelte en de timer in de 8155 te programmeren. Het command register kan alleen met een 8-bits combinatie worden gevuld.

25. Het status register dient om de status van het I/O-gedeelte en de timer uit te lezen t.b.v. geprogrammeerde I/O. Het status register kan alleen worden uitgelezen.

b. Strobed I/O

Voordat de poorten A, B en C kunnen worden behandeld, moet u iets weten over strobed I/O.

De gang van zaken bij strobed input is in fig.13 weergegeven.

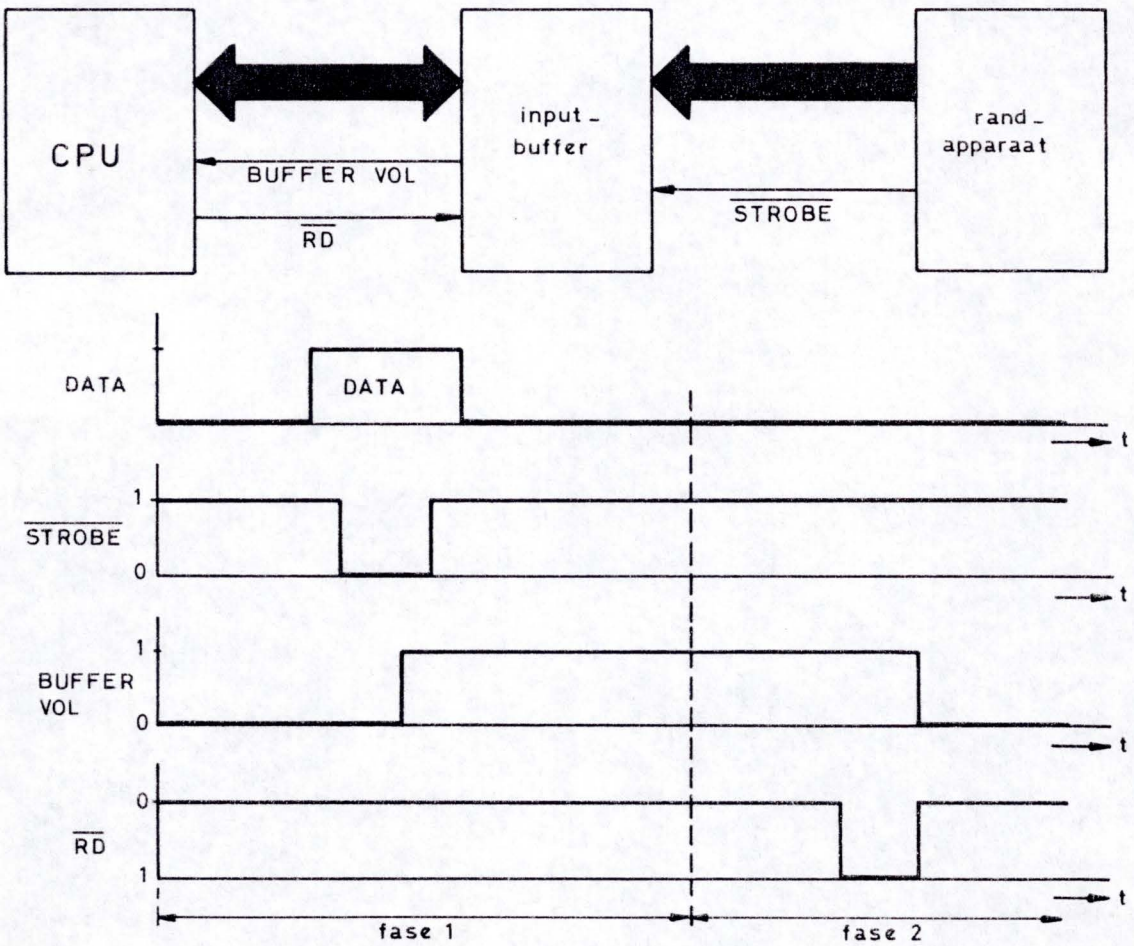

fig.13

Er zijn nl. randapparaten, die kortstondig data afgeven. Op het moment dat de data op de lijnen naar de I/O-module aanwezig is, wordt tevens een z.g. STROBE-signaal afgegeven. Nu zijn er voor het invoeren van de data twee oplossingen.

1. Geprogrammeerde I/O. De CPU moet, m.b.v. input-instructies, het STROBE-signaal continu testen. Is dit signaal actief (in fig.13 geldt 0 = actief), dan kan de data, weer met een input-instructie, worden ingevoerd.

Het testen van het STROBE-signaal kost echter veel tijd, die beter besteed kan worden aan de uitvoering van het eigenlijke programma in de microcomputer. Een nog groter probleem is, dat data en STROBE-signaal in de meeste gevallen veel te kort aanwezig zijn, om op bovengenoemde manier te kunnen worden verwerkt.
Als de CPU een actief STROBE-signaal heeft gedetecteerd, dan is op het moment, dat de data ingevoerd kan worden (enkele µs later) de data al vaak weer verdwenen. Daarom wordt meestal voor de volgende oplossing gekozen.

2. Strobed I/O. In de I/O-module is dan een input-buffer opgenomen (fig.13). Deze input-buffer bestaat uit 8 flip flops (tenminste bij 8-bits microcomputers), die op commando van het STROBE-signaal de data van het randapparaat opslaan. De bijbehorende besturingslogica geeft dan een actief BUFFER VOL-signaal aan de CPU af. Dit signaal blijft actief, totdat de CPU d.m.v. een leesopdracht (b.v. een input-instructie) de inhoud van de input-buffer heeft uitgelezen. Daarna is de input-buffer weer in staat nieuwe data van het randapparaat op te slaan, totdat de CPU deze weer kan overnemen.
De gehele actie is in twee fasen te verdelen, nl. fase 1, de hardware-actie, waarbij de data door het STROBE-signaal in de input-buffer wordt geklokt en fase 2, de software-actie, waarbij de CPU de inhoud van de input-buffer overneemt.

Strobed output via de 8155 verloopt volgens fig.14.

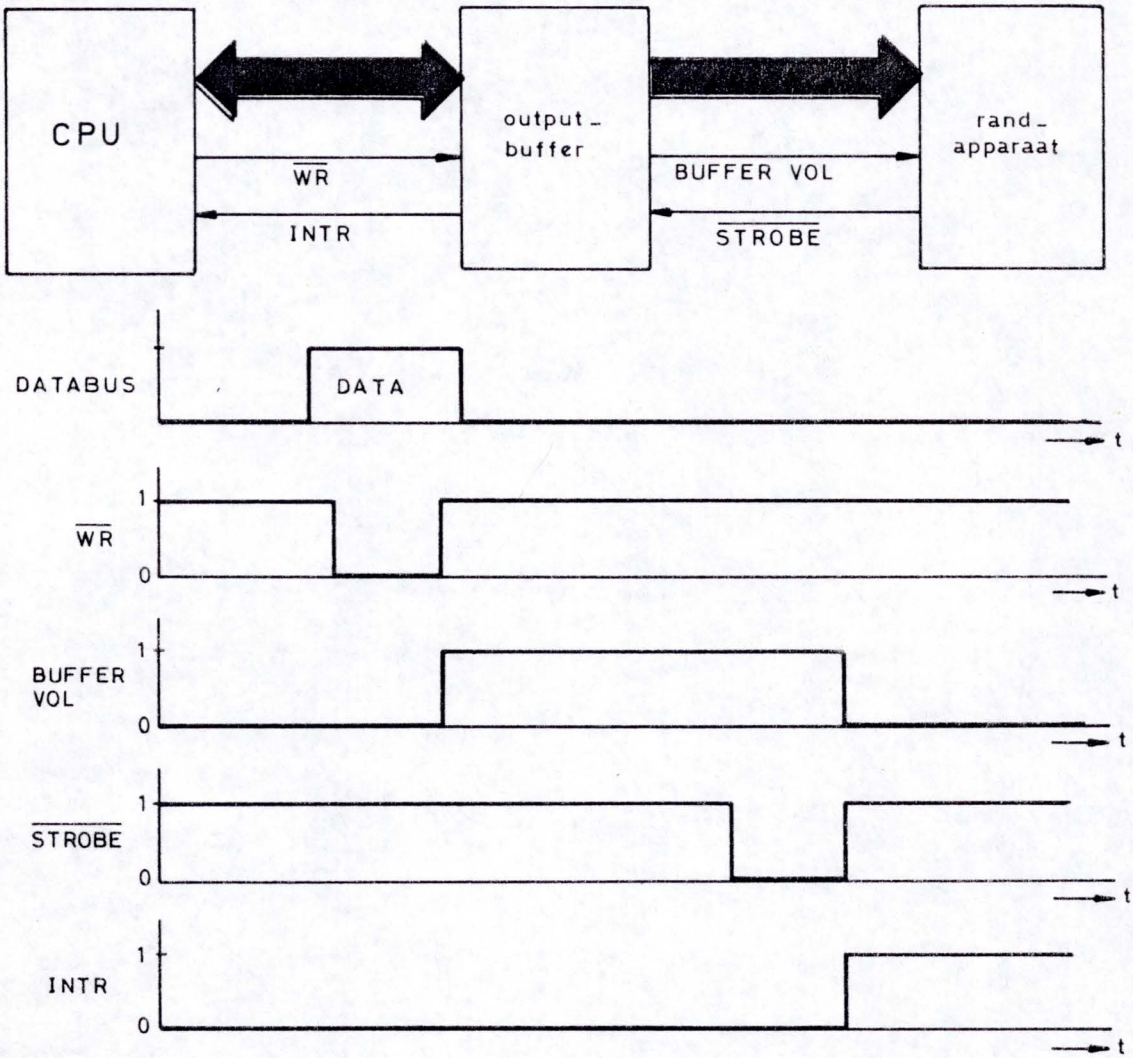

fig.14

De CPU plaatst d.m.v. een output-instructie een uit te voeren karakter in de output-buffer. Deze data staat dan continu op de datalijnen naar het randapparaat.
De bij de output-buffer behorende besturingslogica zendt dan een actief BUFFER VOL-signaal naar het randapparaat. Als deze de data heeft overgenomen, wordt een STROBE-signaal teruggestuurd.
De besturingslogica zal dan
1. het actieve BUFFER VOL-signaal wegnemen.
2. een interrupt request naar de CPU zenden.
Wat er daarna gebeurt, is afhankelijk van de instructies in de bijbehorende interrupt service routine. Er kan b.v. direct een nieuw karakter naar de output-buffer worden gezonden.

SAMENVATTING 7

26. Bij strobed I/O wordt m.b.v. een STROBE-signaal de transmissiesnelheid van de CPU aangepast aan die van een randapparaat, zonder dat er data verloren gaat (van een snel input-apparaat) of de CPU moet blijven wachten (op een traag output-apparaat) totdat er data kan worden uitgevoerd.

27. Bij strobed input klokt een input-apparaat d.m.v. het STROBE-signaal data in de input-buffer. Dan wordt een BUFFER VOL-signaal naar de CPU gezonden.

28. Bij strobed output plaatst de CPU data in de output-buffer. Er wordt dan een BUFFER VOL-signaal naar het output-apparaat gezonden. Als dit de data heeft overgenomen, wordt er een STROBE-signaal teruggestuurd, waardoor er een interrupt request naar de CPU wordt gezonden.

c. Poorten A en B

Fig.15 toont de schakeling voor b_5 van poort A. Deze schakeling komt in de 8155 dus 16 maal voor.

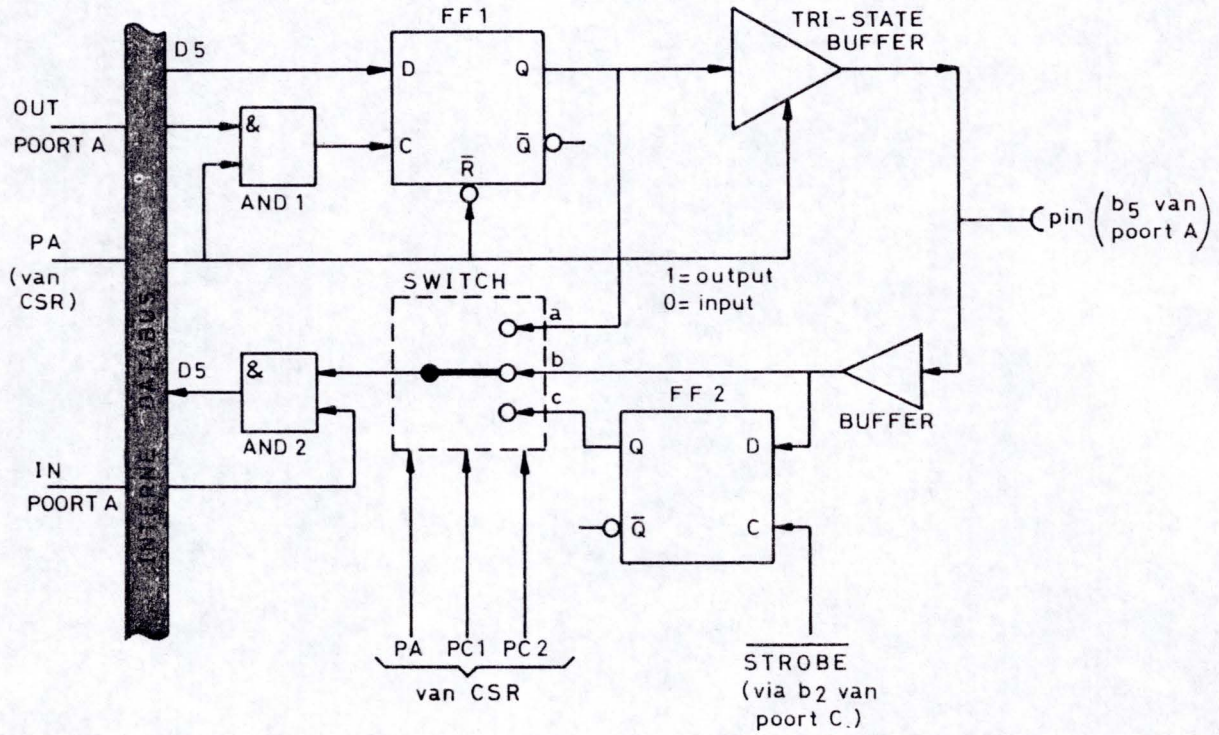

fig.15

In dit schema is FF1 b_5 van de output-buffer en FF2 b_5 van de input-buffer.
Poort A kan als input-poort en als output-poort werken, afhankelijk van PA (dit is b_0 van het command register, zie fig.11).
In beide gevallen kan dan m.b.v. PC1 en PC2 (b_2 en b_3 van het command register) wel of niet voor strobed I/O worden gekozen.
In totaal zijn er dus 4 modes, d.w.z. manieren, waarop poort A kan functioneren.

Input-poort (PA = 0; PC1 = PC2). Omdat geldt PA = 0, spelen FF1, AND1 en de tri-state buffer geen rol. Door de drie besturingssignalen wordt SWITCH (dit is te beschouwen als een elektronische schakelaar met 2 standen) in stand b gezet. FF2 doet dan ook niet mee. Voor dit geval kunnen we fig.15 dan vereenvoudigen tot fig.16a.

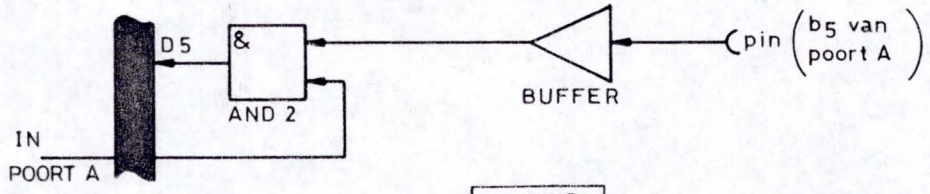

fig.16a

Door een besturingssignaal IN POORT A wordt de op de pin aangeboden informatie via de buffer en EN-poort AND2 op D5 van de interne databus geplaatst. Aangezien in poort A acht van deze schakelingen zitten, hebben we zo een normale 8-bits input-poort verkregen.

Opmerking:

PC1 = PC2 wil zeggen, dat b_2 en b_3 van het command register gelijk zijn. Dus beide 0, of beide 1. Als b_2 en b_3 niet gelijk zijn (PC1 $\neq$ PC2), dan wordt de poort voor strobed I/O geprogrammeerd.

Output-poort (PA = 1, PC1 = PC2). Omdat geldt PA = 1 geeft AND1 het commando OUT POORT A aan de klokingang van FF1. De reset-ingang is niet actief en de tri-state buffer is continu in de geleidende toestand. SWITCH wordt door PA, PC1 en PC2 in stand a geplaatst.
FF2 speelt dus geen rol. Voor dit geval kunnen we fig.15 dan vereenvoudigen tot fig.16b.

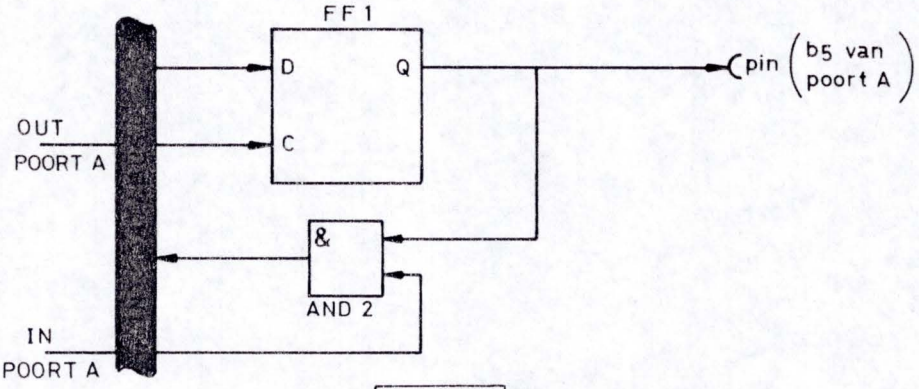

fig.16b

Door een besturingssignaal OUT POORT A wordt D5 van de interne databus in FF1 geklokt en op de Q-uitgang geplaatst. Evenzo voor de overige 7 bits van poort A. Dit is dus een normale output-poort.

Vraag 21: In dit geval kan de uitgevoerde data wel/niet door de CPU worden teruggelezen.

23-09 Antw.21: wel.

Als poort A nu d.m.v. een input-instructie wordt geselecteerd, wordt door het besturingssignaal IN POORT A de uitgevoerde data via de 8 EN-poorten AND2 op de interne databus geplaatst. De CPU kan dus de via een output-poort uitgevoerde data steeds teruglezen.

Strobed input-poort (PA = 0, PC1 ≠ PC2). Omdat geldt PA = 0, spelen FF1, AND1 en de tri-state buffer geen rol. Door PA, PC1 en PC2 wordt SWITCH in stand c geplaatst. Fig.15 is voor dit geval te vereenvoudigen tot fig.16c.

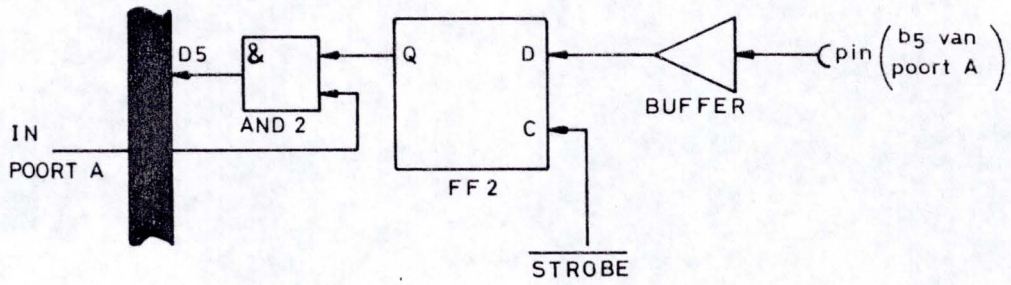

fig 16c

D.m.v. een actief $\overline{\text{STROBE}}$-signaal wordt de toestand van b_5 in FF2 geklokt. Door een commando IN POORT A wordt de inhoud van FF2 op de interne databus geplaatst. Dit gebeurt natuurlijk ook voor de overige 7 bits van poort A. Het bij strobe input behorende besturingssignaal BUFFER VOL wordt door poort C verzorgd.

Strobed output-poort (PA = 1, PC1 ≠ PC2). Door PA, PC1 en PC2 wordt SWITCH in stand a gezet. Fig.15 kan in dit geval weer worden vereenvoudigd tot fig.16b. D.w.z. dat poort A zich als een normale output-poort gedraagt. Wanneer we strobed output willen plegen, moeten de noodzakelijke besturingssignalen BUFFER VOL en INTR (zie fig.14) door de hardware van poort C worden opgewekt.

Alles wat in deze paragraaf over poort A is geschreven, geldt ook voor poort B.

SAMENVATTING 8

29. De poorten A en B kunnen elk voor 4 modes worden geprogrammeerd, nl. als
    normale input-poort,
    normale output-poort,
    strobed input-poort,
    strobed output-poort.

30. Met PA en PB (b_0 en b_1 van het command register) kan de datarichting, d.w.z. input of output, worden geprogrammeerd.

31. Met PC1 en PC2 (b_2 en b_3 van het command register) wordt de keuze tussen normale en strobed I/O gemaakt.

32. Als een poort voor strobed I/O is geprogrammeerd, dan doen 3 lijnen van poort C dienst als besturingslijnen voor de strobed I/O-poort.

33. Als een poort als output-poort is geprogrammeerd, dan kan de CPU steeds de uitgevoerde data m.b.v. input-instructies teruglezen.

d. Poort C

D.m.v. PC1 en PC2 (b_2 en b_3 van het command register, zie fig.11) kan poort C in vier verschillende modes worden geprogrammeerd. We zullen deze 4 modes nu achtereenvolgens kort bespreken.

Mode 1: (PC1 = 0, PC2 = 0). Poort C werkt als een normale 6-bits input-poort. Door een actief besturingssignaal IN POORT C (afkomstig van de I/O-control, zie fig.9), wordt de op de 6 input-lijnen aangeboden data op D0 t/m D5 van de interne databus geplaatst (fig.17a). D6 en D7 zijn dan ongedefiniëerd.

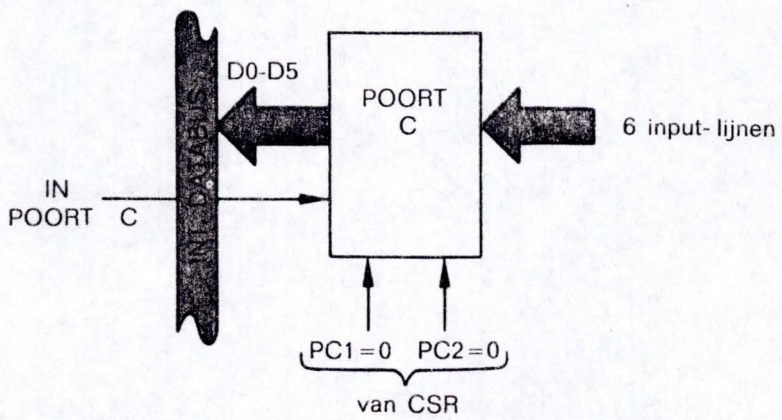

fig. 17a

Mode 2: (PC1 = 1, PC2 = 1). Poort C werkt als een normale 6-bits output-poort. Door een commando OUT POORT C wordt de data van D0 t/m D5 van de interne databus op de 6 output-lijnen geplaatst (fig.17b). De data van D6 en D7 van de databus worden dus niet uitgevoerd.

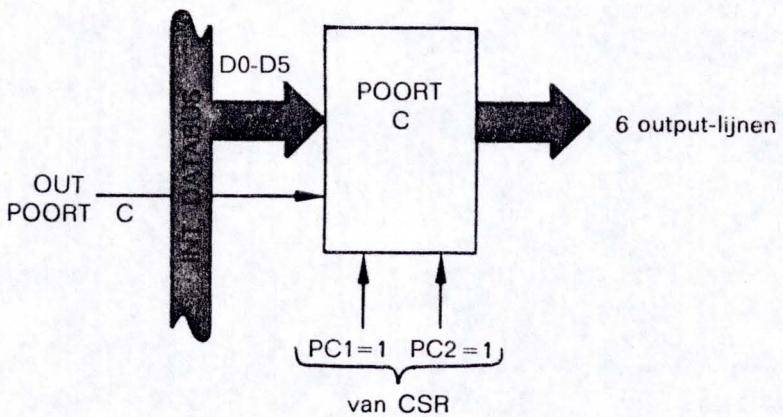

fig. 17b

Mode 3: (PC1 = 1, PC2 = 0). Poort C werkt nu gedeeltelijk als besturings-poort voor poort A en gedeeltelijk als output-poort (zie tabel 5 en fig.17c).

bits van poort C	MODE 3
b_0	A INTR (= interrupt request t.b.v. poort A)
b_1	A BF (= BUFFER VOL t.b.v. poort A)
b_2	A $\overline{\text{STB}}$ (= $\overline{\text{STROBE}}$ t.b.v. poort A)
b_3	output-lijn
b_4	output-lijn
b_5	output-lijn

Tabel 5

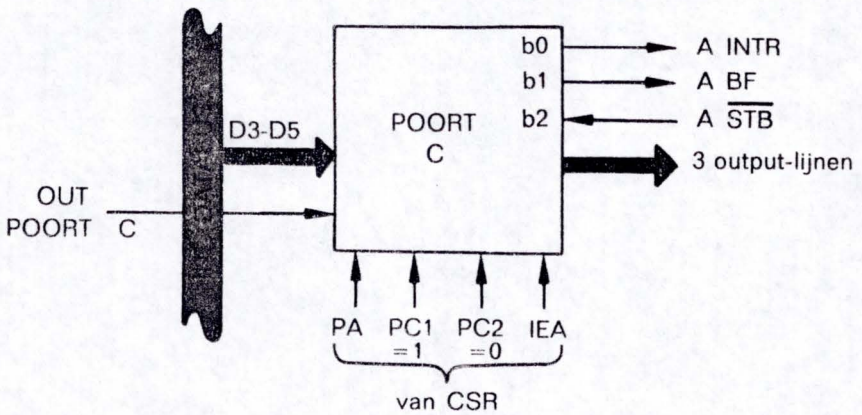

fig. 17c

b_3, b_4 en b_5 van poort C zijn normale output-lijnen. Door een commando OUT POORT C (afkomstig van de I/O-control), wordt alleen de data van D3, D4 en D5 op de drie output-lijnen geplaatst. De overige bits van de databus kunnen in deze mode niet worden uitgevoerd via poort C (natuurlijk wel via poort A of poort B).

Poort A is in deze mode voor strobed input (als PA in het command register 0 is) of voor strobed output (als PA = 1) geprogrammeerd. De drie besturingssignalen voor de strobed I/O via poort A, dus b_0, b_1 en b_2 van poort C moeten hardware-matig met de CPU en het randapparaat worden verbonden. In deze mode werkt poort B dus als normale input-poort (als PB in het command register 0 is) of als normale output-poort (als PB = 1).

Opmerking:

Door het vullen van de interrupt enable bit van IEA (b_4 van het command register) kunnen we aangeven of poort C wel (IEA = 1) of niet (IEA = 0) een interrupt request INTR A aan de CPU moet zenden. Dit is van belang als de programma-uitvoering in de microcomputer gedurende bepaalde tijd niet door interrupt I/O of strobed I/O mag worden onderbroken.

Mode 4: (PC1 = 0, PC2 = 1). Poort C werkt nu geheel als besturingspoort voor strobed I/O via de poorten A en B (zie tabel 6 en fig.17d).

bits van poort C	MODE 4
b_0	A INTR (= interrupt request t.b.v. poort A)
b_1	A BF (= BUFFER VOL t.b.v. poort A)
b_2	A $\overline{\text{STB}}$ (= $\overline{\text{STROBE}}$ t.b.v. poort A)
b_3	B INTR (= interrupt request t.b.v. poort B
b_4	B BF (= BUFFER VOL t.b.v. poort B)
b_5	B $\overline{\text{STB}}$ (= $\overline{\text{STROBE}}$ t.b.v. poort B).

Tabel 6

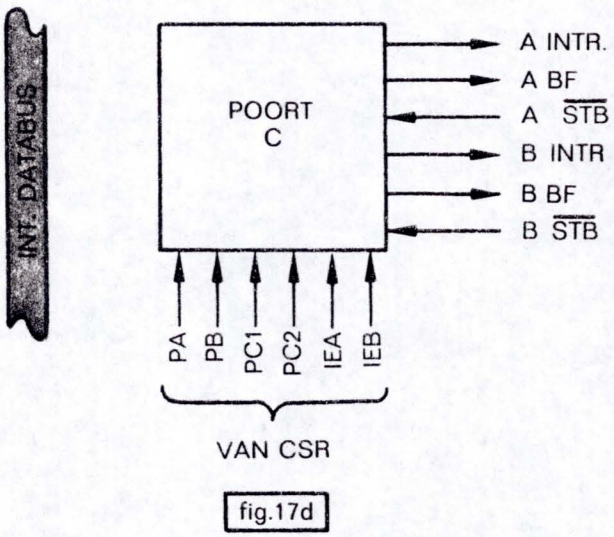

fig.17d

b_0, b_1 en b_2 zijn evenals in mode 3, de besturingssignalen voor strobed I/O via poort A. Nu werken b_3, b_4 en b_5 als de besturingssignalen voor strobed I/O via poort B. In mode 4 werken poort A en poort B dus beide met strobed I/O. Voor elke poort is met PA resp. PB in het command register te programmeren of het strobed input of strobed output betreft. D.m.v. IEA en IEB (interrupt enable bits, b_4 en b_5 van het command register) is het afgeven van interrupt requests A INTR resp. B INTR toe te staan of te verbieden.

Vraag 21: Het is wel/niet mogelijk om poort A als normale input-poort en tegelijk poort B als strobed input-poort te programmeren.

In mode 3 werkt alleen poort A als strobed I/O-poort. In mode 4 werken de poorten A en B beide met strobed I/O. Het is dus niet mogelijk om alleen poort B als strobed I/O-poort en tegelijkertijd poort A als normale I/O-poort te laten functioneren. Als er slechts één strobed I/O-poort vereist is, dan moet dit dus poort A zijn.

M.b.v. de lijnen A INTR en B INTR van poort C is het natuurlijk ook mogelijk om interrupt I/O te plegen.

Antw.21: niet.

De conclusies, die we uit deze en voorgaande paragrafen kunnen trekken, zijn:

1. D.m.v. de 8155 kunnen we 22 (programmeerbare) I/O-lijnen voor normale I/O realiseren.
2. De poorten A en B kunnen ook worden gebruikt voor strobed I/O en interrupt I/O. We moeten dan de 6 I/O-lijnen van poort C opofferen t.b.v. de noodzakelijke besturingssignalen.

e. Voorbeeld

We hebben de mogelijkheden van het I/O-gedeelte van de 8155 in afzonderlijke delen besproken. We zullen nu enkele mogelijkheden samenvoegen in een voorbeeld (fig.18). Hierin zijn alleen de databus en de belangrijkste besturingssignalen weergegeven.

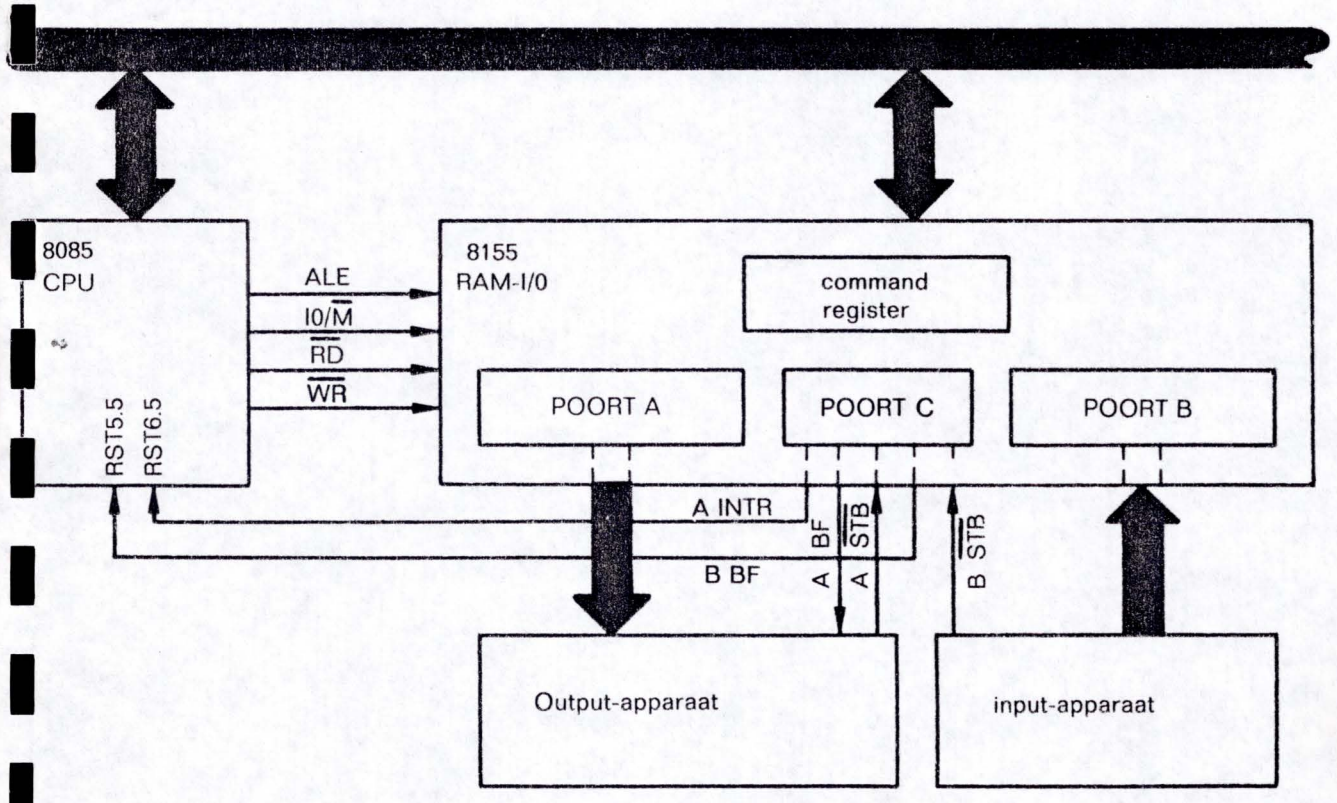

fig. 18

Op de 8155 zijn een input- en een output-apparaat aangesloten, die beide d.m.v. strobed I/O met de rest van het microcomputersysteem communiceren.

Vraag 22: De strobed input vindt plaats op basis van geprogrammeerde/interrupt I/O. De strobed output vindt plaats op basis van geprogrammeerde/interrupt I/O.

De lijnen A INTR en B BF zijn verbonden met interrupt request-ingangen van de CPU. Zowel de strobed output via poort A als de strobed input via poort B vindt plaats op basis van interrupt I/O. In beide gevallen zijn de voordelen van strobed I/O en interrupt I/O dus gecombineerd.

 Antw.22: interrupt; interrupt.

Vraag 23: IEA in het command register moet 0/1 zijn.

Omdat poort C interrupt requests op de lijn A INTR moet plaatsen, moet IEA (b_4 van het command register) 1 zijn. Voor IEB geldt deze voorwaarde niet. De lijn B INTR wordt nl. niet gebruikt, omdat het BUFFER VOL-signaal (B BF) in dit geval als interrupt request dienst doet.

In dit voorbeeld zien we duidelijk het verschijnsel programmeerbare chip naar voren komen. Immers, aan het begin van het uit te voeren programma moet zowel de interne interrupt controller in de CPU als het I/O-gedeelte in de 8155 worden geprogrammeerd. Hiertoe moeten we dus het interrupt mask register en het command register met bepaalde bitcombinaties vullen.

We gaan nu bepalen met welke bitcombinaties we het systeem op de gewenste wijze kunnen laten functioneren. Hierbij gaan we ervan uit, dat de niet-gebruikte bits 0 zijn. Dit zijn b_7, b_6 en b_5 van het interrupt mask register en b_7, b_6 en b_5 van het command register.

Vraag 24: Het interrupt mask register moet worden gevuld met ……$_2$ = ……$_{16}$.

Voor het interrupt mask register moet gelden:
$b_0 = 0$, om RST5.5 te laten honoreren.
$b_1 = 0$, om RST6.5 te laten honoreren.
$b_2 = 1$, $b_4 = 1$ } om RST7.5 niet te laten honoreren.
$b_3 = 1$, om b_0, b_1 en b_2 te kunnen wijzigen.

Het interrupt mask register moet dan m.b.v. een SIM-instructie worden gevuld met $00011100_2 = 1C_{16}$.

Vraag 25: Het command register moet worden gevuld met ……$_2$ = ……$_{16}$.

Voor het command register moet dan gelden:
b_0 = PA = 1 (poort A = output)
b_1 = PB = 0 (poort B = input)
b_2 = PC1 = 0, b_3 = PC2 = 1 } mode 4
b_4 = IEA = 1 (enable A INTR)

Het command register moet dan m.b.v. een OUT-instructie worden gevuld met $00011001_2 = 19_{16}$.

SAMENVATTING 9

34. Poort C kan voor 4 modes worden geprogrammeerd. Hiertoe dienen PC1 en PC2 (b_2 en b_3 van het command register).

35. In mode 1 werkt poort C als normale 6-bits input-poort. De poorten A en B zijn dan ook normale I/O-poorten.

36. In mode 2 werkt poort C als normale 6-bits output-poort. De poorten A en B zijn dan ook normale I/O-poorten.

37. In mode 3 werkt poort C deels als normale 3-bits output-poort (op b_3, b_4 en b_5) en deels als besturingspoort (op b_0, b_1 en b_2) t.b.v. de strobed I/O-poort A. Poort B werkt dan als normale I/O-poort.

38. In mode 4 werkt poort C geheel als besturingspoort voor de strobed I/O-poorten A en B.

12. TIMER-GEDEELTE (8155)

In veel applicaties moet de tijd nauwkeurig worden bijgehouden.
Enerzijds om iets op een bepaald tijdstip te laten gebeuren, anderzijds om te meten welke tijd er tussen twee gebeurtenissen is verlopen.
Dit soort tijdmetingen is mogelijk door gebruik te maken van software wachtlussen. In veel gevallen is dit om twee redenen onpraktisch.
Ten eerste omdat een werkelijk nauwkeurige tijdmeting niet mogelijk is, en ten tweede omdat de CPU in een dergelijke wachtlus zinloos werk uit voert. De tijd zou beter besteed kunnen worden aan de uitvoering van de instructies uit het feitelijke hoofdprogramma.

Vaak wordt daarom een tijdmeting uitgevoerd m.b.v. een hardware timer.
Zo'n timer bestaat uit een digitale teller, met een stabiele (kristal) klok. Als de telfrequentie hoog is en de teller uit voldoende flip flops bestaat, kan én een nauwkeurige én tegelijk lange tijd worden ingesteld.
Een dergelijke timer is in de 8155 opgenomen.

In fig.19 is het blokschema van het timer-gedeelte van de 8155 weergegeven.

De taak van de timer control is het juist laten functioneren van het gehele timer-gedeelte. De timer zelf is een 14-bits down counter, waarvan de inhoud door elke klokimpuls met 1 wordt verlaagd.
Als de inhoud van de timer 0 is geworden, wordt er een signaal aan de timer control afgegeven. Deze geeft dan een TIMER OUT-signaal af.
Welke vorm dit signaal heeft, hangt af van de mode, waarin de timer functioneert.
Het count length register bevat steeds de 14-bits waarde, waarmee de counter wordt gevuld bij het starten. b_{14} en b_{15} (M1 en M2) van het count length register dienen om de timer in een van de 4 mogelijke modes te programmeren.

We zullen nu achtereenvolgens de mogelijkheden en eigenschappen van het timer-gedeelte bespreken.

a. Adresselectie en besturingssignalen

De timer control zorgt er o.a. voor dat de juiste besturingssignalen voor de overige delen binnen het timer-gedeelte worden opgewekt.
Hiertoe worden A0, A1, A2 (van de interne adresbus), IN en OUT (afkomstig van de control logic van de 8155, zie fig.8) gebruikt. Dit is in tabel 7 weergegeven.

A2	A1	A0	IN of OUT	afgegeven
X	X	X	-	-
0	X	X	X	-
1	0	0	OUT	WR CLRLO
1	0	0	IN	RD TMLO
1	0	1	OUT	WR CLRHI
1	0	1	IN	RD TMHI
1	1	X	X	-

X = don't care.
- = geen actief signaal aanwezig.

Tabel 7

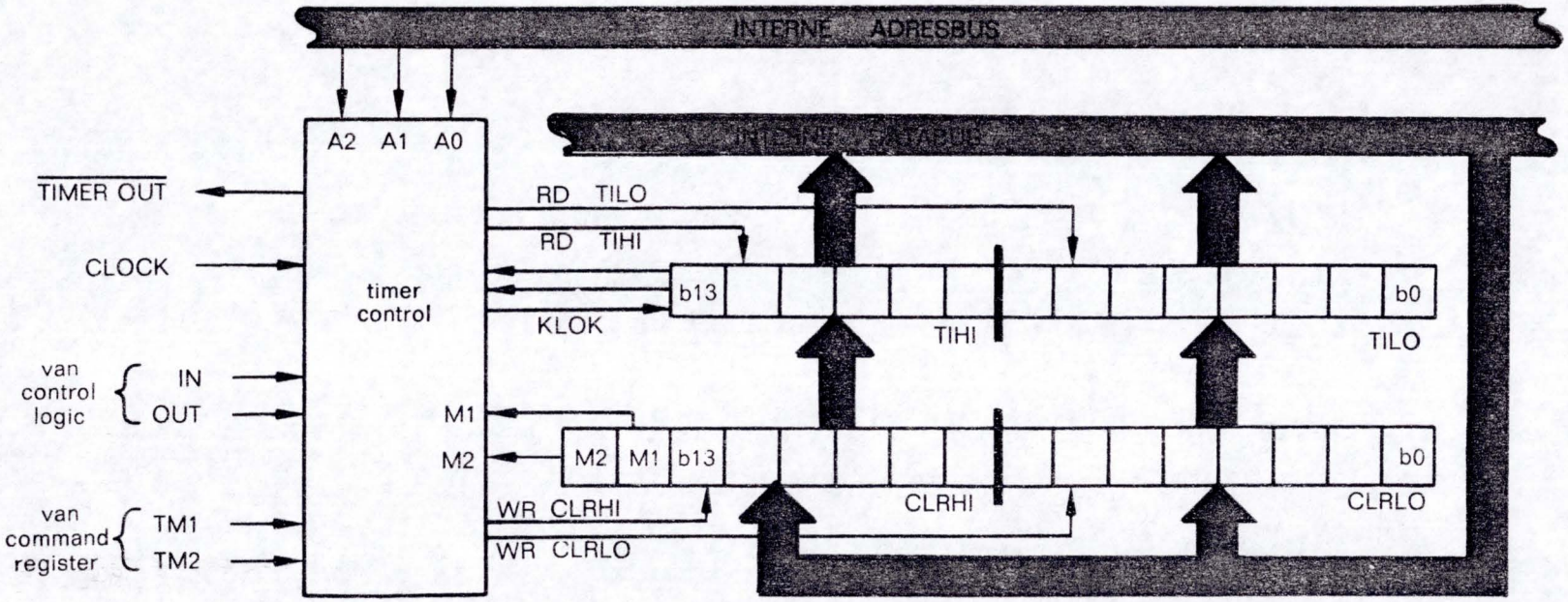
INTERNE ADRESBUS
INTERNE DATABUS
A2 A1 A0
TIMER OUT
CLOCK
timer control
RD TILO
RD TIHI
KLOK
b13
b0
TIHI
TILO
van control logic
IN
OUT
M1
M2
M2 M1 b13
b0
CLRHI
CLRLO
van command register
TM1
TM2
WR CLRHI
WR CLRLO

fig.19

In tabel 7 betekent
CLRLO: low order byte van count length register.
CLRHI: high order byte van count length register.
TILO : low order byte van timer.
TIHI : high order byte van timer.

Uit tabel 7 blijkt dat er alleen een locatie binnen het timer-gedeelte wordt geselecteerd als A2, A1 en A0 de combinatie 100_2 of 101_2 vormen. Er zijn in het timer-gedeelte dus maar 2 locaties die gevuld en uitgelezen kunnen worden.

Vraag 26: Als A2 = 0 wordt een locatie binnen het I/O-/timer-gedeelte geselecteerd.

Door I/O-adressen, waarin A2 = 0, wordt een van de 4 locaties binnen het I/O-gedeelte geselecteerd (zie tabel 4). In de 8155 zijn dus totaal 6 locaties d.m.v. I/O mapped I/O te adresseren.

b. Count length register

Vraag 27: Om het gehele count length register te vullen, zijn 1/2/4 output-instructies nodig.

Het count length register bestaat uit 16 bits, dus 2 bytes. b_0 t/m b_{13} bevatten de beginwaarde, die bij het starten van de timer naar de overeenkomstige 14 bits van de down counter worden overgebracht. b_{14} en b_{15} (M1 en M2) dienen om een van de 4 timer-modes te programmeren.
Met M1 wordt aangegeven of de timer, na het bereiken van de waarde 0, gestopt moet blijven (M1 =0) of automatisch opnieuw moet worden gestart (M1 = 1). M2 geeft aan welke vorm het TIMER OUT-signaal moet hebben. Dit kan een korte impuls (M2 = 1) of een symmetrische blokgolf (M2 = 0) zijn. Fig.20 geeft een overzicht van de vormen, die het TIMER OUT-signaal in elk van de 4 timer-modes heeft.

In elk van de 4 timing-diagrammen wordt de timer op t = 1 gestart en is op t = 3 de inhoud van de timer tot 0 gedaald. De tijd tussen t = 1 en t = 3 is dus de tijd, die we d.m.v. b_0 t/m b_{13} van het count length register hebben ingesteld.

In mode 1 is met M2 = 0 de symmetrische blokgolf geselecteerd. Als de timer-inhoud tot de helft van de beginwaarde is teruggeteld, dan wordt het TIMER OUT-signaal actief (t = 2). Dit signaal blijft actief, totdat de timer-inhoud 0 geworden is (t = 3). TIMER OUT wordt dan 1 en de timer stopt.

Mode 2 verloopt op dezelfde wijze, met dat verschil, dat op t = 3, t = 4, enz., de timer opnieuw vanuit het count length register wordt gevuld, en direct weer wordt gestart. Het TIMER OUT-signaal is dus een continu doorgaande blokgolf, totdat de timer d.m.v. een instructie wordt gestopt.

In mode 3 telt de timer gedurende de geprogrammeerde tijd. Als de timer-inhoud 0 geworden is, wordt er een korte impuls op de TIMER OUT-lijn afgegeven en de timer stopt.

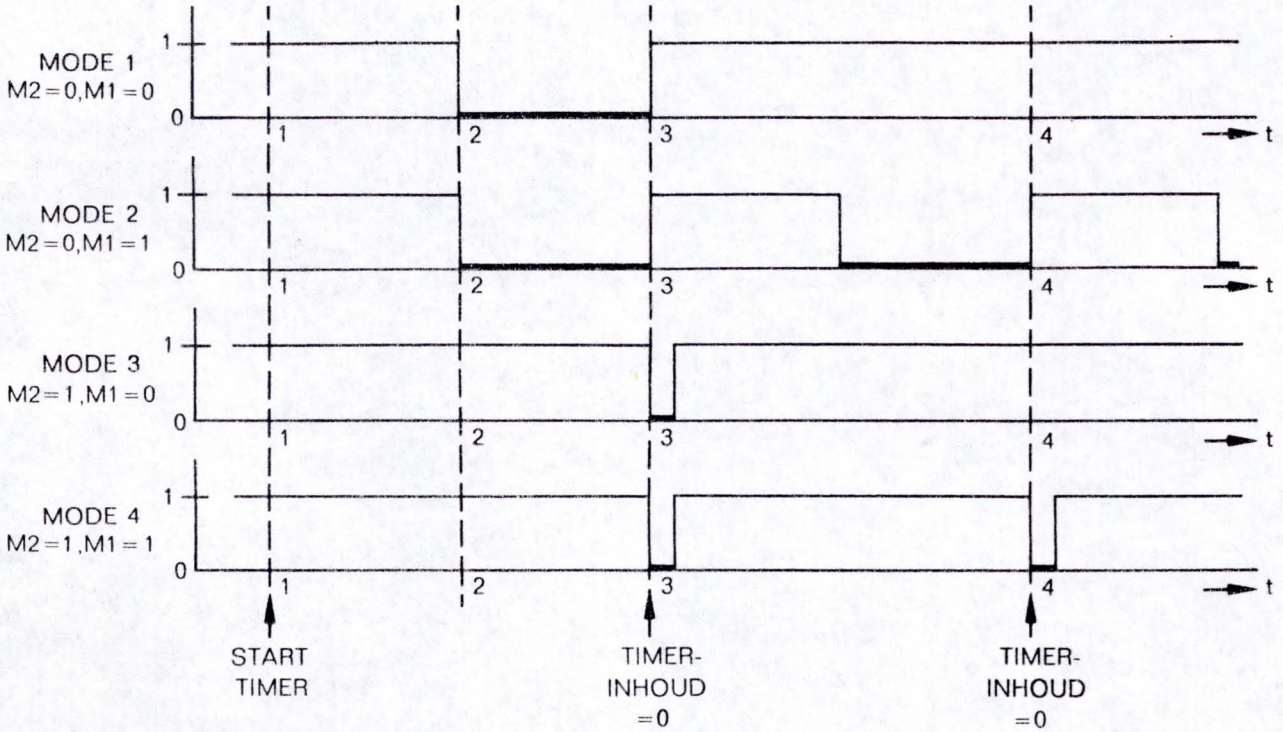

fig. 20

In mode 4 gebeurt hetzelfde, met het verschil, dat op t = 3, t = 4, enz., de timer automatisch opnieuw wordt gevuld en gestart.

Het count length register kan alleen worden gevuld (m.b.v. 2 output-instructies). Wanneer we een input-opdracht met het I/O-adres van CLRLO of CLRHI laten uitvoeren, dan wordt de overeenkomstige byte van de timer uitgelezen.

c. Timer

De timer zelf is een 14-bits down counter, die bij het starten wordt gevuld vanuit b_0 t/m b_{13} van het count length register. Door elke klokimpuls wordt de timer-inhoud met 1 verlaagd. Als de inhoud tot de helft van de beginwaarde of tot 0 is gedaald, worden er signalen aan de timer control afgegeven. Deze bepaalt hiermee de vorm van het TIMER OUT-signaal en of de timer wel of niet direct opnieuw moet worden gestart.

D.m.v. twee input -instructies kan de inhoud van de timer worden uitgelezen. Dit kan van belang zijn, als ergens in een programma moet worden bepaald hoe lang het nog duurt, voordat de timer-inhoud 0 wordt.

d. Starten en stoppen van de timer

Met b_{14} en b_{15} (M1 en M2) van het count length register, bepalen we in welke mode de timer moet opereren. Met b_6 en b_7 (TM1 en TM2) van het command register kunnen we de timer laten starten en stoppen. In tabel 8 zijn de mogelijkheden hiervoor weergegeven.

TM2	TM1	Gevolgen
0	0	Er verandert niets. Als de timer bezig is te tellen, blijft deze doorgaan.
0	1	De timer wordt direct gestopt.
1	0	De timer stopt als de timer-inhoud 0 geworden is. (Als de timer al gestopt was, dan blijft deze gestopt.)
1	1	De timer wordt vanuit het count length register gevuld en gestart. Als de timer al gestart was, dan wordt deze opdracht pas uitgevoerd als de timer-inhoud 0 geworden is.

Tabel 8

We zullen dit verduidelijken in een voorbeeld. Stel, dat de timer al is gestart, maar dat we de timer direct opnieuw willen starten.

Vraag 28: Hiervoor moeten we 1/2/3 maal TM2 en TM1 veranderen.

In tabel 8 komt geen combinatie van TM2 en TM1 voor om de timer direct opnieuw te starten, als deze reeds gestart was. We zullen dus de timer eerst moeten stoppen (met TM2 = 0 en TM1 = 1) en meteen weer starten (met TM2 = 1 en TM1 = 1).

Wanneer we de timer willen starten, moeten we zorgen dat eerst het count length register is gevuld.

Vraag 29: Het starten van de timer beïnvloedt de inhoud van het count length register wel/niet.

Bij het starten van de timer wordt de down counter vanuit het count length register gevuld. De inhoud van het count length register blijft daarbij ongewijzigd. Dit houdt in, dat als de timer een aantal malen gedurende dezelfde tijd moet tellen, we niet steeds het count length register behoeven te vullen. Na de eerste keer kunnen we volstaan met het geven van het commando start timer. Dit doen we dus d.m.v. b_7 en b_6 (TM2 en TM2) van het command register.

Omdat b_0 t/m b_5 van dit command register worden gebruikt voor het programmeren van het I/O-gedeelte, moeten we zorgen dat het starten en stoppen van de timer geen invloed heeft op deze bits. Hoe dit gerealiseerd kan worden, is in paragraaf 13 beschreven.

Antw.28: 2. Antw.29: niet.

SAMENVATTING 10

39. Het gebruik van een hardware timer heeft als voordelen
    a. er is een nauwkeurige tijdmeting mogelijk.
    b. de CPU kan gedurende de tijdmeting doorgaan met het uitvoeren van het programma.

40. De timer in de 8155 kan in 4 modes opereren, nl.
    a. enkele symmetrische blokgolf,
    b. continue blokgolf,
    c. enkele TIMER OUT-impuls,
    d. continue TIMER OUT-impulsen.
    De gewenste mode wordt met M1 en M2 van het count length register geprogrammeerd.

41. De timer kan m.b.v. TM1 en TM2 van het command register worden gestart en gestopt.

42. Bij het starten wordt de down counter vanuit het count length register gevuld.

43. De timer-inhoud is op elk moment m.b.v. 2 input-instructies uit te lezen.

13. DE BASIC 8155 IN DE SDK 85

In de voorgaande paragrafen zijn de eigenschappen en mogelijkheden van de RAM-I/O-timer chip 8155 behandeld. In deze paragraaf wordt beschreven hoe de 8155 op de plaats van de basic RAM-I/O chip is benut. Hierin is de 8155 volgens fig.21 met de CPU verbonden.

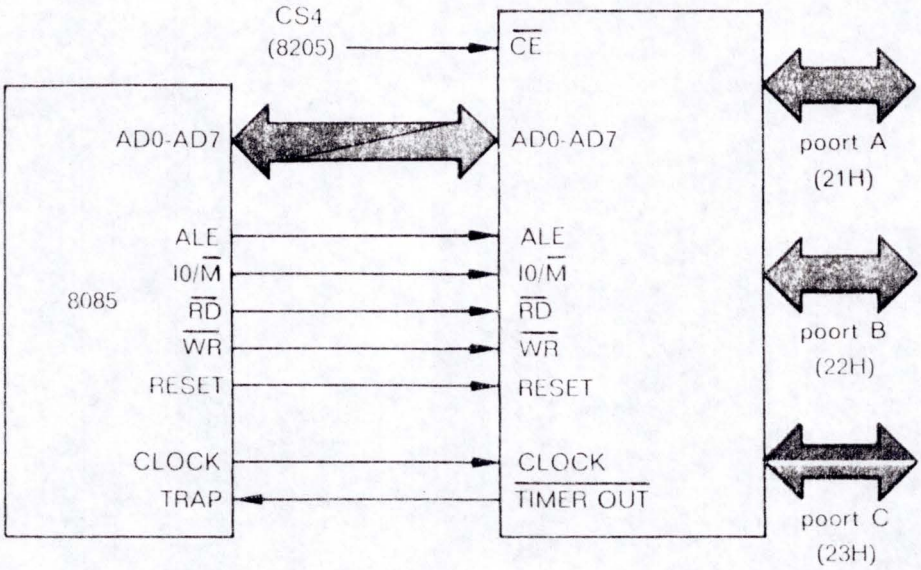

fig. 21

Vraag 30: Er wordt een RAM-locatie geselecteerd als IO/$\overline{M}$ = 0/1.
Het RAM-gedeelte wordt geselecteerd bij de adressen .....$_{16}$ t/m .....$_{16}$. Dit zijn ..... adressen.
De capaciteit van het RAM-gedeelte is ..... bytes.

Om een geheugenwoord in het RAM-gedeelte te selecteren, moet gelden IO/$\overline{M}$ = 0 en $\overline{CE}$ = 0. $\overline{CE}$ is verbonden met de CS4-uitgang van de adresdecoder 8205. Deze uitgang is 0, als er zich op de adresbus een adres bevindt, waarvan A15 t/m A11 de combinatie 00100_2 vormen. Dit zijn de adressen 2000_{16} t/m $27FF_{16}$, dus totaal 2048 verschillende adressen.

Het RAM-gedeelte van de 8155 heeft een capaciteit van 256 bytes. De interne adresbus binnen de 8155 is dan ook 8 bits breed. Intern werkt de 8155 dus met de RAM-adressen 00_{16} t/m FF_{16}.
Doordat A10, A9 en A8 van de adresbus geen rol spelen bij de adresselectie binnen het RAM-geheugen, wordt elk van de RAM-locaties geselecteerd met 8 verschillende adressen. Dit is in tabel 9 vermeld.

Antw.30: 0; 2000; 27FF; 2048; 256.

intern adres	wordt geselecteerd bij de adressen
00	2000, 2100, 2200, 2300, 2400, 2500, 2600 en 2700
01	2001, 2101, 2201, 2301, 2401, 2501, 2601 en 2701
02	2002, 2102, 2202, 2302, 2402, 2502, 2602 en 2702
03	2003, 2103, 2203, 2303, 2403, 2503, 2603 en 2703
.	.
.	.
.	.
FE	20FE, 21FE, 22FE, 23FE, 24FE, 25FE, 26FE en 27FE
FF	20FF, 21FF, 22FF, 23FF, 24FF, 25FF, 26FF en 27FF

Tabel 9

Dit houdt in, dat voor de 256 RAM-locaties in de basic 8155 2048 adressen uit de totale adrescapaciteit van de CPU worden bezet.

Vraag 31: Het I/O- en het timer-gedeelte worden geselecteerd als IO/$\overline{M}$ = 0/1 en $\overline{CE}$ = 0/1.
In de basic 8155 komen de I/O-adressen $\ldots\ldots_{16}$ t/m $\ldots\ldots_{16}$ voor.

Er wordt een locatie binnen het I/O-gedeelte of het timer-gedeelte geselecteerd als IO/$\overline{M}$ = 1. Natuurlijk moet dan de gehele chip zijn geactiveerd met $\overline{CE}$ = 0. Aangezien een I/O-adres zowel via de high als de low order byte van de adresbus wordt verstuurd, is $\overline{CE}$ = 0 bij de I/O-adressen 00100XXX. Dit zijn dus 00100000_2 t/m 00100111_2 = 20_{16} t/m 27_{16}. Hiervan worden er in de 8155 maar 6 gebruikt, nl. 4 voor het I/O-gedeelte en 2 voor het timer-gedeelte. In tabel 10 zijn deze I/O-adressen met de bijbehorende locaties weergegeven.

I/O-adres	geselecteerde locatie
20_{16}	command status register
21_{16}	I/O-poort A
22_{16}	I/O-poort B
23_{16}	I/O-poort C
24_{16}	low order byte van CLR (timer)
25_{16}	high order byte van CLR (timer)
CLR = count length register.	

Tabel 10

Vraag 32: Door de instructie OUT 20H wordt het command/status register geselecteerd.

Het command status register is verdeeld in het command register, te vullen met OUT 20H, en het status register, uit te lezen met IN 20H.

Vraag 33: De timer in de basic 8155 mag wel/niet door de gebruiker worden aangesproken.

Antw.31: 1; 0; 20; 25. Antw.32: command. Antw.33: niet.

De timer in de basic 8155 is toegewezen aan de SINGLE STEP-functie van de monitor van de SDK 85. Door een actief TIMER OUT-signaal wordt altijd naar de interrupt service routine van deze SINGLE STEP gesprongen. Dit houdt in dat we deze timer niet voor een ander doel mogen gebruiken.

Hierbij is echter nog een belangrijk punt, dat zowel de monitor als de gebruiker het command register aanspreekt. De monitor moet immers de timer starten en stoppen en de gebruiker wil vaak het I/O-gedeelte programmeren. Dit is mogelijk, mits de monitor alleen b_6 en b_7 van het command register beïnvloedt en de gebruiker zich tot b_0 t/m b_5 beperkt. Aangezien het command register alleen in zijn geheel is te vullen (dus alle 8 bits tegelijk) moeten er twee oplossingen worden gevonden.

a. Als de gebruiker het command register vult, moet hij(zij) ervoor zorgen, dat de werking van de timer niet wordt beïnvloed.

   Vraag 34: b_6 moet dan worden gevuld met 0/1 en b_7 met 0/1.

   De gebruiker moet b_6 en b_7 van het command register dan vullen met 0. Dit heeft nl. geen invloed op het gedrag van de timer (zie tabel 8).

b. Als de monitor het command register vult, moet in b_0 t/m b_5 die combinatie worden gezet, die de gebruiker er eerder heen had gestuurd. Hiertoe moet de monitor weten welke combinatie dit was.

   Vraag 35: Het command register is wel/niet uit te lezen.

   De inhoud van het command register is echter niet terug te lezen. De gebruiker moet in dit geval zorgen, dat de bitcombinatie, die naar het command register wordt gestuurd, tevens op een door de monitor adresseerbare locatie wordt geplaatst.

   In de SDK 85 is dit adres $20FF_{16}$ in het RAM-geheugen. Dit is de reden, dat in voorgaande lessen bij het initialiseren van de 8155 steeds de instructie STA 20FFH is opgenomen. Wanneer de monitor nu het command register aan moet spreken, "kijkt" deze eerst op adres $20FF_{16}$ welke combinatie de gebruiker in b_0 t/m b_5 wil hebben. Hiervoor wordt dan de combinatie voor b_6 en b_7 geplaatst. Zo voert de monitor onderstaande instructies uit om de timer te starten.

```
LDA  20FFH
ORI  C0H
OUT  20H
```

   Door de ORI-instructie worden b_7 en b_6 met 1 gevuld, terwijl b_0 t/m b_5 ongewijzigd blijven.

Antw.34: 0; 0. Antw.35: niet.

14. DE EXPANSION 8155 IN DE SDK 85

In fig.22 is weergegeven hoe de expansion RAM-I/O-timer chip 8155 met de CPU is verbonden.

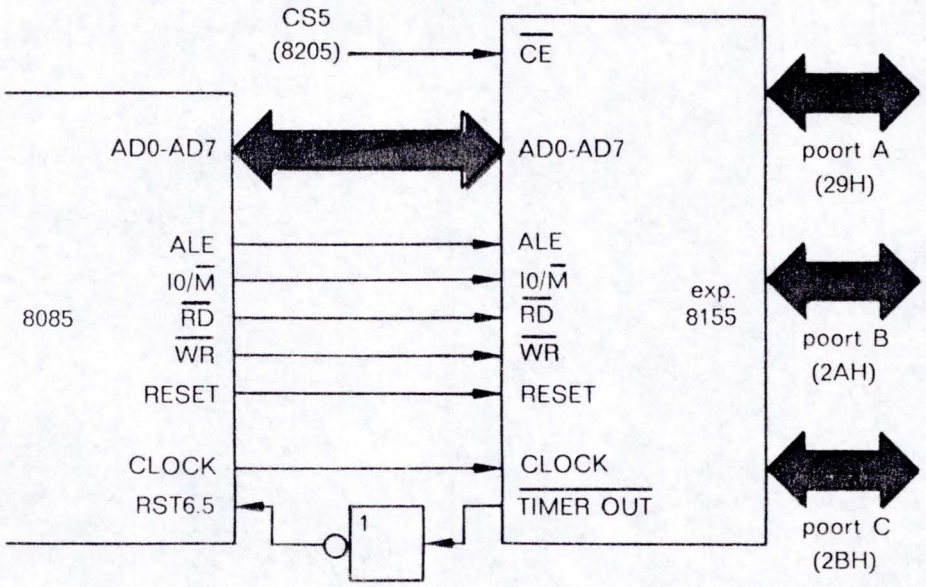

fig.22

Ook in de expansion 8155 wordt een geheugenwoord in het RAM-gedeelte door 8 verschillende adressen geselecteerd. Zo kunnen we b.v. het geheugenwoord 2800_{16} ook adresseren met 2900_{16}, $2A00_{16}$, $2B00_{16}$, $2C00_{16}$, $2D00_{16}$, $2E00_{16}$ en $2F00_{16}$.

De I/O-adressen in de expansion 8155 zijn in tabel 11 vermeld.

I/O-adres	geselecteerde locatie
28_{16}	command status register
29_{16}	I/O-poort A
$2A_{16}$	I/O-poort B
$2B_{16}$	I/O-poort C
$2C_{16}$	low order byte van CLR (timer)
$2D_{16}$	high order byte van CLR (timer)
CLR = count length register	

Tabel 11

Het gebruik van de timer in de expansion 8155 is in de les "Systeemeigenschappen hardware" al besproken. We beperken ons hier tot een verklaring, waarom de timer steeds in mode 2 (zie fig.20) is geprogrammeerd.
Als de timer nl. in mode 3 opereerde, dan zou dit boven mode 2 de volgende voordelen hebben gehad.

a. Na een TIMER OUT-impuls blijft de timer gestopt. We behoeven de timer dan niet m.b.v. 2 extra instructies te stoppen.
b. De maximale tijd tussen starten en de TIMER OUT-impuls zou twee maal zo lang zijn.

De TIMER OUT-uitgang is via een NIET-poort (inverter) met de RST6.5-ingang van de CPU verbonden.
De impulsen, die op de TIMER OUT-lijn in mode 3 (of mode 4) optreden, zijn echter te kort voor de CPU, om gedetecteerd te kunnen worden.
Daarom moet voor mode 1 of 2 worden gekozen.
Omdat de timer na elke actie wordt gestopt, is er in dit geval geen verschil tussen mode 1 en mode 2. U mag dus beide gebruiken.
Het bijbehorende timing diagram is in fig.23 weergegeven.

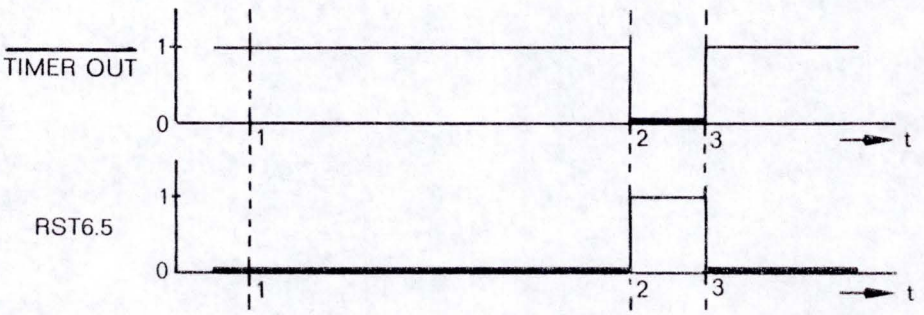

fig. 23

Op t = 1 wordt de timer gestart. Op t = 2 is de timer-inhoud tot de helft van de beginwaarde gedaald. Het TIMER OUT-signaal wordt laag, RST6.5 wordt hoog. De timer blijft nog doortellen, want de inhoud is nog niet 0 geworden. Door het honoreren van de interrupt request RST6.5 wordt naar de bijbehorende interrupt service routine gesprongen. Aan het begin hiervan wordt de timer gestopt (t = 3) en het TIMER OUT-signaal wordt hoog, RST6.5 wordt laag.

Opmerking:

De NIET-poort tussen TIMER OUT en RST6.5 is noodzakelijk, omdat de RST6.5-ingang van de CPU high level sensitive is (zie de les "Interrupt").

SAMENVATTING 11

44. Omdat bij de adresselectie van RAM-geheugenwoorden in de SDK 85 A10, A9 en A8 geen rol spelen, kan elke RAM-locatie door 8 verschillende adressen worden geselecteerd.

45. Omdat zowel de monitor als de gebruiker van de SDK 85 het command register in de basic 8155 aanspreekt, moeten beide rekening houden met de bits, die niet veranderd mogen worden.

46. Omdat de $\overline{\text{TIMER OUT}}$-impuls van de expansion 8155 te kort is voor de RST6.5-ingang van de CPU, moet de timer zodanig worden geprogrammeerd, dat op de $\overline{\text{TIMER OUT}}$-lijn een symmetrische blokgolf optreedt.

15. SLOTOPMERKINGEN

De in deze les beschreven programmeerbare chips 8355 en 8155 zijn slechts voorbeelden uit een grote reeks. Ze kunnen de systeemontwerper veel gemak bieden.
Men moet er evenwel steeds op letten, dat het ook noodzakelijk is zo'n programmeerbare chip te gebruiken.
B.v. wanneer men een eenvoudige serial interface nodig heeft, dan kan dat met een 8251 programmeerbare serial interface chip gebeuren.
Maar misschien is de serie in- en uitgang van de 8085 al voldoende.
Vooropgezet dat de tijd toereikend is.

In het algemeen bevatten alle programmeerbare chips veel hardware, die door het "setten" van "schakelaars" door software commando's geselecteerd en geactiveerd worden.
Daarnaast zijn zij zo intelligent, dat zij eenvoudige tot zeer moeilijke taken zelfstandig, dus zonder tussenkomst van de CPU, kunnen uitvoeren.
Dit leidt zelfs tot het toepassen van microprocessoren in gebieden, waar vroeger minicomputers gebruikt werden.

Voorbeelden van zulke intelligente programmeerbare chips zijn o.a.

- Floppy disk controller
- CRT controller
- Serial interface controller
- Cassette controller
- Dot matrix printer controller.

Er zijn zelfs chips, die eigenlijk een complete microcomputer bevatten.
Deze zijn zodanig opgebouwd, dat ze voor velerlei interface-toepassingen geschikt zijn.
Dit noemt men Universal Peripheral Interface microcomputers (b.v. de 8041). Hiermee kan een speciale intelligente interface worden gemaakt, door het schrijven van een programma, dat in een ROM in deze single chip microcomputer wordt opgeslagen.
Via een databus kan met een gewoon microprocessorsysteem data en commando's worden uitgewisseld net zoals bij b.v. de 8155.