

Slotexpander voor MSX 1 en MSX 2

Peter van Over-beek, PTC-PRINT nummer 36

Scanned, ocr'ed and converted to PDF by HansO

Zoals bekend beschikt elke MSX-computer over vier "slots". Minstens één, maar meestal twee daarvan zijn vrije slots, uitgerust met een stekker die van buiten de computer bereikbaar is. Hierin passen dan weer insteekmodules voor speciale functies, zoals een modem of een module ter uitbreiding van het aanwezige geheugen. Van de interne slots is erin MSX-2 computers altijd één geëxpandeerd, dat wil zeggen in vier "subslots" onderverdeeld. Maar ook de vrije slots kunnen van een zelfgebouwde slotexpander voorzien worden en dat is minder ingewikkeld dan menigeeen denkt.

Waarom een slotexpander?

Veel insteekmodules bevatten in ROM of EPROM vastgelegde programma's. Dergelijke in hardware verankerde software wordt firmware genoemd. Het voordeel van firmware is, dat de hierin opgelagen programma's onmiddellijk beschikbaar zijn, ze hoeven niet meer van disk of cassette geladen te worden. Bovendien nemen ze geen RAM-geheugen in beslag, zodat dit beschikbaar blijft voor andere toepassingen. Maar zo'n module bezet wel een van de vrije slots. Willen we de module uitwisselen, dan moet voor de veiligheid de computer eerst uitgeschakeld worden. Met een slotexpander wordt een slot in vier subslots onderverdeeld. Het vrije slot krijgt daardoor een vier maal zo grote capaciteit. Belangrijk is wel om een echte slotexpander te maken, die door de computer ook als zodanig wordt herkend. MSX-computers weten in een echte slotexpander namelijk zelf de weg naar alle subslots te vinden. Na het inschakelen van de computer onderzoekt deze alle slots en subslots op de mogelijke aanwezigheid van programma's. De gevonden firmware wordt in een tabel bijgehouden en wanneer later een bepaald programma met CALL <naam> aangeroepen wordt, weet de computer het feilloos in slot en subslot te vinden. Een slotexpander is daarom vooral geschikt voor uitbreidingen met ROM of EPROM geheugens die al software bevatten. In principe zou met een slotexpander ook het RAM-geheugen uitgebreid kunnen worden, maar daarvoor is een memory mapper toch beter geschikt. Die kan namelijk ook pagina's verwisselen en uitgebreid worden tot maximaal 256 banken van 16 kByte elk. Een memory mapper wordt echter bij het inschakelen van de computer niet doorzocht. Wanneer een memory mapper dan ook software in ROM zou bevatten, zal de computer de daarin aanwezige programma's niet zelf kunnen vinden.

Firmware uitbreiding

Elke MSX-computer bevat van huis uit al een flinke hoeveelheid in ROM vastgelegde firmware: BIOS en MSX-BASIC. MSX-2 computers bevatten ook nog enkele programma's die met een CALL op te roepen zijn: de ingebouwde (maar helaas zo trage) RAM-disk: CALL MEMINI, CALL MFILES, CALL MKILL en CALL MNAME. Deze

firmware zit meestal in subslot 3-0. In subslot 3-3 zit het disk formatteerprogramma, aan te roepen met CALL FORMAT en een programma dat MSXDOS van disk opstart: CALL SYSTEM. Wanneer CALL <naam> ingetikt wordt of als de computer dit in een BASIC-programma tegenkomt, worden alle daarvoor in aanmerking komende (sub)slots doorzocht. In de MSX- documentatie heet dit een "Statement handler". Als de <naam> gevonden is wordt het bijbehorende programma uitgevoerd. Na afloop daarvan keert de computer terug naar de oorspronkelijke toestand. Was dat een BASIC programma, dan wordt dat gewoon voortgezet. Subslot 3-0 bevat een zogenaamde "Device handler" voor het device "MEM:" dat is de eerder genoemde RAM-disk. Na het initialiseren wordt deze herkend, bijv. in: OPEN "MEM:" FOR INPUT etc. Door het plaatsen van een module met programma's in ROM of EPROM kan de al aanwezige firmware uitgebreid worden. Hebben we ook een slotexpander dan past er vier maal zoveel firmware in het vrije slot. Het 64 kByte adresbereik van de MSX-computer is, zoals bekend, verdeeld over vier pagina's van 16 kByte elk. Elke pagina mag in een ander slot of subslot worden gekozen. Firmware staat meestal in pagina 1 (adressen &H4000 tot &H7FFF) of in pagina 2 (adressen &H8000 tot &HBFFF). De computer herkent de aanwezigheid daarvan aan de code "AB" waarmee de pagina dan begint. Ook pagina 0 (adressen 0 tot &H3FFF) kan worden gebruikt, in dat geval moet aan het begin de code "CD" staan. Programma's in firmware kunnen niet beginnen op pagina 3 (adressen &HC000 tot &HFFFF). Het gebruik van pagina 3 is niet onmogelijk maar wel ingewikkeld. Verderop staat daar wat meer over.

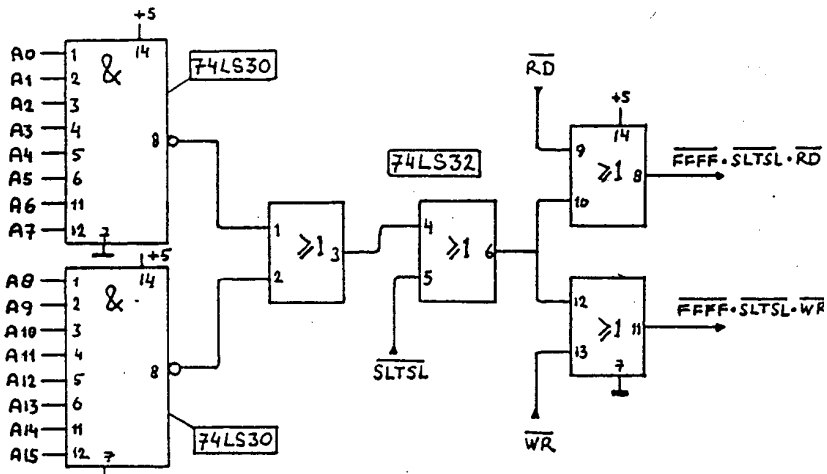


Fig.1 Decoder voor adres 14FFFF

PVO 12/89

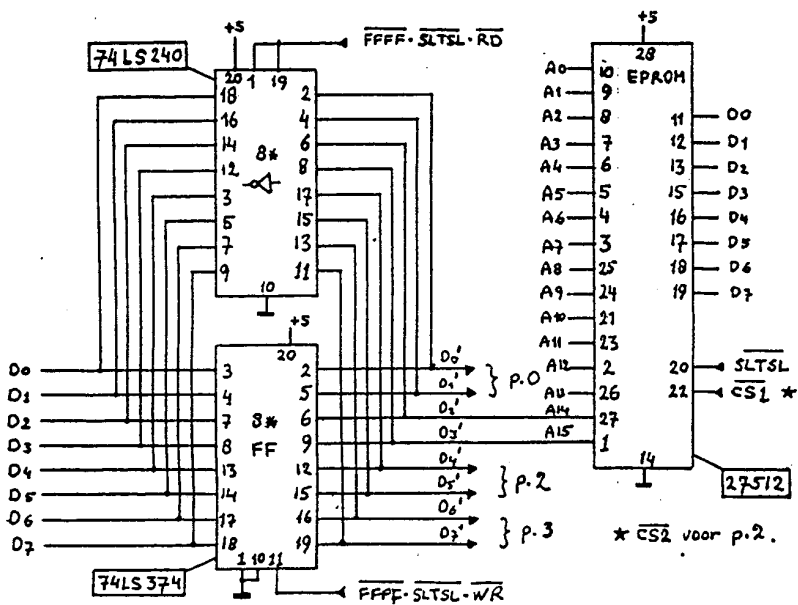


Fig.2 MSX SLOTEXPANDER VOOR FIRMWARE

PVO 12/89

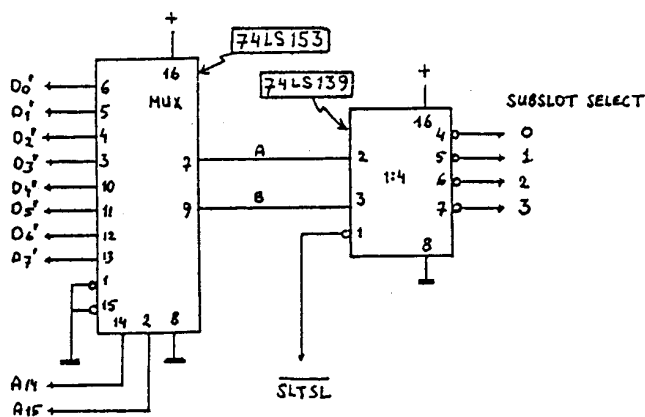


Fig.3. SUBSLOT DECODER

PVO 12/89

Hoe werkt een slotexpander?

Na het inschakelen van de computer wordt van elk slot het hoogste adres (&HFFFF oftewel -1) onderzocht. De computer probeert enkele keren een waarde naar dit adres te schrijven en leest vervolgens het resultaat. Alleen als daarbij het complement van de oorspronkelijke waarde wordt gevonden (alle bits zijn dan geïnverteerd) weet de computer dat het slot geëxpandeerd is. Het resultaat voorslot 0 t/m3 wordt bijgehouden in een tabelletje op de adressen &HFCC1 t/m &HFCC4. Hier staat 128 (&H80) als het slot in subslots onderverdeeld is of 0 als dit niet het geval is. Vervolgens worden de vier slots en alle gevonden subslots per pagina onderzocht op de aanwezigheid van firmware. De resultaten worden bijgehouden in een tabel die op adres &HFCC9 begint. Het bij dit artikel afgedrukte korte programma leest deze tabel uit en zet de inhoud in leesbare vorm op het scherm. Er zijn drie mogelijkheden, die ook gecombineerd per pagina voor kunnen komen:

- een Statement handler die in actie komt als de BASIC-vertaler CALL <naam> tegenkomt. Dit is de meest in aanmerking komende mogelijkheid voor eigen firmware uitbreidingen.
- een Device handler die zoekt naar de devicenaam zoals die voorkomt in OPEN"<devicenaam>:<filenaam>".
- een BASIC programma dat direct na het inschakelen van de computer uitgevoerd wordt. Beschrijving van de schakeling.

De schakeling bevat allereerst een decoder voor adres &HFFFF, (zie fig. 1) bestaande uit twee poorten van type 74LS30. In plaats van 74LS typen mag ook overal de 74HCT serie worden gebruikt. Met de vier OF-poorten van een 74LS32 wordt adres &HFFFF gecombineerd met het slot selectiesignaal SLTSL en dan enerzijds met het schrijfsignaal WR, anderzijds met het leessignaal RD. In fig.2 zien we een achtvoudige flip-flop 74LS374 die als een-byte geheugen dient voor adres &HFFFF. Via de achtvoudige inverter 74LS240 zijn de uitgangen van deze subslot-geheugenplaats weer met de databus verbonden, zodat de computer het complement van de inhoud kan lezen. De flipflops en inverters worden geactiveerd met de signalen uit de adresdecoder in fig.1, dus alleen voor adres &HFFFF in het betreffende slot. De slot expander zelf is daarmee in feite al compleet. De flipflops geven de te kiezen subslots als volgt aan: bits 0 en 1 voor pagina 1, bits 2 en 3 voor pagina 1, bits 4 en 5 voor pagina 2 en bits 6 en 7 voor pagina 3. Met deze signalen wordt het gewenste ROM of EPROM bereik gekozen. In het schema is een truc aangegeven om de kosten en het stroomgebruik van de slotexpander zoveel mogelijk te beperken. De vier subslots voor pagina 1 zijn daarin in één enkele EPROM, type 27512, opgeslagen, in plaats van in vier aparte EPROMs. De subslot-selectie wordt hierbij uitgevoerd via de adreslijnen A1 4 en A1 5 van deze EPROM. Hierin staat zodoende subslot 0 van adres 0 tot &H3FFF, subslot 1 van &H4000 tot &H7FFF, subslot 2 van &H8000 tot &HBFFF en subslot 3 van &HCOOO tot &HFFFF. De computer echter ziet vier maal pagina 1, dus van &H4000 tot &H7FFF maar dan wel in de vier verschillende subslots! Op soortgelijke wijze zijn ook alle vier de pagina's 1 in één EPROM vast te leggen. Natuurlijk kunnen de subslotselect signalen ook direct uit de inhoud van de acht flipflops gedecodeerd worden, zie fig.3. Dit is nodig als er met aparte ROMs of EPROMs per subslot wordt gewerkt. Dat moet ook gedaan worden als de slotexpander van vier stekers wordt voorzien om daarin de vier bestaande

insteekmodules te kunnen plaatsen. Alle signalen van het originele slot worden hierbij doorgelust, alleen op pin 4 worden de gedecodeerde subslot signalen aangesloten in plaats van het originele slotselect signaal SLTSL.

Het gebruik van de slotexpander

Zoals we al zagen is de slotexpander vooral geschikt voor uitbreiding van de firmware. Dat kunnen insteekmodules zijn met programma's die vaak gebruikt worden. In dat geval is een uitvoering met vier stekers voor de modules de beste oplossing. Dit heeft natuurlijk alleen zin als de programma's in die modules met een CALL aangeroepen kunnen worden en na afloop weer naar BASIC terugkeren. Degenen die vaak hulpprogramma's gebruiken die nu telkens van disk geladen moeten worden, kunnen die het beste in EPROM (laten) zetten. Voorbeelden zijn memory- of diskmonitor, assembler/disassembler, BASIC compiler, BASIC uitbreidingen enz. En wie zelf thuis is in het maken van machinetaal programma's zet natuurlijk zijn eigen programma's in zijn slotexpander. Mijn slotexpander bevat enkele tientallen kleinere programma's zoals een uitgebreide directory, overzichten van de gebruikte variabelen, van de (sub)slots, van de CTRL-codes, van de toetscodes en de printer codes, instellen van functietoetsen voorkeur, UN-NEW en nog veel meer. Van de kortere programma's passen er heel wat in een enkele pagina van een subslot. De pagina's 1 en 2 laten zich het gemakkelijkst in firmware vastleggen. Dat zijn voor de vier subslots al acht pagina's van 16 kByte, in totaal dus 128 kByte. Pagina 0 is ook te gebruiken, maar nu kunnen BIOS-routines alleen nog maar indirect aangeroepen worden. Het gebruik van pagina 3 is nog ingewikkelder. Daar geplaatste firmware wordt door de computer niet gevonden. Starten we toch een programma in pagina 3 dan zal dit vrijwel zeker een vastlopende computer veroorzaken. Toch hoeven we pagina 3 niet helemaal te vergeten. Wat namelijk wel vrij eenvoudig kan is het verplaatsen van programma's of gegevens vanuit pagina 3 naar een van de andere pagina's in RAM. Doe dit met een stukje machinetaal dat zelf niet in pagina 3 staat. Dit zet eerst de interrupts uit en schakelt vervolgens slot en subslot van de gewenste pagina 3 in. Gebruik de blok-verplaatsings-instructie LDDR of LDIR om de inhoud te verplaatsen naar RAM-geheugen. Schakel de originele pagina 3 weer in en zet dan de interrupts weer aan. Pagina 3 is zo te gebruiken om daarin teksten, plaatjes enz. op te slaan die gebruikt worden bij programma's die zelf in een andere pagina werken. Bijvoorbeeld overzichten van toetscodes, printer codes enz. Sommige programma's die in RAM wel goed lopen blijken dat niet meer te doen zodra ze in EPROM vastgelegd zijn. Dit komt dan meestal omdat binnen het programma zelf resultaten opgeslagen worden, wat in RAM wel maar in EPROM natuurlijk niet werkt. Toch heeft het nut zo'n programma in EPROM te zetten. Als het aangeroepen wordt laten we het eerst zichzelf naar RAM verplaatsen en daar uitvoeren. Dat is altijd nog veel sneller dan het van disk te laden. Ook in zo'n geval kan het programma best in pagina 3 vastgelegd zijn, omdat het daar toch niet uitgevoerd wordt. Alleen het allerlaatste byte van pagina 3, adres &HFFFF is echt niet te gebruiken. Dit adres is immers al in gebruik voor de slotexpander zelf.

Testprogramma slot expander

```
10 'MSX subslots met aanwezige firmware
20 'Peter van Overbeek, november 1989
30 CLS:M$="Slot:## Pagina:* "
40 PRINT"Overzicht MSX slots:"
50 FOR I=0 TO 3:PRINT"Slot"; I;
60 IF PEEK(&HFCC1+I)=&H80 THEN E(I)=3 ELSE E(I)=0
70 IF E(I) THEN PRINT"Vier subslots"ELSE PRINT"Geen subslots"
80 NEXT: PRINT"Aanwezige firmware:"
90 FOR I=0 TO 3:FOR J=0 TO E(I):FOR P=0 TO 3
100 Q=PEEK(&HFCC9+16*I+4*J+P)
110 IF Q AND &H20 THEN PRINT USING M$;I,J,P;: PRINT "Statementhandler"
120 IF Q AND &H40 THEN PRINT USING M$;I,J,P;: PRINT "Device handler"
130 IF Q AND &H80 THEN PRINT USING M$;I,J,P;: PRINT "BASIC programma"
140 NEXT:NEXT:NEXT
```

Correctie!

Theo Maassen PTC DECEMBER 1992 • NUMMER 62

Nee, niet weer 'n slotexpander maar enkele aanvullingen en correcties op het artikel van Peter van Over-beek in PTC-PRINT nummer 36.

De MSX-slotexpander zoals beschreven is uitermate geschikt voor gebruik van zogenaamde ROM-cartridges. Maar dan moet men de schakeling in figuur 3 van het genoemde artikel er ook bij bouwen. Deze schakeling decodeert de afzonderlijke subslots uit één enkel slot. Wanneer de slotexpander voorzien is van vier afzonderlijke connectors, is het mogelijk om in elke connector 'n module te plaatsen. De computer zoekt tijdens het initialisatie-proces (na een koude of warme start) in elk (sub)slot naar eventuele uitbreidingen. Zijn deze gevonden, dan wordt op een daarvoor bestemde plaats in het geheugen bijgehouden wat voor soort en vooral waar de computer deze module vinden kan. Na deze volzin is het u misschien nog niet helemaal duidelijk geworden, maar dit is in het kort wat er nu werkelijk gebeurt. Het artikel van Peter is hierin zeer duidelijk (bent u zelf niet in het bezit van PTC PRINT nr. 36? Vraag eens na op uw afdeling !) en om nogmaals hetzelfde te vertellen lijkt mij overbodig.

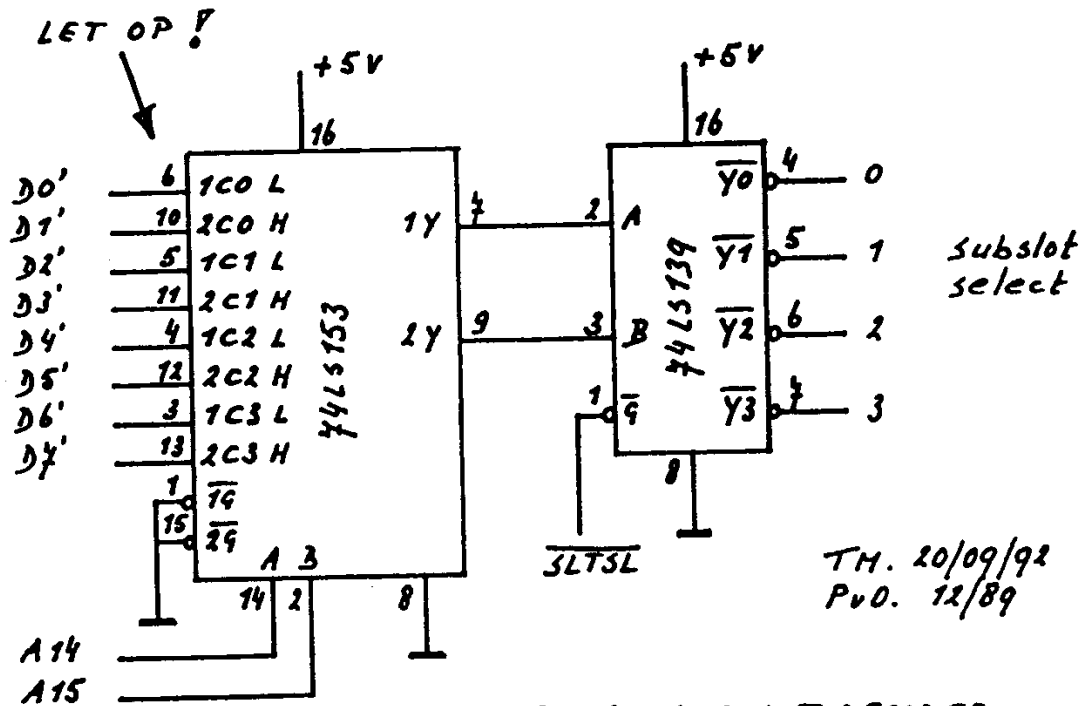


Fig. 3a SUBSLOT DECODER.

Fout

Helaas is in het artikel van Peter een nogal rampzalige fout gemaakt in fig. 3. Het werkt zo namelijk niet!. En waarom niet? Dat zal ik u proberen duidelijk te maken aan de hand

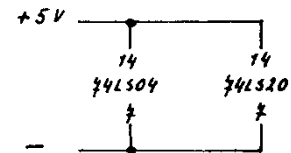
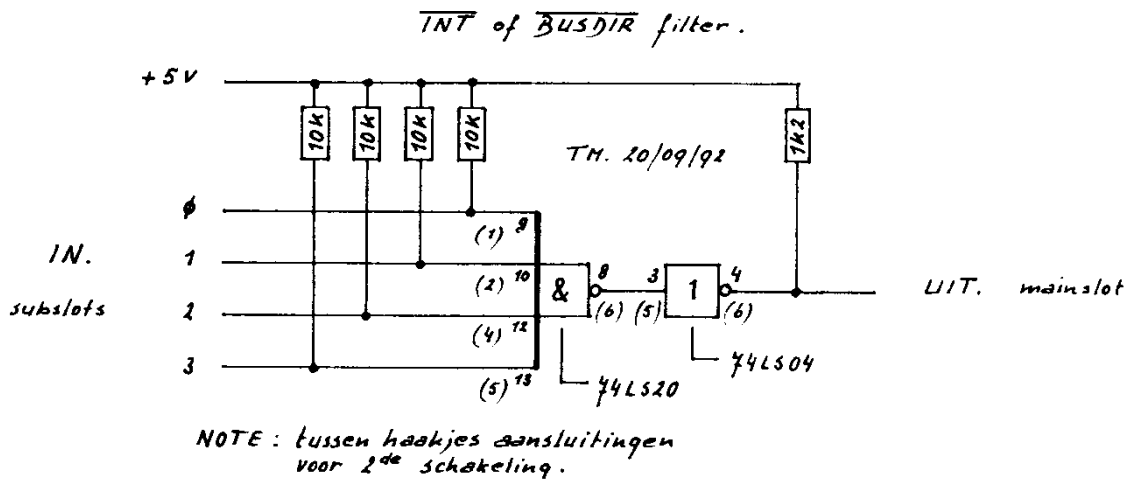
van de bij dit artikel geplaatste tekening. IC 74LS153 is een tweevoudige 4 naar 1 data multiplexer. Met behulp van de adreslijnen A14 en A15 wordt respectievelijk multiplexer 1 of 2 geselecteerd. Multiplexer 1 neemt de zogenaamde Low-bit voor z'n rekening en multiplexer 2 het High-bit. Deze Low- en High bits selecteren samen het subslot wanneer daarom gevraagd wordt. De bit-paren worden gevormd door DO'-D1', D2'-D3', D4'-D5' en D6'-D7' zoals ook te zien is in figuur 2 van het artikel in PTC-PRINT nummer 36. Na wat speurwerk kwam ik tot de ontdekking dat de aansluitingen van de Datalijnen (DO' t/m D7) niet zoals hiervoor beschreven waren aangesloten. Trouwens deze schakeling staat ook in het officiële MSX Technical Data Book (1.5.5 - Sample Circuit Diagram of Expanded Slot Select Signal). Na een telefoontje met Peter bleek dat hij deze schakeling daaruit had overgenomen. Had hij nou ook maar een paar bladzijden terug gebladerd, dan was het allemaal goed gekomen (1.4.9 - Slots / Advantages of slot structure). Maar niet getreurd, op deze tekening staat het wel goed en het nu werkt het goed.

Aanvullingen

Na deze correctie volgen er nu enkele aanvullingen, die speciaal bedoeld zijn voor hardware-uitbreidingen. Met hardware uitbreidingen bedoel ik dingen zoals MSX muziek module, modem, barcode-reader, RS232 interface en noem maar op. Deze modules beschikken vaak over wat extra aansluitingen zoals BUSDIR en INT (BUSDIR = Data richting in of uit, INT = external Interrupt op de processor, 280). Overigens, alle andere aansluitingen kunnen eenvoudig één op één worden verbonden met elkaar (m.u.v. SLTSL, BUSDIR en INT).

Voeding

Verder is het ook raadzaam om de +5, +12 en -12 Volt aansluitingen van een externe voedingsbron te betrekken. De MSX kan maximaal 300 mA op de +5 Volt en 50 mA op de +12 en -12 Volt leveren. Bij gebruik van meer dan één van de eerder genoemde modules wordt het allemaal een beetje te veel (De muziekmodule alleen al vraagt +5 Volt/ 200 mA, +12 Volt/30 mA en -12 Volt/16 mA). Daar het SLTSL signaal al besproken is gaan we nu het BUSDIR signaal bekijken.



BUSDIR

Met BUSDIR is ook iets bijzonders aan de hand. De computer wil graag met één module tegelijk communiceren. Dus wil niet beïnvloed worden door andere modules die eveneens op deze slotexpander zijn aangesloten. Er moet dus een filter gemaakt worden, dat er voor zorgt dat er maar één module tegelijk aanspraak kan maken op het data-richtings-register. En ook hiervan staat een voorbeeld van in het MSX Technical Data Book (1.5.5). Het BUSDIR signaal heeft negatieve logica wat wil zeggen dat het actief is als het laag is (TTL niveau).

NAND

Met behulp van een NAND (bijv. 74LS20) wordt het filter gemaakt. Het nadeel van dit filter is dat het signaal zich zogezegd omkeert. Als er een 0 (nul) ingaat komt er een 1 (een) uit en dat mag natuurlijk niet. We moeten dit signaal daarom opnieuw omkeren met behulp van een Inverter (bijv. 74LS04) zodat er tenslotte weer een 0 (nul) uitkomt. De tekening zal alles duidelijk maken. Voor het INT signaal kunnen we kort zijn, hiervoor geldt namelijk hetzelfde als voor het BUSDIR signaal. In de praktijk kunnen we volstaan met maar twee extra IC's voor beide signalen. In het IC 74LS20 (NAND) zitten twee van deze NAND poorten, en in het IC 74LS04 (Inverter) zes. Om de zaak wat op te fleuren kan men eventueel tussen de afzonderlijke subslot-select signalen schakelaars aanbrengen, waarmee het mogelijk wordt om een subslot met de hand uit te schakelen, zonder dat daarvoor de module uit de connector verwijderd hoeft te worden.

En door toevoeging van enkele lampjes (LED's) kan men zien dat de subslots ook daadwerkelijk geselecteerd worden (wel snel kijken a.u.b. !). Let op !! Alleen schakelen als de computer UIT staat. Voor de Schakelaars en LED's schakelaars kan men het beste zogenaamde 'dipswitches' gebruiken, die in een blokje van vier verkrijgbaar zijn. Voor de LED's kan men beter naar low-current exemplaren vragen in verband met het lage stroomverbruik (2 mA tegen 20 mA voor normale LED's). Eveneens i.v.m. het lage stroomverbruik kan men voor de IC's ook typen uit de HCT-serie nemen (74HCT20 en 74HCT04). Maak de lengte van de draden (bandkabel) tussen computer en slotexpander niet te lang (maximaal 50 centimeter), anders moet men met behulp van weerstanden de signalen gaan 'optrekken' naar hun respectievelijke niveaus (laag of hoog).

P.S. Normaal bouw ik de schema's van Peter van Overbeek zo na. Ze werken namelijk altijd, omdat ik weet dat Peter deze schakelingen altijd zelf uitprobeert. Maar laat hij nu net de uitbreiding volgens figuur 3 in zijn artikel niet gebouwd hebben! Zand (silicium) erover.