De XYLCD

deel 2 (slot)

door Wim Kruyf PAoWV

In deze aflevering komen o.a. het testen, de software ontwikkeling en te verwachten moeilijkheden bij de bouw van de 128 maal 256 punts grafische LCD display aan bod.

Testen van de busdeelschakeling met een ROM erin

Het ROM geeft onzintekst, vaak is de originele testzin te herkennen met verschoven stukken van de letters erin. Diagnose is daaruit niet te stellen. Mogelijk is de 8 MHz pulstrein te snel voor het ROM, dat al 20 jaar oud is. Ruit hora.
Daar is achter te komen door de 8 MHz klok te verlagen tot 4 MHz of lager (2 en 1 MHz). Daartoe wordt het draadje dat de 8 MHz aanvoert op pen 14 van de 74LS157 op positie d4 losgemaakt en aan die pen 14 wordt een hulpdraadje gesoldeerd naar pen 14 van de 74LS161 op positie a2, die levert 4 MHz. Dan werkt het wel, dus het vermoeden was correct.

Omdat we de schakeling niet met ROM maar met SRAM willen bedrijven herstellen we de zaak weer zoals hij was voor deze proef. Je kunt ook zonder ROM de schakeling testen, als volgt:
Poot d0 t/m d7 op de RAM-voet beurtelings aarden moet 32 verticale lijnen op de display geven die steeds bij de volgende d een pixel verder verschoven zijn in horizontale richting.
Vervolgens als dat klopt beurtelings de adresdraden A0 t/m A11 op de RAM voet verbinden met pen 15 d3.

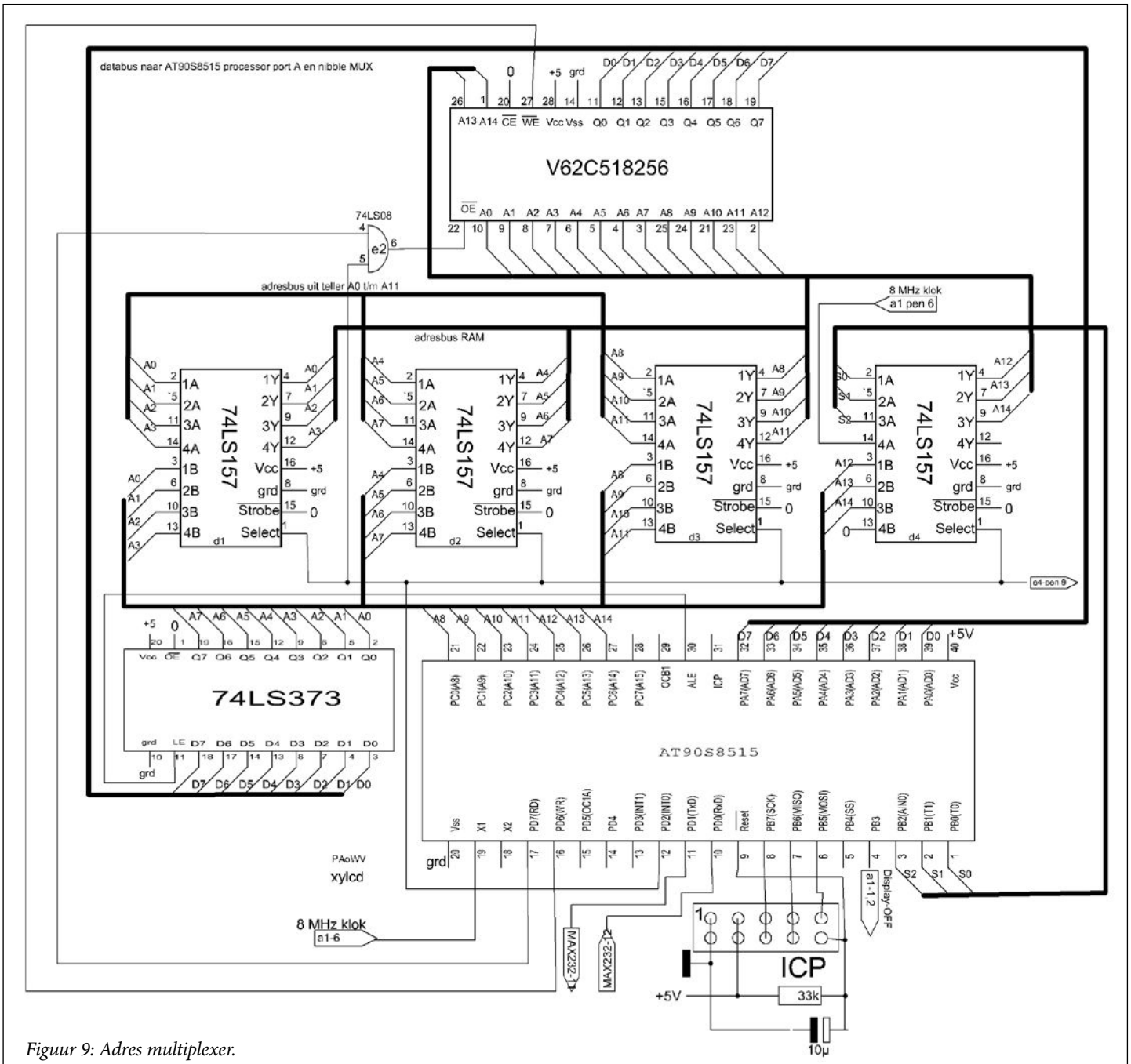
A0 pen 10: 16 verticale zwarte lijnen equidistant
A1 pen 9: 16 verticale lijnen in 8 paar van 2
A2 pen 8: 4 maal 4 vert. lijnen
A3 pen 7: 2 sets van 8 lijnen, in een set 7 gespatieerd
A4 pen 6: een set van 16 lijnen links in beeld, onderling 7 gespatieerd
A5 pen 5: 32 verticale stippellijnen 64 stippels per lijn
A6 pen 4: 32 vert. stippellijnen 32 stippels per lijn
A7 pen 3: idem maar nu 16 stippels per lijn
A8 pen 25: idem maar nu 8 stippels per lijn
A9 pen 24: met 4 stippels per lijn
A10 pen 21: met 2 stippels per lijn 32
A11 pen 23: met 1 stippel per lijn (alleen bovenste helft van 32 lijnen dus)

Als dit allemaal werkt kan de XYLCD adresbus eruit omdat daar de multiplexers mee in serie gezet worden. Tevens kunnen we het 32 k RAM monteren dat dan 8 in plaats van 2 pagina's toelaat, dat vereist dat ook de adresdraden A12, A13 en A14 gewijzigd worden.
Het schema in figuur 9 (zie volgende bladzijde) moet dus worden gerealiseerd met de soldeerbout.

Het RAM eist nu ook een lees- en schrijfpuls, we moeten voorkomen dat het RAM output geeft als erin geschreven wordt of als voor lezen het lage deel van het adres wordt aangeboden. Het kan met een overgebleven nor gate als inverter die WE inverteert op de OE pen van het RAM zetten. en tevens de CE wordt verbonden met de ALE. Dat is echter niet netjes, vooral niet omdat de ALE bij sommige typen chips niet gegarandeerd altijd laag is. Dus heb ik een extra IC 74LS08 gemonteerd dat maar voor een kwart gebruikt wordt, en dat de juiste logische functies aflevert zodat buscontentie bij aanbieden adres en schrijven van het RAM voorkomen wordt.

De verdere bouw

De schakeling in figuur 4 is, na sloop van de adresdraden A0 t/m A4 en A11 van de teller naar het ROM uitgebreid met de multiplexer uit figuur 9. De adresbus en databus van de processor zijn er nog niet en dan kan een en ander geprobeerd worden.
In principe is het verstandig met een pieper, dat is een schakeling met een heel lage open spanning, 100 mV of zo, die een pieptoon geeft als je de meetpennen kortsluit, om daarmee te kijken of er geen IC pennen contact maken met hun wederzijdse buren. Ook controleren of de pennen allemaal bezet zijn tenzij het schema anders aangeeft. Je kunt weer testen met het ROM in de schakeling geprikt. Dan 5 volt erop en kijken of de stroom de 150 mA (bij LSTTL, bij HTC TTL is dat maar 50 mA) niet significant overschrijdt.

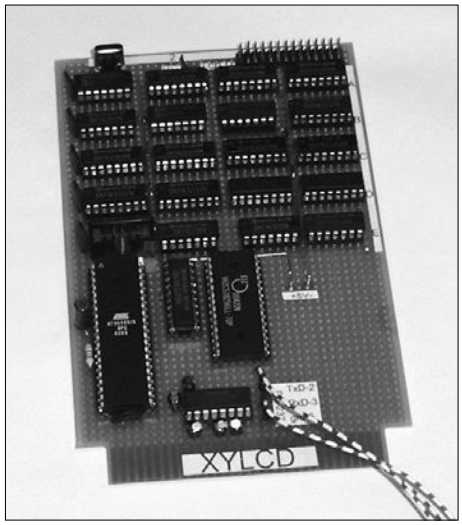


Figuur 9: Adres multiplexer.

Met een teller kun je de frequentie meten die aan de displaypennen wordt afgeleverd, en met een scope kun je bekijken of de CP pulsen burstgewijs worden afgeleverd in bundeltjes van 128 stuks. Als dat allemaal in orde is de display er pas aanhangen.
Vervolgens kun je als alles werkt de processor erin monteren en de adreslatch 74LS373. De bussen aan processorzijde aansluiten.
Testen kun je dan met de software doen, eerst primitief en als de display eenmaal bestuurbaar is vanuit de processor kun je die gebruiken voor de verdere softwareontwikkeling en debuggen.

De ontwikkeling van de software

Nu we een werkende processor hebben kunnen we het apparaat voor allerlei doeleinden gebruiken en gemakkelijk aanpassen. Je kunt er smith charts mee tekenen,



assen en meetwaarden erin zetten en tekstdisplay.

Natuurlijk volstaat hier de zin, dat je bij mij een geprogrammeerd IC kunt aanschaffen tegen kostprijs, of gratis geprogrammeerd als je zelf een IC aanlevert. Klaar. Ik wil echter je hier laten zien hoe je met amper niks zo'n programma opbouwt, want daar ontbreekt het de meesten niet aan die hun eerste schreden op dit pad willen wagen.

De (oude versie) AVR assembler die ik gebruik geeft als eerste regel van een object file onzin. Daar programmeer ik omheen door de eerste regel te laten veroorzaken door:

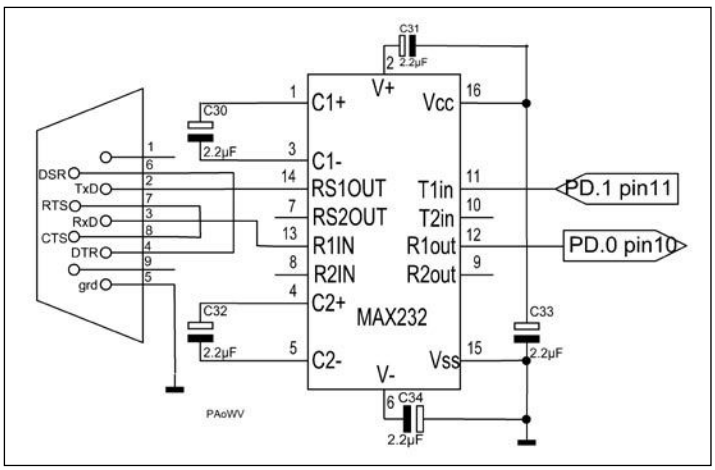
```
org 0
dw $FFFF
org 0
```

Gevolg is dat de eerste regel nutteloos is en gemist kan worden. De nieuwe versie

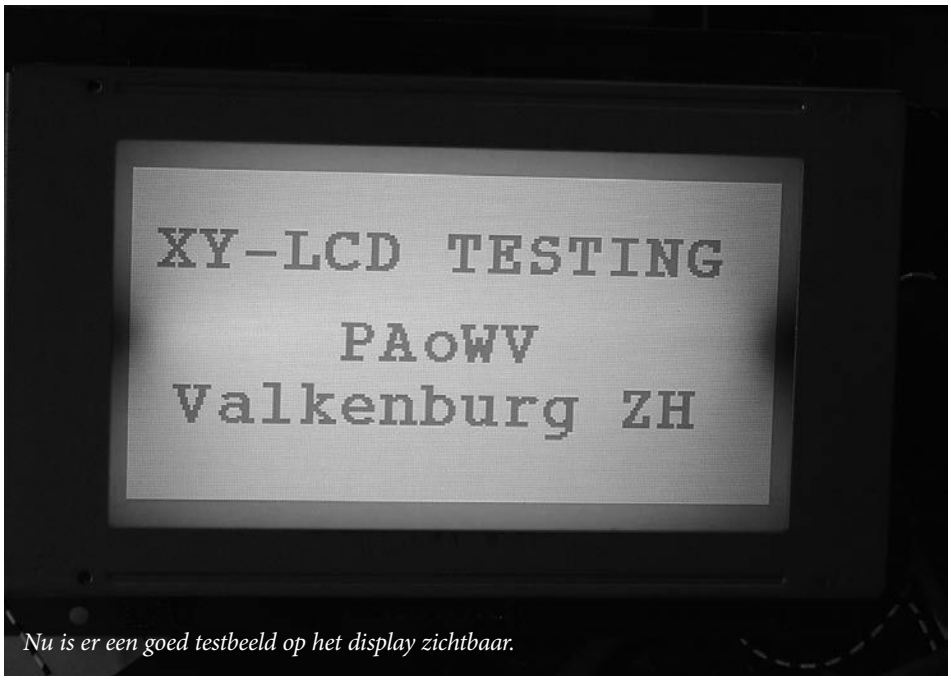
heeft dat niet en die geeft op die omzeiling van mij weer een foutbericht, hi.
Voorts moet je eraan denken dat je het ICP (in circuit programming) alleen kunt doen als de chip eerst gewist is, dat in tegenstelling tot de AT89S8252.

De eerste stap is te kijken of de selectpuls van de multiplexers op de externeinterrupt0 pin 12 staat. Makkelijkste met een scope, je kunt ook luisteren met een koptelefoon. Als het goed is staat die puls er met de correcte duty cycle van 15 hoog en 1 laag en is de frequentie 3904 Hz. De duty cycle kun je op een scope zien en als je die niet hebt met de gemiddelde waarde van de uitgangsspanning, die vormt een indicatie voor de duty cycle, omdat je dan de gelijkstroomcomponent meet.
3904 Interrupts zullen er dus per seconde gegenereerd worden op de achterflanken (opflanken) om het begin te markeren dat de processor mag schrijven in extern memory. Dat is het eerste stuk dat we gaan ontwikkelen.

De tweede stap is om de opflank van die puls een interrupt te doen veroorzaken, die een toegangsvlag voor toegang van de processor tot het extern geheugen op groen zet. Tevens start die interruptroutine een timer.
Dan is die interrupt klaar.
Loopt de timer af, dan wordt in de daardoor veroorzaakte timerinterrupt de toegangsvlag voor extern memory op rood gezet en tevens wordt de timer gestopt en gepreset op de gewenste beginstand. Daarmee is de timerinterrupt klaar.
De interruptroutine moet geschreven worden en daarvoor neem ik eerst een proef door portb op pen 1 beurtelings op 1 en op 0 te zetten, doordat de interrupt, als die werkt, die pen steeds complementeert.
Meten op pen 1 moet dan een blokgolf tonen van de halve interruptfrequentie dus 1953 Hz.
Klopt. Kun je met een koptelefoon ook beluisteren en de duty cycle meten met een analoge AVO op pen 1, die moet dus 50% zijn.



Figuur 10: De seriële interface.



Nu is er een goed testbeeld op het display zichtbaar.

Volgende stap

De int0 externe interrupt en de timerinterrupt programmeren met de taken hiervoor omschreven. Routines van slechts enkele regels.
Vergeet niet alle resources (variabelen) die worden gebruikt in de interrupt en het PSW (processor status word) op de stack te pushen, anders krijg je onvoorspelbare fouten, want je weet niet wanneer de interrupt optreedt gedurende de uitvoering van je programma.

Globaal zal de timer interrumperen na 15/16 van 1/3906 seconde, dus na 240 microseconde of minder. Van de datasheet gaat blz. 27 en 28 over de timers, dat moet dus bestudeerd met de bovenliggende vraag in gedachten.
Op grond daarvan wordt besloten de prescaler van timer 0 eerst maar op waarde 2 te zetten, die deelt dan door 8, dat is een count per microseconde, en de timer een interrupt te laten geven op overflow.
We gaan eerst proberen die timer interrupt aan de gang te krijgen.
De overflow interrupt enabelen we door TOIE1 in register TIMSK op 1 te zetten, en de prescaler op de klokfrequentie van 1 MHz door bit 1 van register TCCR0 op 1 te zetten. Chip wissen, assembleren en testen op pen 1.

De prescaler is voor de proef gezet op *8, de van 0 verschillende vlagwaarde is dus volgens de tabel van de datasheet de voorstand (= preset hi) van de 256 teller op 20. De tellerduur is dan (256-20)*8*klokfreq=236 us. We willen iets minder dan (15/16)*lijntijd=240 µs.
Meten met de dubbelstraalsscope op pen 1 die steeds geïnverteerd wordt en pen 12 die de selectpulsinput krijgt levert als resultaat: pen 1 start met op en neergaan na de achterflank van de selectpuls en houdt daarmee op 0,1 µs voor de aanvang van de voorflank van de multiplexer selectpuls. Prima dus.

Het blijkt dan dus dat met een prescaler/8 en een presetwaarde van 20 de teller bezig is tot dicht voor de volgende selectdownflank, dus dicht voor het moment dat de display de bus nodig heeft.
Narekenen: (256-20)*8 klokcycles = 236 µs en 15/16 maal 256=240.

Om te kijken of het geheugen goed werkt wordt pen 3 van e4 losgesoldeerd, de select blijft dan staan op processor selected.
We kunnen dan een geheugentestprogramma laden, dat bestaat uit het volschrijven van het RAM met zijn eigen adressen en vervolgens teruglezen. Wordt er een foute teruglezing gevonden die niet gelijk is aan het adres, dan wordt er een puls gegenereerd op pen 4 van de processor met een routine. Op die pen wordt een teller gehangen in de totaalstand en we gaan koffie drinken.
Je kunt nog eens je hand plat op de contacten van de schakeling leggen en de spanning een volt omhoog (beslist niet meer) en omlaagdraaien. Als er dan geen fouten opgetreden zijn is het geheugen goed. Voor de zekerheid sluit je even twee datapennen op het RAM met elkaar kort, dan moet het dus fouten regenen. Bij het omlaagdraaien van de spanning blijft de foutenteller op 0 staan, zelfs als de spanning 0 is geworden, hi.

Bij alle metingen dus steeds ervoor zorgen dat je begrijpt wat je aan het doen bent.

Moeilijkheden

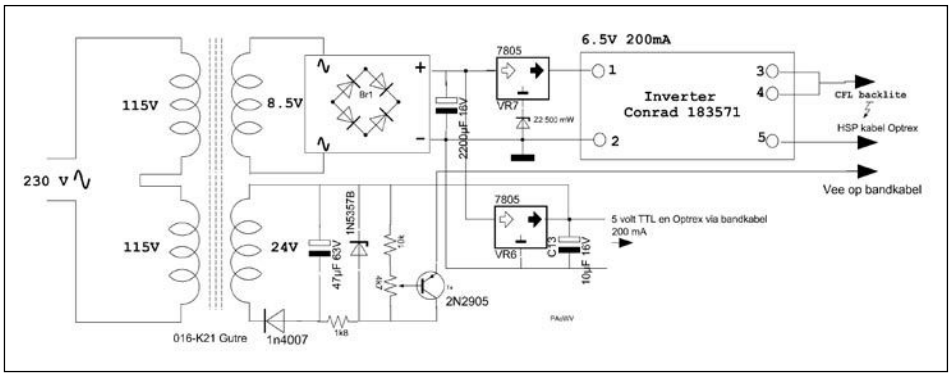
En dan denk je dat je er bent, maar nee hoor. De geheugentest is OK, je kunt weer het EPROM in de RAM-voet zetten en je krijgt je oude plaat (op 4 MHz) maar dat is het; verder klopt er geen jota van.

Drie dagen heb ik intensief zitten zoeken, meten, nadenken en proefjes zitten verzinnen om er achter te komen dat de databus die op het RAM zit altijd verbonden blijft met de port A op de controller en die daar gewoon adreswaarden op zet. Als de controller bezig is met lezen haalt hij dat keurig weg en leest de data, maar als hij niet bezig is en een andere schakeling wil lezen voor de LCDisplay dan stoort die adreswaarde.
Opgelost door de externe busRAM alleen in de software aan te sluiten tijdens de processortoegangstijden in de twee interruptroutines. Toen werkte het nog niet goed, na de derde dag bleek dat het kwam omdat als je de externe RAM niet gebruikt de oorspronkelijke portwaarden er instaan, en dat was in dit geval een 0 uit de schrijfpuls. Drie dagen tegenslag, toch doorgegaan.

Enerzijds denk je: het is hobby en dit is niet leuk meer, maar anderzijds: opgeven is verknoeide tijd en geen bevredigend resultaat. Als je het een tijdje laat liggen om wat anders te doen waar je wel zin in hebt, ben je de draad kwijt en belandt het in de junk box als de zoveelste unvolendete. DAT heb ik intussen wel geleerd. Het werkt nu feilloos en betrouwbaar, zonder Zuster Buitenhuis effecten en verschijnselen. En je leert er weer van, hoewel ik daar langzamerhand te oud voor ben, want morgen ben ik dat geleerde weer vergeten.

Toen de zaak eenmaal werkte is de UART van de controller geprogrammeerd en een RS232 DCE interface geplaatst (figuur 10).
Enkele routines geschreven voor controle van waarden in het geheugen met een PC als hyperterminal op de RS232 poort, en een geheugen hex dump, die op zijn beurt weer in een bmp file kan worden omgezet voor bestudering met paint onder windows. Zo krijg je de debug mogelijkheden beter van de vloer.
Tot zover de beschrijving van de ontwikkeling van het programma, het geeft de gang van zaken goed weer, en de rest ontwikkelen is van hetzelfde laken een pak.

Op de Dag van de Amateur 2008 heeft er een exemplaar van de XYLCD gestaan die geprogrammeerd is als FeldHell ontvanger. De basisband hellsignalen worden op externe interrupt 1 aangeboden.
Een timer werkt als 245 baud klokfrequentie zodat elk inkomend bit twee keer



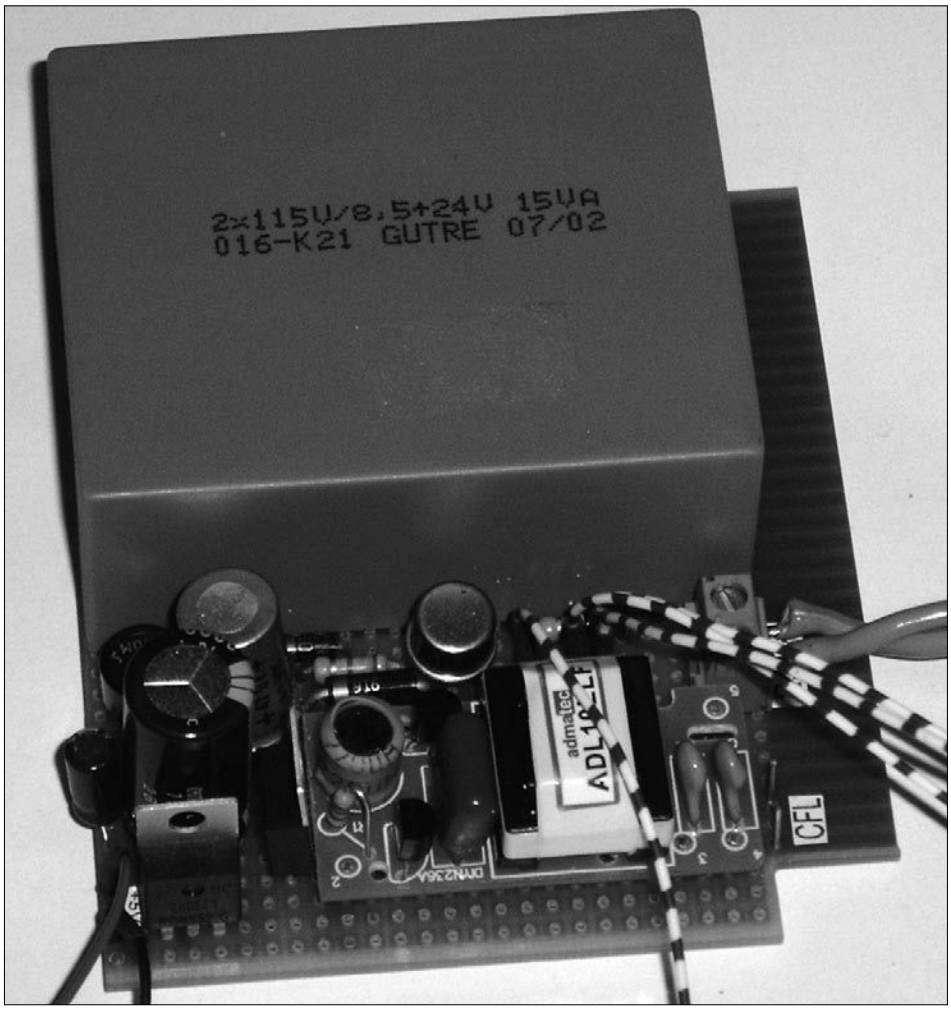
Figuur 11: De voeding.

bemonsterd wordt. Elke flank in het signaal zet de teller halverwege zodat in het midden van de bits wordt bemonsterd. Allemaal simpel te programmeren. Het geheugen wordt gevuld onderaan op het scherm 32 bits hoog waarvan 28 bits worden beschreven met de ontvangen punten in duplo. Als een regel vol is scrolt het beeld een band van 32 omhoog.

Slotopmerkingen

Als de duurdere Conrad losse TDK inverter wordt gebruikt (fig. 11), dan beide uitgangen pen 4 en 5 parallel zetten als ene kant, dan de CFL voeding en de andere uitgangspen 3 als de andere kant.
Voeding trekt bij 6,5 V dan 200 mA. Een 7805 met een 2 V zener in de aardpoot levert die waarde. Als trafo is een dumpgeval van Baco IJmuiden gebruikt 15 VA, 8,5 en 24 V gescheiden wikkelingen, die respectievelijk 0,8 en 8 Ω gelijkstroomweerstand vertonen. Die nemen dus een gelijk wikkelvolume in beslag en kunnen derhalve evenveel vermogen leveren. Beide dus 7,5 watt. Een zener van 20 V maakt een stabiele spanning voor de helderheidsregeling van Vee. De 24 V wikkeling wordt enkelfasig gelijkgericht met een 1N4007 en levert onbelast 42 V op de elco. 2k2 Ω volstaat dus als serieweerstand van de zener, omdat de display maar enkele mA trekt.
Een bedradingslijst van de print is beschikbaar op http://pa0wv.fol.nl/xylcd/bedrlyst.txt. Voor vragen en hulp pa0wv@vrza.nl.

PAoWV



De voeding, compleet met inverter.