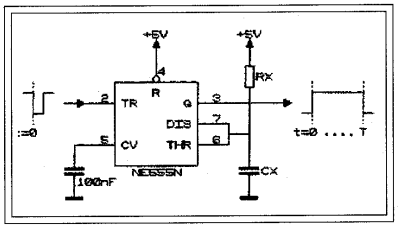
Condensatoren meten met behulp van UNIFACE

Inleiding

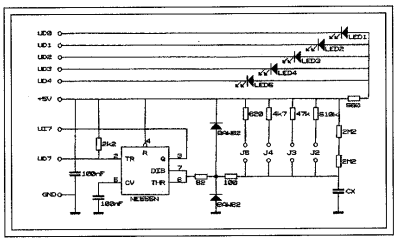
Enige tijd geleden heb ik een UNIFACE PC-interface aangeschaft en in mijn PC geplaatst. Hierdoor kreeg ik de beschikking over een aantal data-lijnen, waarmee de computer via twee I/O adressen met de buitenwereld te verbinden is, de UNIFACE bus. Om wat ervaring op te doen met UNIFACE en ook met I/O-gebruik op de PC, heb ik een schakeling gemaakt om condensatoren te meten. Een dergelijke schakeling had ik in het verleden al eens gemaakt voor gebruik op een geheel ander type computer.
De schakeling waarmee ik de hard- en software heb getest, is gebouwd op een experimenteerprintje. De hardware is opgebouwd rond een timer IC NE555N. De computer start en meet de timer over de UNIFACE bus en rekent de condensatorwaarde uit. Deze mag liggen tussen enkele tientallen pF en circa 2000 µF. Het gebruikte programma is geschreven in Turbo Pascal 5.0, vereist geen kennis van machinetaal en draait bij mij op een Philips PC-XT P3120.



De Hardware

Meetschakeling

Het principe van de one shot schakeling is te zien in figuur 1. Door de NE555N aan te sluiten zoals getekend, verkrijgen we een monostabiele multivibrator. Door een hoog-laag-hoog signaal op de trigger input aan te bieden, start de one shot. De output, die zich in de rusttoestand op laag niveau bevindt, gaat naar hoog niveau. Deze situatie blijft bestaan gedurende een tijd, welke voor een NE555N gegeven is door 1.1 * RX * CX seconden. Na deze tijd gaat de output weer naar laag niveau. Als we de tijds-duur van de puls aan de output bekijken, krijgen we een maat voor het RX * CX produkt. Als RX bekend is kunnen we CX laten uitrekenen.



Meetbereiken

De pulstijd is gelijk aan 1.1 * RX * CX. Indien we nu voor RX een tien maal lagere waarde nemen, meten we met een tien maal grotere condensator dezelfde pulstijd. Hiervoor kunnen we dan dezelfde tijdmeetroutine gebruiken. Voor elk gewenst meetbereik hebben we een andere weerstand RX nodig. Voor de hierna beschreven experimentele schakeling heb ik de bereikkeuze gedaan met behulp van een jumper.
Bij elk bereik horen enkele constanten om de condensator uit te kunnen rekenen.

Waarden van RX.

De minimum en de maximum waarde voor RX zijn af te leiden uit de specificatie van de NE555N. De laagste waarde wordt bepaald door de stroom, die naar "DIS" mag lopen. (Zie figuur 1.) Met een voeding van +5 Volt is 470 Ω voor RX te beschou-wen als minimum. De hoogste waarde voor RX wordt begrensd door lekstromen, welke door "DIS" en "THR" lopen. Het manual beveelt aan om de waarde 3,4 M Ωniet te overschrijden. Met wat tussenliggende waarden zijn mooi aansluitende bereiken te maken. Mijn eigen keuze om te kunnen testen met een factor 10 tussen opeenvolgende meetbereiken, leidde uiteindelijk tot een reeks van vijf weerstandswaarden. In het tijdbepalend stuk gedragen deze zich als de reeks 470E, 4k7, 47k, 470k, 4M7. Hiermee is het met een NE555N haalbare bereik ruimschoots benut.

Het complete schema

Het principeschema van de meetschakeling, zoals getekend in figuur 1, is verder uitgewerkt. Om de reproduceerbaarheid van het meten te kunnen bekijken en het program-ma te kunnen ontwikke-len en testen is de meetschake-ling van figuur 2 gebruikt.
De bereikkeuze vindt plaats door een jumper te plaatsen op de positie, die met een LED wordt aangewezen (zie naschrift. Red). Enkele delen van de schakeling worden hierna nog toegelicht.

Aansturing van de LEDs

In figuur 2 is te zien, hoe de LEDs zijn aangesloten op de UNIFACE data-uit lijnen UD0 t/m UD4. Om een bereik aan te geven, moet een logische "0" naar de desbetreffende datalijn gestuurd worden. UD7 is de triggerlijn voor het timer-IC en moet normaal altijd op "1" staan. UD6 en UD5 doen niet mee en mogen dus alles zijn. De meetbereiken 1, 2, 3, 4, en 5 kunnen dan aangegeven worden met achtereenvolgens de bytewaarden 254, 253, 251, 247 en 239. Zetten we één van deze bytes op de UNIFACE data-uit bus, dan brandt de juiste led en de conditie voor de triggerlijn UD7 is goed. Ik heb

de hardware zo gemonteerd, dat de oplichtende led in lijn staat met de positie van de jumper.

Opgeladen condensatoren

Het is mogelijk, dat een te meten condensator nog opgeladen is. Het is aan te bevelen om hem dan eerst te ontladen door de condensator even kort te sluiten. In de schakeling van figuur 2 is voorzien in een beveiliging met behulp van 100 Ω en 82 Ω serieweerstand naar de ingang, gecombineerd met twee diodes BAW62 naar aarde en voeding. Eventuele lading wordt zo naar aarde of naar de +5V afgevoerd. Normaal staan beide diodes gesperd.

Invloed van de beveiliging

Bij gebruik van de laagste waarde voor RX verhindert de 182 Ohm een volledige ontlading van condensator CX in de rustsituatie. Hierdoor wordt de oplaadtijd korter dan 1.1 * RX * CX. Dit effect is gecompenseerd door RX te verhogen van 470 Ω tot 620 Ω. Wat betreft timing gedraagt de schakeling zich dan weer alsof er precies 470 Ω zit. Bij gebruik van de hoogste waarde voor RX kan de lekstroom van de twee diodes van invloed zijn op de timing. Het beste zijn diodes met een gespecificeerd lage lekstroom, zoals bijvoorbeeld de BAW62, BA281, BAS15 of BAS45. Er is overigens ook al enige lekstroom van de ingangen van het IC. Met 4M4 als hoogste waarde voor RX gedroeg het gebruikte IC zich alsof er 4M7 zat.

De Software

Meten van de pulstijd

Zoals in figuur 2 te zien is, heb ik gekozen voor UD7 (MSB bit van het UNIFACE data-uit byte) om de startpuls te geven. Evenzo heb ik gekozen voor UI7 (MSB bit van het UNIFACE data-in byte) om de one shot output binnen te halen. Het principe van de Turbo Pascal routine, die de tijdmeting verzorgt, is dan als volgt:

```
T:=0; Begin met de teller op nul.
PORT[$310]:=128;    Genereer een
PORT[$310]:=0;      startpuls
PORT[$310]:=128;    op UD7.
REPEAT
  T:=T+1;           Verhoog teller met 1.
  DIN:=PORT[$310];  Lees UNIFACE Data-IN byte.
  DIN:=DIN AND 128  Zet bits UI0 t/m UI6 op 0.
UNTIL DIN=0;        Herhaal lus tot bit UI7=0.
```

Na het uitvoeren van deze routine zit in de variabele T het aantal malen, dat de lus is doorlopen.

Met een gegeven RX is hieruit de werkelijke condensatorwaarde af te leiden.

Basisopzet

Zie voor het complete programma de hierbij afgedrukte listing met het commentaar in de tekst. De basisopzet is als volgt:
Het programma geeft de mogelijkheid een meetbereik te kiezen. Na de keuze worden enkele variabelen gevuld met waarden, behorend bij het gekozen bereik. Dan wordt de led van dit bereik aangestuurd. Nu moet de jumper op de juiste positie worden geplaatst en ter bevestiging moet een <return> worden gegeven. Hierna begint het meten.

Verloop van een meting

De procedure METING begint met twee seconden wachten t.b.v. het ontladen van de condensator CX. Daarna wordt de one shot tijd geteld. Hiertoe wordt eerst even de systeemklok gestopt en meteen na de telling weer aangezet. Dit stoppen en starten van de klok wordt geregeld via de Pascal procedures KLKUIT en KLKAAN. De waarde van de condensator wordt vervolgens uitgerekend en in een geschikt formaat op het beeldscherm weergegeven. De meetcyclus wordt net zo lang herhaald, tot er een toets wordt ingedrukt.

De echte tijdsduur

Indien we het aantal malen, dat de tel-lus doorlopen wordt, bijhouden in een variabele van het PASCAL type "WORD", dan kunnen we tellen tot maximaal 65535. Op mijn PC-XT op 10 MHz bleek de tel-lus 18,7 µsec. te duren. De meettijd voor de grootste condensator in ieder bereik duurt dan ruim 1 sec. Hierbij komt een extra wachttijd van 2 µ sec. per meetcyclus.

Anticipatie op problemen

Meten zonder condensator Zoals in figuur 2 te zien is, worden er vier jumperposities gebruikt om vijf meetbereiken te maken. Bereik (1) ontstaat door geen jumper te plaatsen. Hierdoor zal bij meten zonder condensator de telroutine van paragraaf 9 altijd meteen eindigen, want in elke situatie zonder CX is de threshold input van het IC dan hoog. De waarde van de RX voor het tweede bereik is gecompenseerd voor de 4M4, door niet 470K, maar 510K te nemen.

Met Uniface, een kleine elektronische schakeling en een Pascal programma meet de computer de waarde van een condensator.

Een kortgesloten condensator

Bij een condensator met relatief veel lekstroom kan het gebeuren, dat het threshold level van het timer-IC niet bereikt wordt. De tel-routine stopt in dit geval, als de condensator uit de meetvoet wordt gehaald. Dan ontstaat namelijk de situatie zonder condensator. Voor dit lekstroom probleem bestaan ook software oplossingen. Die zijn hier niet gebruikt.

Verstoringen in de meetcyclus

De nauwkeurigheid van de meting hangt af van de nauwkeurigheid, waarmee de telroutine verloopt. Deze routine wordt uitgevoerd onder microprocessor besturing. Een probleem in de PC-XT onder DOS is, dat er op interrupt basis nog een aantal andere zaken worden bijgehouden. Op ieder moment kan de aandacht van de microprocessor worden gevraagd voor een andere klus, waar deze dan zijn huidige karwei even voor onderbreekt. Als dat net de telling van de one shot timing is, krijgen we in dat geval een te lage aanwijzing. Twee mogelijke oorzaken hiervan worden hierna besproken.

Interrupts van de muis

Als de muisdriver memory-resident geladen is tijdens de boot procedure van de computer, dan werkt die altijd. Ook als er geen programma loopt, dat die muis gebruikt. Een kleine beweging van de muis geeft een interrupt af naar de processor, die dan even aandacht aan de muis schenkt. Remedie: veeg de muis-driver uit de `CONFIG.SYS` file en laad die driver alleen als hij echt gebruikt wordt. Zie ook de bij de muis geleverde documentatie.

Interrupts van de klok

Elke 55 millisec. wordt een interrupt gegenereerd door de systeemklok. Hierop wordt onder andere de tijd onder DOS bijgehouden, waar bijvoorbeeld weer wachtlusjes voor het starten en stoppen van de drives mee worden bestuurd. Meettijden korter dan 55 msec. vallen zo, dat er soms wel en soms niet zo'n interrupt invalt. Op mijn P3120 betekende dat, dat de teller kon springen met 11 eenheden. Bij kleine condensatoren gaf dat een grote fout. Door ervoor te zorgen dat, gedurende het tellen van de one shot tijd, de interrupt controller de klok-interrupts negeert, hebben we daar geen last meer van. Het gevolg is wel, dat de tijd in DOS achter gaat lopen.

Aanpassen voor andere computers

De constanten

De gegevens voor de vijf bereiken zijn te vinden in de procedure BERCONST. Dit is de eerste procedure van het programma. Elk bereik wordt met drie gegevens gekarakteriseerd. Eerst C0. Dit is het resultaat van de telprocedure als we geen condensator aansluiten. Dan FC. Dit is de factor. De capaciteit per eenheid van telling in het gekozen bereik. Tot slot LD. Dit is de byte, die het bereik aangeeft. Hiermee wordt de juiste led aangestuurd, de lijn UD7 hoog gehouden en het juiste weergave formaat op de monitor gekozen.

De ijking

C0 en FC kunnen op andere computers heel anders uitvallen. Dit moet daarom bekeken worden op het systeem waar men mee wil meten. Na het invullen van nieuwe waarden in de procedure BERCONST en opnieuw compileren, is het dan goed. De bepaling van C0 is heel simpel: meten zonder condensator. FC kan gemakkelijk worden gevonden door het meten van een bekende condensator. Als FC duidelijk anders wordt, moet ook de tekst in het keuze-scherm aangepast worden. Hier staat namelijk bij benadering de maximaal te meten waarde in een bepaald bereik. De gegevens in het programma zijn gebaseerd op mijn P3120 PC-XT op 10 MHz en geven dan een zeer nauwkeurig en goed reproducerend resultaat. Aanpassen voor andere XT en AT's is niet moeilijk.

MSX en P2000

Voor MSX en P2000 is aansturing ook mogelijk met UNIFACE. Ik kan echter niet overzien of met de gebruikte programmeertalen daar een bruikbare meetroutine te maken is. In machinetaal gaat het echter altijd. Ook zal de problematiek met de interrupts daar anders liggen. De adressering in het UNIFACE-deel is hetzelfde. De I/O adressen zijn dat echter niet. Zie hiervoor de UNIFACE documentatie.

Nabeschouwing

UNIFACE EXPIO kaartje

Een uitvoering op een UNIFACE EXPIO kaartje heb ik als volgt in gedachten:
- Het verlagen van de 4M4 RX bereik-weerstand tot 3M3. Hoewel ik er geen problemen mee heb gehad, blijft de schakeling dan altijd bij nabouwen binnen door de fabrikant opgegeven specificaties.
- Verdelen in vier meetbereiken in plaats van vijf. De bereiken zijn nu ver overlappend. Dit komt

- Verdelen in vier meetbereiken in plaats van vijf. De bereiken zijn nu ver overlappend. Dit komt door het gebruik van een 16-bits integer om de one shot tijd te tellen.
- De indicatie-leds vervangen door reed relais, die zelf de bereiken schakelen. Hiervan zijn er dan drie nodig. De computer kan hierdoor de bereiken zelf sturen. Dit geeft de mogelijkheid van "autoranging" bij het meten.

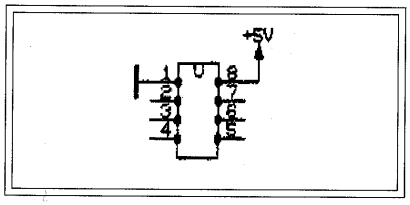
Indien er belangstelling voor bestaat, kan ik aan de uitvoering op een EXPIO-kaartje en de daarbij horende software (over enkele maanden) ook een artikel wijden.

Geraadpleegde documentatie

- UNIFACE - De techniek. Uitgave PTC
- UNIFACE documentatie bij de PC-interface
- Philips Databoek IC11 Lineair Products
- Philips Databoek SC01 Diodes
- Signetics Application Note AN170 (NE555 Applications).

Wil Luijten

Tijdens de Open Dag demonstreerde Wil met een verbeterde schakeling. Er zaten geen jumpers meer in, maar reed-relais. Het apparaat zoekt nu zelf het juiste meetbereik.
Redactie PTC PRINT



figuur 3. De NE555N van boven gezien. Wat niet in de schema's staat: pin 1 moet aan "aarde" en pin 8 aan +5V.

Correctie MSX en de Datum

Enkele lezers signaleerden een foutje in MSX en de Datum (PRINT nr 56, pag 42). Op regel 20 van boven staat Bits 7,4 en 1 zijn "1". Dat moet zijn Bits 7,4 en 0 zijn "1".

```
PROGRAM CAPMETEN; {W. Luijten 19911218. Condensatoren meten op
plaatje                                  met jumpers voor
bereikkeuze, m.b.v. Uniface.
                    Programma in Turbo Pascal 5.0 op P3120
PC-XT}

USES CRT;              {t.b.v. ClrScr, Keypressed, Delay}
VAR  RESPONS: CHAR;            {Antwoord op vraag}
     FACT: REAL;               {Factor cap/count}
     CAPO, BERLED: BYTE;{Counts zonder cap, bereik-Led}

PROCEDURE BERCONST(A:CHAR; VAR FC:REAL; VAR C0,LD:BYTE)
 {Levert de constanten van een bepaald meetbereik.
 De gekozen led zit op de bin.pos. v/d 0 in LD-byte}
 BEGIN
    CASE A OF
     '1' : BEGIN C0:=7; FC:=3.776; LD:=254 END; {FC in pF/count}
     '2' : BEGIN C0:=2; FC:=36.72; LD:=253 END;
     '3' : BEGIN C0:=1; FC:=368.6; LD:=251 END;
     '4' : BEGIN C0:=1; FC:= 3.715; LD:=247 END: {FC in nF/count}
     '5' : BEGIN C0:=1; FC:=36.17; LD:=239 END
    END
 END;

PROCEDURE JUMPER(LED:BYTE); {Plaats jumper op goede bereik}
 BEGIN
    PORT [$310]:=LED;
    WRITE('Zet de jumper bij de oplichtende LED en geef
<ret>.');
    READLN
 END;

PROCEDURE KLKUIT; {Zet de interrupt voor de systeemklok af}
 VAR   V:BYTE;
 BEGIN     V:=PORT[$21];
           V:=V AND 254;
           PORT[$21]:=V END;

PROCEDURE TEL(BLD:BYTE; VAR N:WORD); {Meet timing van 555 one
shot}
 VAR DIN:BYTE;   {Uniface data-in byte}
    T:WORD;   {Lokale variabele voor teller}
 BEGIN
    KLKUIT;
    T:=0; {Teller op 0 beginnen}
    PORT[$310]:=0;      {"0" op UD7 start IC-timer}
    PORT[$310]:=BLD;    {UD7="1", LED aan}
    REPEAT
       T:=T+1;   {Verhoog teller met 1}
       DIN:=PORT[$310];Lees Uniface data-in byte}
       DIN:=DIN AND 128{Zet bits UI0 t/m UI6 op 0}
    UNTIL DIN=0; {Herhaal de lus tot UI7=0}
    KLKAAN;
    N:=T  {T naar var-parameter.}
 END;
```

```
PROCEDURE VERTOON(LED:BYTE; COND:REAL);
  BEGIN
    IF LED=254 THEN WRITELN('Condesator is',COND:8:0,'pF');
    IF LED=253 THEN BEGIN COND:=COND/1000;
      WRITELN('Condensator is',COND:8:1,'nF') END;
    IF LED=251 THEN BEGIN COND:=COND/1000;
      WRITELN('Condensator is',COND:8:0,'nF') END;
    IF LED=247 THEN BEGIN COND:=COND/1000;
      WRITELN('Condensator is",COND:8:2,'uF') END;
    IF LED+239 THEN BEGIBN COND:=COND/1000;
      WRITELN('Condensator is',COND:8:1,'uF') END
  END;

PROCEDURE METING(LED,ZVL:BYTE; FCT:REAL); {Zero VaLue = C0}
  VAR N:WORD;{Tellen tot max. 65535}
    COND:REAL; {Condensatorwaarde}
    D:CHAR; {Dummy om keypress char kwijt te raken}
  BEGIN
    REPEAT
      DELAY(2000); {Wacht om cap. te ontladen}
      TEL(LED,N);{Tel timing}
      WRITE(N:6,' counts. ');
      N:=NN-ZVL; {Trek C0 eraf}
      COND:=FCT*N; {Condensator uitrekenen}
      VERTOON(LED,COND); {Schrijf op de monitor}
    UNTIL KEYPRESSED;
    D:=READKEY {Ledig keyboeard buffer na keypressed}
  END;

BEGIN
  CLRSCR; {Beginnen met schone lei}
  PORT[$310]:=255;{Triggerlijn"1" en alle Leds uit}
  REPEAT
      WRITELN;
      WRITELN('Meten tot circa 220nF, geef keuzel.');
      WRITELN('Meten 2u2 2.');
      WRITELN('Meten    22uF  3.');
      WRITELN('Meten    220uF  4.');
      WRITELN('Meten    2200uF 5.');
      WRITELN('Stoppen, geef keuze S)top of Q)uit');
      WRITE('Tik je keuze in. (1,2,3,4,5,Q of S) -> ');
        READLN(RESPONS);
      IF RESPONS IN ['1','2','3','4','5'] THEN
      BEGIN
        BERCONST(RESPONS,FACT,CAPO,BERLED);
          {Constanten van meetbereik}
        JUMPER(BERLED); {Zet jumper bij bereik}
        METING(BERLED,CAPO,FACT)      {Meet de cap}
      END;
    UNTIL RESPONS IN ['S','s','Q','q'];
  PORT[$310]:=255 {Meetkaart netjes achterlaten}
END.
```