CODEN:LUTFD2/(TFRT-5278)/1-057/(1982)

# PI-ALGORITMER VID DIGITAL REGLERING

JAN HENRIKSSON

INSTITUTIONEN FÖR REGLERTEKNIK
LUNDS TEKNISKA HÖGSKOLA

AUGUSTI 1982

LUND INSTITUTE OF TECHNOLOGY
DEPARTMENT OF AUTOMATIC CONTROL
Box 725
S 220 07 Lund 7 Sweden

Document name
Report

Date of issue
August 1982

Document number
CODEN: LUTFD2/(TFRT-5278)/1-057/(1982)

Author(s)
Jan Henriksson

Supervisor
D Renborg, J Ancker, B Wittenmark

Sponsoring organization

Title and subtitle
Digital algorithms for PI-control
(PI-algoritmer vid digital reglering)

Abstract
In this report a theoretical and an experimental investigation is made of different types of PI-algorithms in digital control. Four different ways to translate transfer functions from the Laplace- and to the 2-transform are investigated. Six different programs for PI-controllers are presented, and are tested with different sample times and constants. The comparison is made for different types of transfer functions. Setting time, overshoots, static errors and "reset windups" are compared.

Key words

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title

ISBN

Language
Swedish

Number of pages
57

Recipient's notes

Security classification

**ASEA**

Teknisk rapport
Technical Report

**TR** YTK 82-012


Författare - Author
Jan Henriksson
Godkännare - Approved by
D Renborg, J Ancker
Uppdragsgivare - Requested by
J Ancker


Titel - Title
PI-algoritmer vid digital reglering.
Examensarbete.

Från - From: YTKU
Datum - Date: 82-08-05

[x] Utredning, teoretisk undersökning - Analysis, theoretical investigation
[x] Provning, experim. undersökning - Test, experimental investigation
[ ] Delrapport
[x] Slutrapport
Provning/undersökning avslutad
Test/investigation finished

Reg.: 0159, 5701
Sida - Page: 1
Ordernr - Ref. No.
Debiteras ordernr
Pkl/Akl
Antal textsidor - No. of pages of text: 37
Antal bilagesidor - No. of supplem. pages: 20


Sammanfattning - Summary

DENNA HANDLING FÅR EJ DELGIVAS ANNAN UTAN GODKÄNNARENS MEDGIVANDE.

I detta examensarbete studeras och testas olika PI-algoritmer vid digital reglering.

Syftet med arbetet var dels att göra en teoretisk genomgång om överföringen mellan Laplace- och z-transformen, dels att realisera ett antal olika PI-regulatorer i assembler för 6801, som sedan testas.

Rapporten behandlar fyra olika överföringar mellan Laplace- och z-transformen, nämligen impulsinvariant transformation, enkel differentiering, bilinjär transformation och den s k utvidgade bilinjära transformationen.

Sex olika program för PI-regulatorer presenteras. Funktionen av de olika programmen testas inbördes. Testerna sker mot 1:a och 2:a ordningens system. Inverkan av integratormättning visas. Jämförelser görs för olika överföringar mellan Laplace- och z-transformen. Regleringarnas prestanda jämförs för små och stora steg, samt för olika sampeltider och konstanter. Insvängningstid, överslängar samt eventuella statiska fel observeras. Dessutom jämförs algoritmerna med hjälp av en särskild programdel, som mäter integralen av felet.

Hårdvaran har bestått av ett färdigt kretskort (EVM) samt A/D- och D/A-omvandlare.

En viktig slutsats man kan dra av testerna är att man bör försöka undvika att använda villkorlig integration som integratormättning. Dessutom visade sig transformen med enkel differentiering ge något bättre resultat än de övriga.




Distribution

YTKU-arkiv (cirk YTK-kontor), förf (3)

Enbart sida 1 - Page 1 only

KSB YTK, YTKB, YTKC, TKU, YLB, YLK

| Nyckelord - Ämnesord | Keywords | Övriga nyckelord |
|---|---|---|
| Digital reglering<br>PI-regulator | | |

Anm. Huvudregeln är att nyckelorden skall skrivas på svenska i vänstra kolumnen.
I kolumnen för keywords införs då så önskas engelska motsvarigheter till ämnesorden samt engelska uttryck utan svensk motsvarighet.
Med övriga nyckelord avses t ex materialbeteckning, produktbeteckning, leverantör, kund, etc.


6917 0300-BA (930) 80-04 8000 Ansvarig: KSB

Innehållsförteckning

# 1 Teoretisk genomgång om överföring mellan Laplace- och z-transformerna

## 1.1 Inledning

Genom att först söka en analog systemfunktion H(s), som uppfyller våra ställda krav, så kan vi sedan transformera denna till en diskret systemfunktion H(z). Viktigt är då att om vi har en stabil analog systemfunktion, så skall vi även få en stabil tidsdiskret systemfunktion. Detta innebär att vänstra s-halvplanet skall avbildas inom enhetscirkeln i z-planet. Dessutom bör även den tidsdiskreta systemfunktionen H(z) uppfylla de ställda kraven i frekvens- eller tidsplanet.

Fyra olika metoder behandlas, nämligen:

1. Impulsinvariant transformation
2. Enkel differentiering
3. Bilinjär transformation
4. Utvidgad bilinjär transformation

### Impulsinvariant transformation

Denna metod innebär att man utgående från det analoga impulssvaret $h^c(t)$ bildar det tidsdiskreta pulssvaret h(k). Metoden kräver att den analoga systemfunktionen H(s) har enkla poler, samt att täljarens gradtal är mindre än nämnarens gradtal. De analoga tidsegenskaperna bibehålls, men det finns däremot risk för att frekvensegenskaperna ändras (s k vikning).

### Enkel differentiering

Denna metod är snarlik den bilinjära transformationen. Vi får även här s som funktion av z, som vi sedan substituerar.

### Bilinjär transformation

Metoden utgår ifrån att vi avbildar s-planet på z-planet med transformationen $z = e^{sT_s}$. Vi använder däremot endast den första termen i serieutvecklingen av $s^{-1}$. Detta gör att vi får s som funktion av z. Sedan fås den tidsdiskreta tidsfunktionen genom en rent algebraisk substitution. Vid denna metod slipper vi problemet med vikningsdistorsion. Däremot finns det risk för frekvensdistorsion, som medför restriktioner, när metoden kan användas.

### Utvidgad bilinjär transformation

Till skillnad ifrån den bilinjära transformationen så tar vi med de två första termerna i serieutvecklingen av $s^{-1}$.

1.2
Impulsinvariant transformation

Den analoga systemfunktionen H(s) partialbråksuppdelas

$$H(s) = \frac{d_o+d_1 s+d_2 s^2+\ldots\ldots+d_m s^m}{1+c_1 s+c_2 s^2+\ldots\ldots c_n s^n} = \frac{e_1}{s-p_1} + \frac{e_2}{s-p_2} + \ldots\ldots + \frac{e_n}{s-p_n}$$

Nödvändiga förutsättningar för att metoden skall fungera är att: H(s) har enkla poler $p_i$, i = 1...n och att m < n

$$\text{Impulssvaret } h^c(t) = \frac{1}{j2\pi}\int_{-\infty}^{\infty} H(s)e^{st}\,ds$$

$$h^c(t) = \sum_{i=1}^{n} h_i(t) = \sum_{i=1}^{n}\left(\frac{1}{j2\pi}\int_{-\infty}^{\infty}\frac{e_i}{s-p_i}e^{st}\,ds\right) = \sum_{i=1}^{n} e_i e^{p_i t}\cdot v(t)$$

Det tidsdiskreta pulssvaret h(k) bildas genom:

$$h(k) = \begin{cases} T_s \cdot h^c(kT_s) & k \geq 0 \\ 0 & k < 0 \end{cases}$$

$$h(k) = \sum_{i=1}^{n} h_i(k) \quad \text{där} \quad h_i(k) = T_s h_i^c(kT_s) = \begin{cases} T_s e_i e^{p_i k T_s} & k \geq 0 \\ 0 & k < 0 \end{cases}$$

Den tidsdiskreta systemfunktionen H(z) blir:

$$H(z) = \sum_{i=1}^{n} H_i(z) \qquad \text{där} \qquad H_i(z) = \sum_{k=0}^{\infty} h_i(k) \cdot z^{-k} = \frac{T_s \, e_i}{1 - e^{\beta_i T_s} z^{-1}}$$

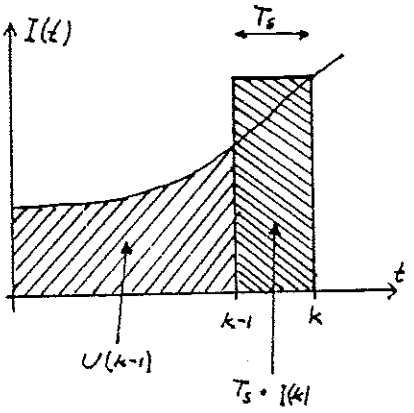
$$H(z) = \frac{T \cdot e_1}{1 - e^{\beta_1 T_s} \cdot z^{-1}} + \cdots \cdots \frac{T \cdot e_n}{1 - e^{\beta_n \cdot T_s} z^{-1}}$$

Vid denna transformation tar man inte hänsyn till en eventuell hållkrets, utan man tittar bara på impulssvaret.

## 1.3 Enkel differentiering

Om vi har en insignal I(t) som vi integrerar så är

$$H(s) = \frac{1}{s}$$




Vid enkel differentiering så adderar vi $T_s \cdot I(k)$ till det gamla värdet U(k-1) för att erhålla den nya tidsdiskreta utsignalen U(k).

$$U(k) = U(k-1) + I(k) \cdot T_s$$

$$U(k) - U(k-1) = I(k) \cdot T_s$$

$$(1 - z^{-1}) \cdot U(z) = I(z) \cdot T_s$$

$$\frac{U(z)}{I(z)} = T_s \frac{1}{1 - z^{-1}} = \frac{1}{s}$$

$$\Rightarrow \quad s = \frac{1}{T_s} (1 - z^{-1})$$

Den tidsdiskreta systemfunktionen H(z) erhålles ur den analoga systemfunktionen H(s) genom substitutionen

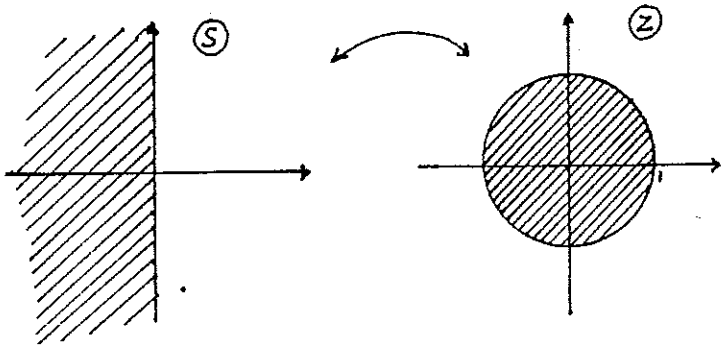
$$H(z) = H(s) \Big|_{s = \frac{1}{T_s}(1 - z^{-1})}$$

1.4
Bilinjär transformation

Eftersom $z \equiv e^{sT}$ och $e^{sT} = 1 + sT + \frac{(sT)^2}{2^1} + \frac{(sT)^3}{3^1} + \ldots$

så blir $s^{-1} = \frac{T}{2}\left[\frac{1}{v} - \frac{v}{3} - \frac{4v^3}{45} - \frac{44\,v^5}{945} - \ldots\right]$ där $v = \frac{1-z^{-1}}{1+z^{-1}}$

Vänstra s-halvplanet avbildas inom enhetscirkeln i z-planet




Vid bilinjär transformation så medtages endast första termen i serieutvecklingen av $s^{-1}$

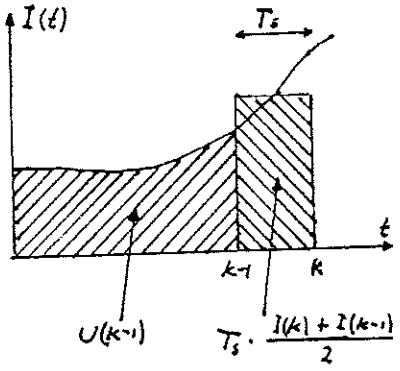
$$s^{-1} = \frac{T_s}{2} \cdot \frac{1}{v} = \frac{T_s}{2} \cdot \frac{1}{\frac{1-z^{-1}}{1+z^{-1}}} = \frac{T_s}{2} \cdot \frac{1+z^{-1}}{1-z^{-1}} \Longrightarrow \underline{s = \frac{2}{T_s} \cdot \frac{1-z^{-1}}{1+z^{-1}}}$$

Genom substitutionen $H(z) = H(s)\Big|_{s = \frac{2}{T_s} \cdot \frac{1-z^{-1}}{1+z^{-1}}}$ så erhålles

den tidsdiskreta systemfunktionen H(z).

Om vi har en insignal I(t) som vi integrerar så är

$$H(s) = \frac{1}{s}$$




$$\frac{1}{s} = \frac{T_s}{2} \cdot \frac{1+z^{-1}}{1-z^{-1}}$$

$$U(z) = I(z) \cdot \frac{T_s}{2} \cdot \frac{1+z^{-1}}{1-z^{-1}}$$

$$(1-z^{-1}) \cdot U(z) = I(z) \cdot \frac{T_s}{2} (1+z^{-1})$$

$$U(k) - U(k-1) = \frac{T_s}{2} (I(k) + I(k-1))$$

$$U(k) = U(k-1) + T_s \cdot \frac{I(k)+I(k-1)}{2}$$

Vid bilinjär transformation så tar vi alltså medelvärdet av I(k) och I(k-1) och multiplicerar med $T_s$.

## 1.5 Utvidgad bilinjär transformation

Vid denna transformation så medtages de två första termerna i serieutvecklingen $s^{-1} = \frac{T_s}{2} \left( \frac{1}{v} - \frac{v}{3} - \frac{4v^3}{45} - \frac{44v^5}{954} + \ldots \right)$,

där $v = \frac{1-z^{-1}}{1+z^{-1}}$

$$\Rightarrow s^{-1} = \frac{T_s}{2} \left( \frac{1}{v} - \frac{v}{3} \right) = \frac{T_s}{2} \left( \frac{1+z^{-1}}{1-z^{-1}} - \frac{1-z^{-1}}{3+3z^{-1}} \right) = \frac{T_s}{3} \cdot \frac{1+4z^{-1}+z^{-2}}{1-z^{-2}}$$

$$s = \frac{3}{T_s} \cdot \frac{1-z^{-2}}{1+4z^{-1}+z^{-2}}$$

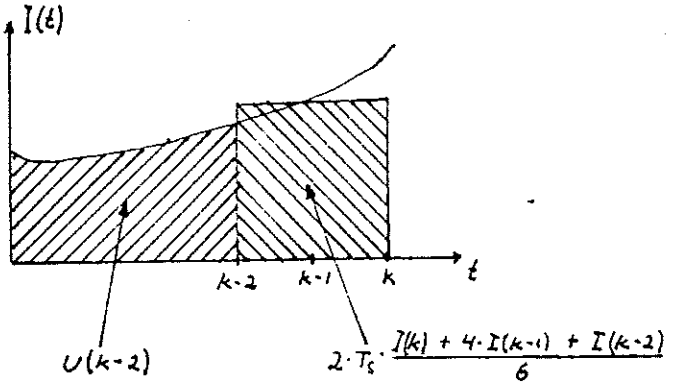
---

För att erhålla H(z) så gör vi substitutionen

$$H(z) = H(s) \Big|_{s = \frac{3}{T_s} \cdot \frac{1-z^{-2}}{1+4z^{-1}+z^{-2}}}$$

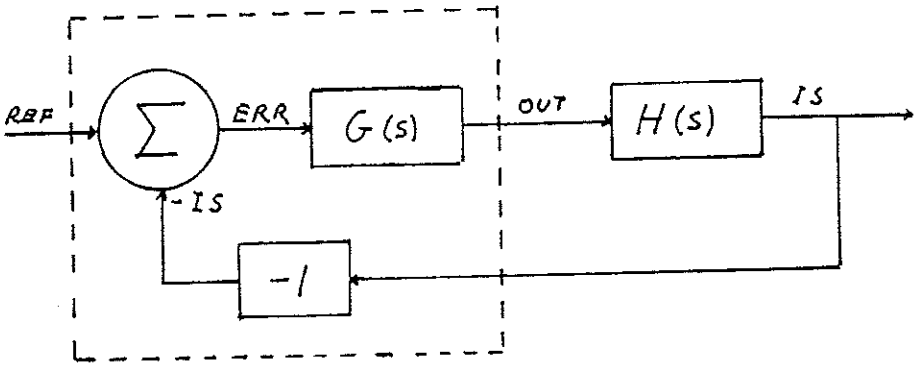
Genom att integrera en insignal I(t) så erhåller vi:

$$U(k) = U(k-2) + \frac{T_s}{3} \left( I(k) + 4(I(k-1) + I(k-2) \right)$$




Vid utvidgad bilinjär transformation så adderar vi ett viktat medelvärde från de tre sista samplen från insignalerna och multiplicerar med $2 \cdot T_s$. Dessutom adderar vi inte till det senaste integralvärdet utan det från gången innan.

1.6
Tillämpningar på en PI-regulator




För en PI-regulator är överföringsfunktionen $G(s) = K\left(1 + \frac{1}{sT_i}\right)$

Impulsvariant transformation

$$G_{pI}(s) = K\left(1 + \frac{1}{sT_i}\right) = \frac{K \cdot s \cdot T_i + K}{sT_i} = \frac{K + K \cdot T_i \cdot s}{T_i \cdot s}$$

Metoden går ej att använda ty m = n = 1, alltså är ej gradtalet på täljaren mindre än gradtalet på nämnaren, vilket krävs.

## Enkel differentiering

$$S = \frac{1}{T_s}\left(1 - z^{-1}\right)$$

$$G_{PI}(z) = G_{PI}(s)\Big|_{s=\frac{1}{T_s}(1-z^{-1})} = K\left[1 + \frac{T_s}{T_i}\cdot\frac{1}{(1-z^{-1})}\right] = K\left[\frac{1 + \frac{T_s}{T_i} - z^{-1}}{1 - z^{-1}}\right]$$

$$G_{PI}(z) = \frac{OUT(z)}{ERR(z)}$$

$$OUT(z) = K\left[1 + \frac{T_s}{T_i}\right]\cdot ERR(z) - K\cdot z^{-1}\cdot ERR(z) + z^{-1}\cdot OUT(z)$$

## Bilinjär transformation

$$S = \frac{2}{T_s}\cdot\frac{1 - z^{-1}}{1 + z^{-1}}$$

$$G_{PI}(z) = G_{PI}(s)\Big|_{s=\frac{2}{T_s}\cdot\frac{1-z^{-1}}{1+z^{-1}}} = K\left[1 + \frac{1}{2}\frac{T_s}{T_i}\cdot\frac{1+z^{-1}}{1-z^{-1}}\right] = K\left[\frac{1 - z^{-1} + \frac{1}{2}\frac{T_s}{T_i}(1+z^{-1})}{1 - z^{-1}}\right] =$$

$$= K\left[\frac{\left(1 + \frac{1}{2}\frac{T_s}{T_i}\right) + \left(\frac{1}{2}\frac{T_s}{T_i} - 1\right)\cdot z^{-1}}{1 - z^{-1}}\right]$$

$$G_{PI}(z) = \frac{OUT(z)}{ERR(z)}$$

$$OUT(z) = K\left(1 + \frac{1}{2}\frac{T_s}{T_i}\right)ERR(z) + K\left(\frac{1}{2}\frac{T_s}{T_i} - 1\right)\cdot z^{-1}\cdot ERR(z) + z^{-1}\cdot OUT(z)$$

6917 0907-AB (999) 81.12 100000 Ansvarig: CKRB

Utvidgad bilinjär transformation

$$S = \frac{3}{T_s} \cdot \frac{1 - z^{-2}}{1 + 4z^{-1} + z^{-2}}$$

$$G_{PI}(z) = G_{PI}(s)\Big|_{s = \frac{3}{T_s} \cdot \frac{1 - z^{-2}}{1 + 4z^{-1} + z^{-2}}} = K\left(1 + \frac{T_s}{3T_i} \cdot \frac{1 + 4z^{-1} + z^{-2}}{1 - z^{-2}}\right) =$$

$$= K\left(\frac{\left(1 + \frac{T_s}{3T_i}\right) + \frac{4}{3}\frac{T_s}{T_i} \cdot z^{-1} + \left(\frac{T_s}{3T_i} - 1\right) \cdot z^{-2}}{1 - z^{-2}}\right)$$

$$OUT(z) = K\left(1 + \frac{T_s}{3T_i}\right) ERR(z) + K \cdot \frac{4}{3}\frac{T_s}{T_i} \cdot z^{-1} \cdot ERR(z) + K\left(\frac{T_s}{3T_i} - 1\right) z^{-2} ERR(z) + z^{-2} \cdot OUT(z)$$

## 2
## Hårdvarubeskrivning

### 2.1
### Beskrivning av EVM

MEX 6801 EVM Microcomputer Evaluation Module är ett hård- och mjukvarusystem för utvecklingsarbete med mikrodatorn MC 6801. På ett enda kretskort ryms systemets hårdvara, se fig 2.1.a. Det finns plats för:

- Microprocessorn MC 6801 L1
- 4k-byte RAM-minne
- 2k-byte ROM-minne
- Adresslatch och adressavkodare
- ACIA
- PTM
- Wire-wrap area

### 2.2
### Beskrivning av timern

MC 6801 innehåller en "on-chip" programmerbar timer, se fig 2.2.a. I detta fall används timern endast till avbrotts-generering.

Nyckeldelen i timern är en 16-bitars kontinuerligt gående räknare (Free Running Counter, FRC), vilken stegas upp vid varje negativ halvpuls från MPU klockan (1,2 MHz). Räknarens värde kan när som helst läsas av mjukvaran.

Varje gång FRC stegas upp så jämförs värdet med ett 16-bitars register (Output Compare Register, OCR). När FRC och OCR funnits vara lika sätts en flagga (OCF) i "Timer Control and Status Register" TCSR. Detta register består av 8 bitar, som alla är läsbara, men endast de 5 lägsta bitarna är skrivbara.

För att möjliggöra avbrott skall man skriva $Ø8 till TCSR. Detta gör att IRQ2 kommer att genereras då bit 6 (OCF) blir hög.

## 2.3 A/D-omvandlare och D/A-omvandlare

Genom att skriva i minnescell 2ØØØ-2ØØ7 kan val av ingångskanal samt start av A/D-omvandlaren ske. Resultatet kan läsas efter ca 30 µs genom läsning av cellerna 200C och 200D. Resultatet ges 2'komplement representation enligt följande:

| +/- | $R_8$ | $R_7$ | $R_6$ | $R_5$ | $R_4$ | $R_3$ | $R_2$ | cell 2ØØC |
|---|---|---|---|---|---|---|---|---|
| $R_1$ | $R_0$ | 0 | 0 | 0 | 0 | 0 | 0 | cell 2ØØD |

Mellan start av A/D och läsning av resultatet måste minimum en tid av 30-35 µs ha förflutit för att erhålla rätt resultat.

De båda D/A-omvandlarna laddas med digitaldata, enligt samma form som A/D, genom skrivning i cellerna 2ØØ8-2ØØ9 (DA1) och 2ØØA-2ØØB (DA2).

## 2.4 Inställning av DIP-switchar

Totalt finns 4 st DIP-switchar, som kan läsas på adress $3ØØC-$3ØØF. Av dessa används 3 st till att ställa in TSAMP, K och $T_s/T_i$. Dessutom används den 4:e i PIPROG 5 till att ställa in K1.

DIP 1

Inställning av sampeltiden, TSAMP, i millisekunder:

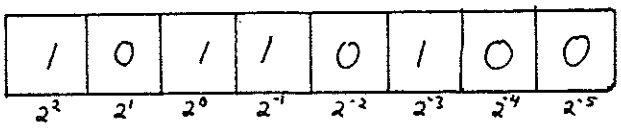
| 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
|---|---|---|---|---|---|---|---|
| $2^3$ | $2^2$ | $2^1$ | $2^0$ | $2^{-1}$ | $2^{-2}$ | $2^{-3}$ | $2^{-4}$ |

TSAMP < 16 millisekunder
Ex: Här är
TSAMP = 11,1875 ms

(Högerskift 4 steg i programmet)

DIP 2 och DIP 4

Inställning av förstärkningsfaktorerna K och K1



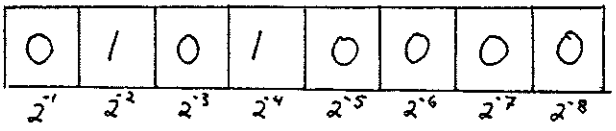

$K < 8$, $K1 < 8$

Ex: Här är
$K = 5{,}625$

(Högerskift 5 steg i programmet)

DIP 3

Inställning av kvoten mellan sampeltiden och integrationstiden $T_s/T_i$:

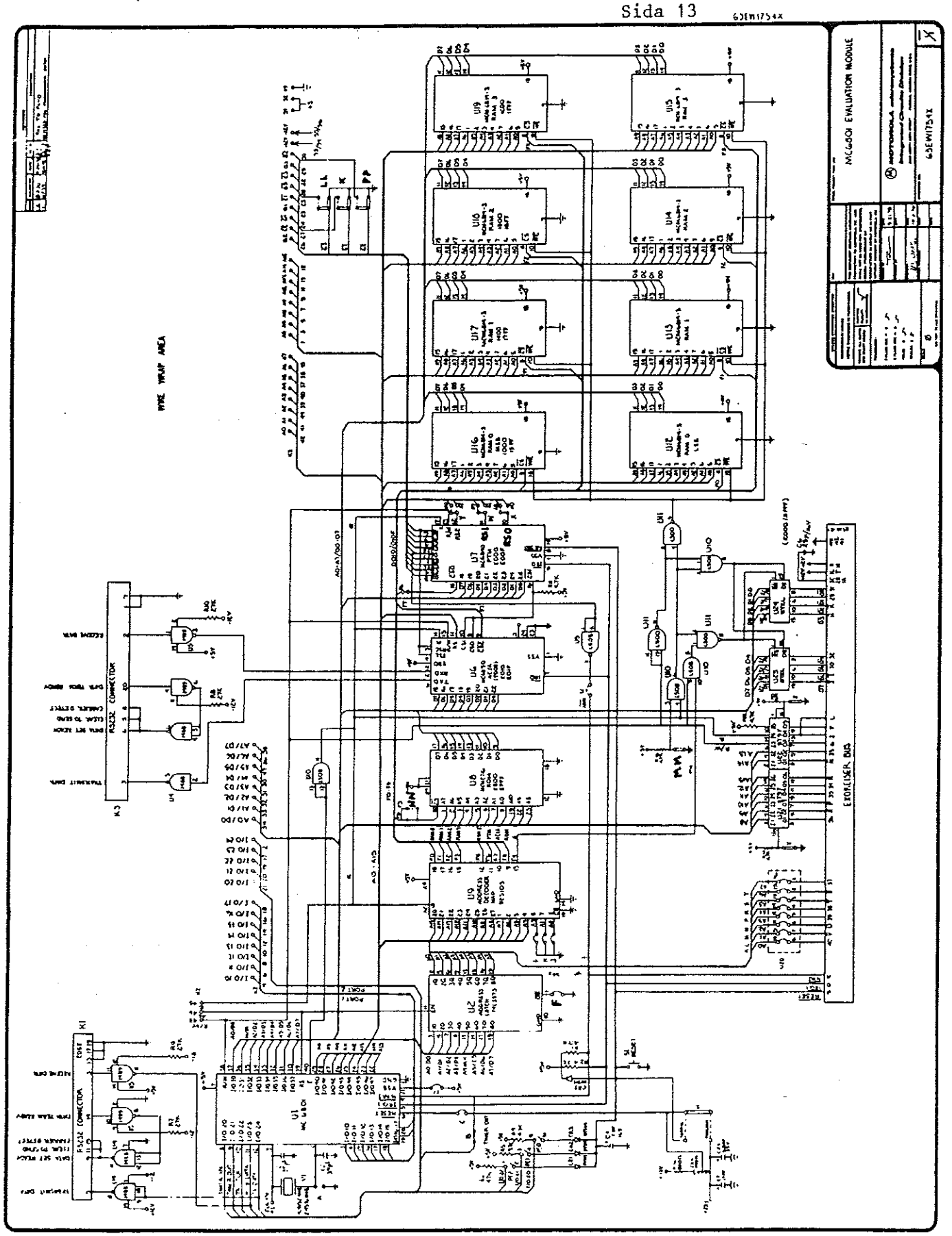



$T_s/T_i < 1$

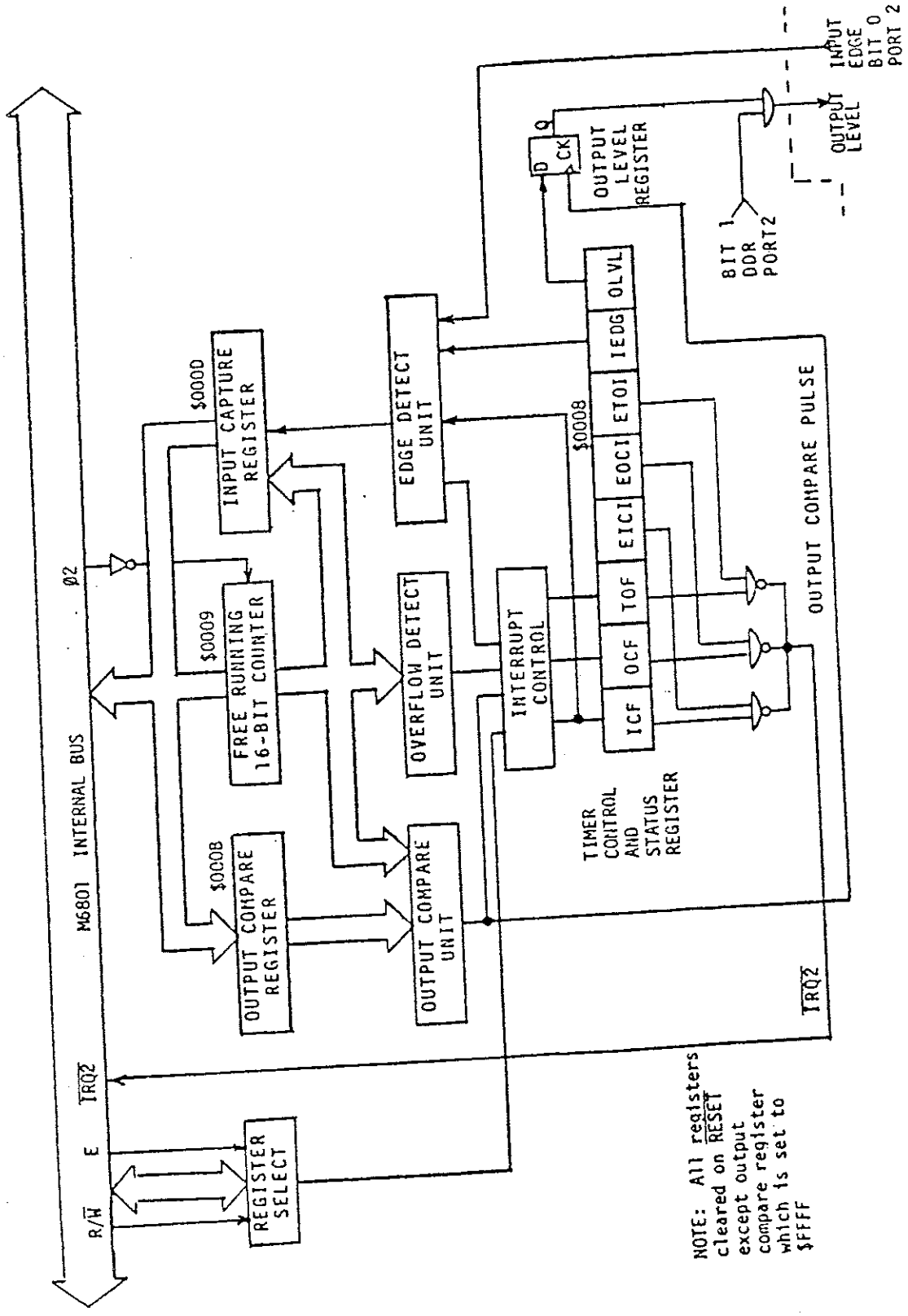
Ex: Här är
$T_s/T_i = 5/16$

(Högerskift 8 steg i programmet)

6917 0907-AB (999) 81.12 100000 Ansvarig: CKRB


MC6809 EVALUATION MODULE
MOTOROLA
WIRE WRAP AREA
EXORCISER BUS

Figur 2.1a


MC6801 Programmable Timer
M6801 INTERNAL BUS
Ø2
E
R/W
IRQ2
REGISTER SELECT
FREE RUNNING 16-BIT COUNTER
$0009
INPUT CAPTURE REGISTER
$000D
OUTPUT COMPARE REGISTER
$000B
OUTPUT COMPARE UNIT
OVERFLOW DETECT UNIT
EDGE DETECT UNIT
INTERRUPT CONTROL
TIMER CONTROL AND STATUS REGISTER
$0008
ICF
OCF
TOF
EICI
EOCI
ETOI
IEDG
OLVL
OUTPUT COMPARE PULSE
OUTPUT LEVEL REGISTER
D
CK
Q
BIT 1 DDR PORT2
OUTPUT LEVEL
INPUT EDGE BIT 0 PORT 2
NOTE: All registers cleared on RESET except output compare register which is set to $FFFF

Figur 2.2 a

# 3 Praktiska aspekter på PI-regulatorn

## 3.1 Integratormättning

Ibland kan man få problem med integraldelen i en PI-regulator. Om man har ett trögt system och om man gör stegvisa ändringar i referensvärdet, så hinner integraldelen bli stor, innan felet blir litet. Den stora integraldelen gör sedan att utsignalen fortsätter förbi sitt önskade värde och vi får nu ett stort fel med motsatt tecken. Vi får alltså stora överslängar. Problemet uppstår alltså pga man har en begränsad styrsignal.

Integraldelen fyller ju främst funktionen att ta bort stationära fel och behövs egentligen inte när reglerfelet är stort. Integraldelen är mest användbar till att justera för små reglerfel.

Problemet med integraldelen kan lösas med villkorlig integration. Detta innebär att vi har integraldelen inkopplad endast om felet är mindre än ett visst givet tal. Denna metod har dock en uppenbar nackdel. Om vi har en P-del som gör att vi får ett för stort fel, kopplas ju aldrig integraldelen in. Detta gör att vi får ett stationärt fel.

Ett annat sätt att komma åt problemet med integraldelen är att hela tiden justera integraldelen så att regulatorns utsignal överensstämmer med begränsningen. Med denna metod slipper vi de nackdelar, som villkorlig integration innebär.

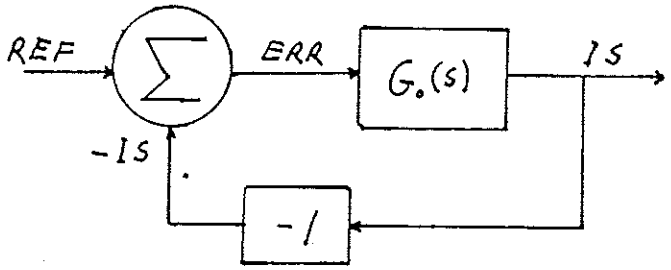
## 3.2 Skalning

För att undvika att få stora kvantiseringsfel så bör normala variationer av signalen gå över hela A/D-omvandlarens mätområde. Annars utnyttjar man ju inte hela noggrannheten hos A/D-omvandlaren.

I detta fall har vi en A/D-omvandlare, som i sitt skalområde kan registrera en mätsignal mellan 0 och $\pm$ 10 V. Vid 10-bitars upplösning ger detta en noggrannhet på $1/2^{10} \approx 0{,}001$, dvs 0,01 V upplösning. Antag att signalen endast varierar inom området 5,0-5,5 V. Då blir ju A/D-omvandlarens upplösning endast $1/50 \sim 1/2^{6}$, dvs motsvarande 6-bitars upplösning.

Det är önskvärt att omvandla de mätvärden som presenteras till lämpliga måttenheter. Detta kan ske om man multiplicerar med en skalningskonstant. Dessutom bör man justera för eventuella nollpunktsfel.

3.3
Villkorlig integration




$G_o(s)$ hos PI-regulatorn = $K\left(1 + \frac{1}{T_i \cdot s}\right)$

$G_o(s)$ hos P-regulatorn = K

Det slutna systemets överföringsfunktion = $\frac{IS(s)}{REF(s)} = \frac{G_o(s)}{1 + G_o(s)}$

Det bestående felet $E(s) = IS(s) - REF(s) = \frac{1}{1 + G_o(s)} \cdot REF(s)$

Hos P-regulatorn är $G_o(s) = K$

$\Longrightarrow E(s) = \frac{1}{1 + K} \cdot REF$

Sätt $EPS = \alpha \cdot \frac{1}{1 + K} \cdot REF \qquad (\alpha \geq 1)$

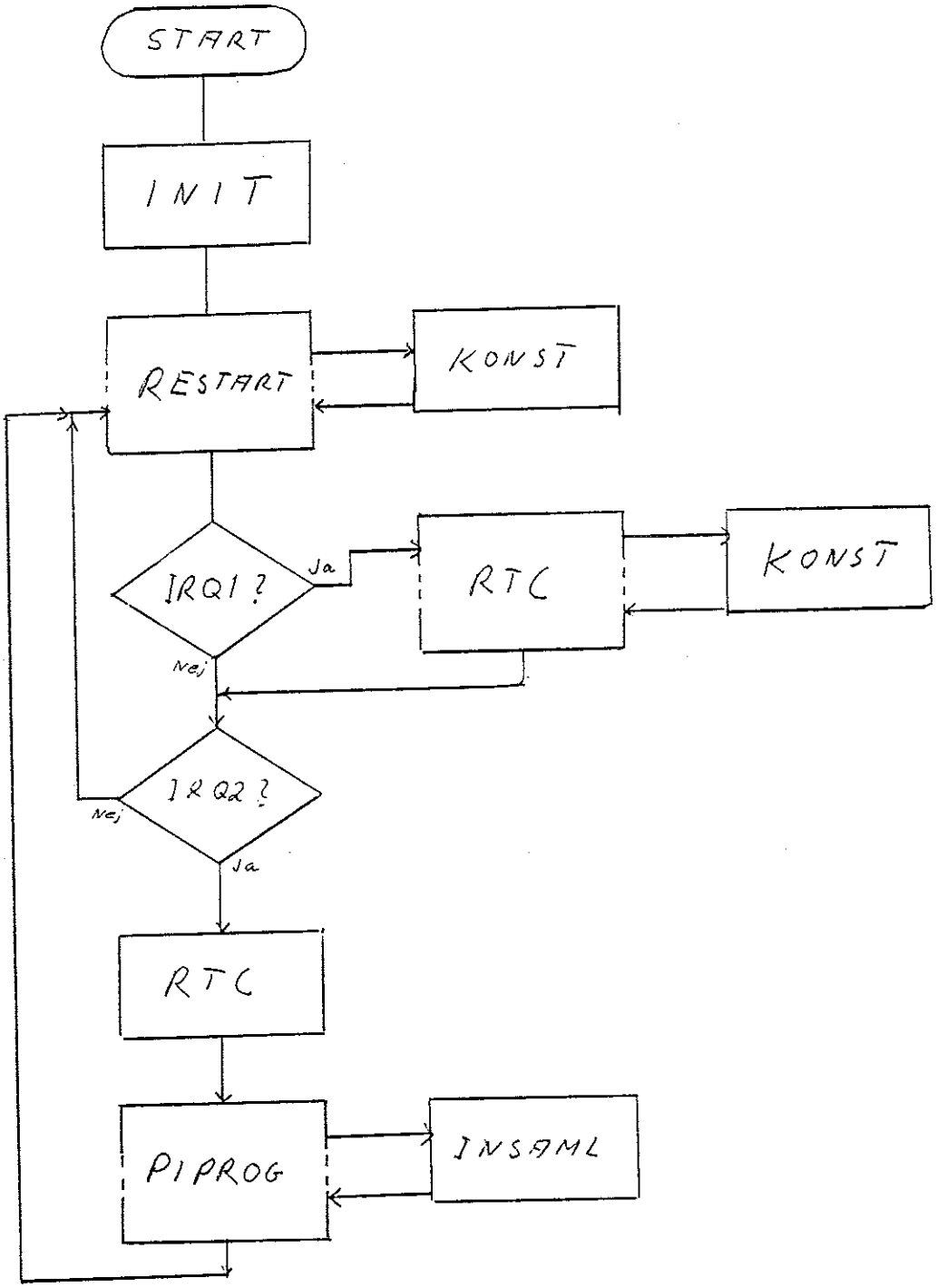
Vid villkorlig integration hos PI-regulatorn så integrerar vi endast om ERR < EPS.

Problemet är här att man inte får låta EPS vara för litet, ty då integrerar vi aldrig och vi får ett statiskt fel. Detta är den stora nackdelen med denna metod för integrator-mättning.

6917 0907-AB (999) 81.12 100000 Ansvarig: CKRB

# 4
# Mjukvarubeskrivning

## 4.1
## Flödesschema över programstrukturen




6917 0907-AB (999) 81.12 100000 Ansvarig: CKRB

4.2
Beskrivning av de olika programdelarna

INIT: Här definieras alla hårdvaruadresser, avbrottsvektorn definieras, timern initieras. Dessutom definieras de två macroprogrammen samt en del globala parametrar.

RESTART: Nollställning sker av en del variabler, initiering av nästa clock-avbrott sker. Här finns loopen, där väntan på timeravbrott sker.

KONST: Denna subroutine anropas dels vid uppstartning av programmet i RESTART och dels genom IRQ1-avbrott. Här avläses de tre DIP-switcharna. Sampeltiden omräknas till antalet klockpulser på realtidsklockan. Dessutom nollställs här en del variabler.

RTC: Här behandlas de två avbrottsrutinerna IRQ1 och IRQ2. Vid IRQ1-avbrott, som är ett avbrott från en bistabil vippa, så anropas subroutinen KONST för erhållande av nya konstantvärden. Vid IRQ2-avbrott, som är timeravbrottet, initieras först nästa clockavbrott, sedan sker hopp till regulatorprogrammet.

INSAML: Denna subroutine anropas av regulatorprogrammet och här avläses de två A/D-omvandlarnas värde, som sedan lagras i minnet.

PIPROGØ: Detta regulatorprogram utnyttjar enkel differentiering vid transformering från Laplace- till z-transform. I detta program finns ingen mättning på integraldelen.

PIPROG1: Här används enkel differentiering som transformering mellan Laplace- och z-transform. Integraldelen mättas genom att den justeras så att den överensstämmer med begränsningen på utsignalen.

PIPROG2: Även här används enkel differentiering som transformering mellan Laplace- och z-transform. I detta program används villkorlig integration som integratormättning. Endast när felet är mindre än 1/2 av referenssignalen så sker integrering.

PIPROG3: I detta program används bilinjär transformation som överföring mellan Laplace- och z-transformen. Integraldelen mättas så att den överensstämmer med begränsningen på utsignalen.

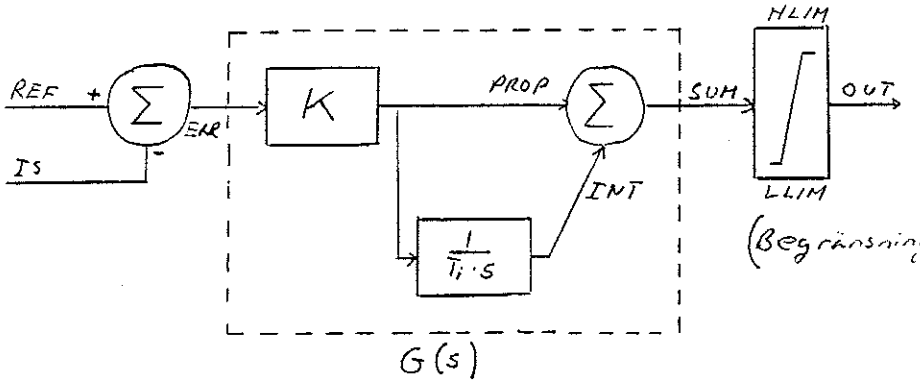
PIPROG4: Detta regulatorprogram utnyttjar den s k utvidgade bilinjära transformationen som överföring mellan Laplace- och z-transformen. Integraldelen mättas så att den överensstämmer med begränsningen på utsignalen.

PIPROG5: I detta program låter man P-delen jobba endast med referenssignalen, och I-delen jobbar endast med felet. Som transformering mellan Laplace- och z-transformen används enkel differentiering. Integraldelen justeras så att den överensstämmer med begränsningen på utsignalen.

MINFEL: Denna programdel ingår inte i själva PI-referensen utan utnyttjas endast för att avgöra vilken PI-algoritm som är "bäst".

4.3
De olika regulatorprogrammen i "Pascal-liknande notation"

## PIPROG Ø



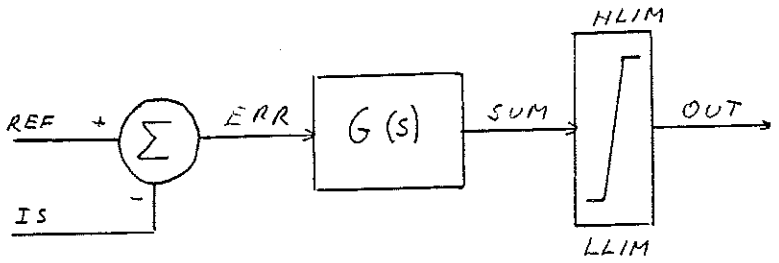

(Begränsningen gäller SUM, ej INT)

$G(s) = K\left(1 + \frac{1}{T_i \cdot s}\right)$ ; Enkel differentiering. $s = \frac{1}{T_s}(1 - z^{-1})$

$SUM = \underbrace{K\ ERR}_{PROP} + \underbrace{K \cdot \frac{T_s}{T_i} \cdot ERR}_{INT - INTOLD} + \underbrace{OUTOLD - K\ ERROLD}_{INTOLD}$

```
begin
ERR := REF - IS ;
PROP := K * ERR ;
INT := PROP * Ts/Ti + INTOLD ;
SUM := PROP + INT
if  SUM > HLIM  then
    begin
    SUM := HLIM ;
    end
if  SUM < LLIM  then
    begin
    SUM := LLIM ;
    end
OUT := SUM ;
INTOLD := INT ;
end
```

# PIPROG 1



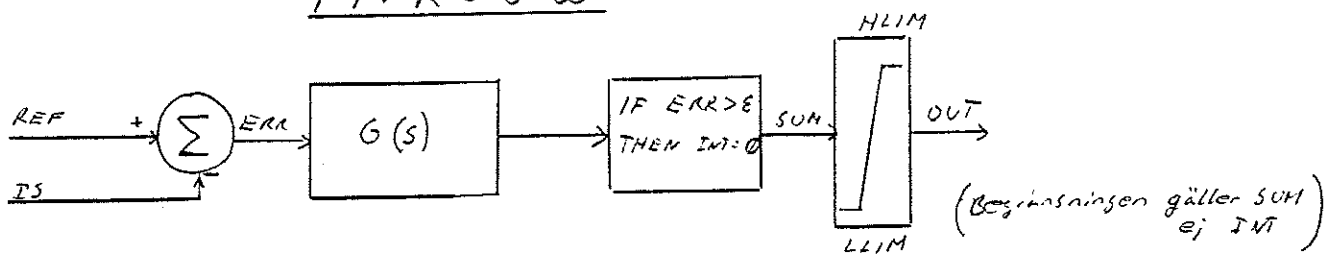

$G(s) = K\left(1 + \frac{1}{T_i \cdot s}\right)$ ; Enkel differentiering : $\frac{1}{T_s}(1 - z^{-1})$

$SUM = K\left[\left(1 + \frac{T_s}{T_i}\right) \cdot ERR - ERROLD\right] + OUTOLD =$

$= \underbrace{K \cdot ERR}_{PROP} + \underbrace{K \cdot \frac{T_s}{T_i} \cdot ERR}_{INT - INTOLD} + \underbrace{OUTOLD - K \cdot ERROLD}_{INTOLD}$

```
begin
ERR := REF - IS ;
PROP := K * ERR ;
INT := PROP * Ts/Ti + INTOLD ;
SUM := PROP + INT
if SUM > HLIM then
    begin
    SUM := HLIM ;
    INT := HLIM - PROP ;
    end
if SUM < LLIM then
    begin
    SUM := LLIM ;
    INT := LLIM - PROP ;
    end
OUT := SUM;
INTOLD := INT ;
end
```

# PIPROG 2



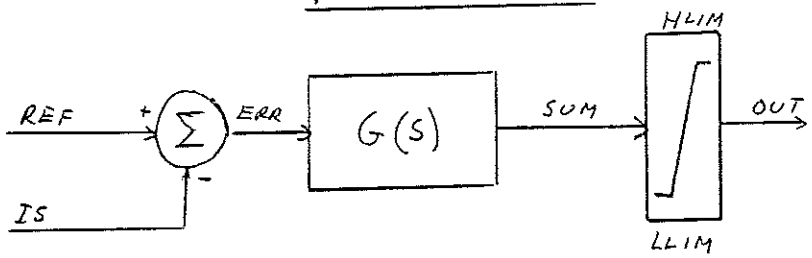

$G(S) = K\left(1 + \frac{1}{T_i \cdot S}\right)$ ; Enkel differentiering: $S = \frac{1}{T_s}\left(1 - z^{-1}\right)$

$$SUM = K\left[\left(1 + \frac{T_s}{T_i}\right) \cdot ERR - ERROLD\right] + OUTOLD =$$

$$= \underbrace{K \cdot ERR}_{PROP} + \underbrace{K \cdot \frac{T_s}{T_i} \cdot ERR}_{INT - INTOLD} + \underbrace{OUTOLD - K\ ERROLD}_{INTOLD}$$

```
begin
ERR:=REF-IS;
PROP:=K*ERR;
EPS:=ALFA*(1/(1+K))*REF;
INT:=PROP*Ts/Ti+INTOLD;
if abs(ERR) > abs(EPS) then
    begin
    INT:=0;
    end
SUM:=PROP+INT;
if SUM > HLIM then
    begin
    SUM:=HLIM;
    end
if SUM < LLIM then
    begin
    SUM:=LLIM
    end
OUT:=SUM;
INTOLD:=INT;
end
```

# PIPROG 3




$$G(s) = K\left[1 + \frac{1}{T_i \cdot s}\right] \quad ; \text{ Bilinjär transformation : } s = \frac{2}{T_s} \cdot \frac{1 - z^{-1}}{1 + z^{-1}}$$

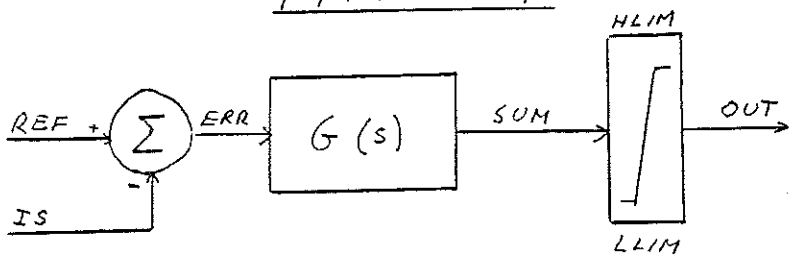
$$SUM = K\left[\left(1 + \tfrac{1}{2}\tfrac{T_s}{T_i}\right) \cdot ERR + \left(\tfrac{1}{2}\tfrac{T_s}{T_i} - 1\right) \cdot ERROLD\right] + OUTOLD =$$

$$= K\left[\tfrac{1}{2}\tfrac{T_s}{T_i}\left(ERR + ERROLD\right) + ERR - ERROLD\right] + OUTOLD =$$

$$= \underbrace{\tfrac{1}{2} \cdot \tfrac{T_s}{T_i}\left(PROP + PROPOLD\right)}_{INT - INTOLD} + \underbrace{K \cdot ERR}_{PROP} + \underbrace{OUTOLD - K \cdot ERROLD}_{INTOLD}$$

```
begin
ERR := REF - IS ;
PROP := K * ERR ;
INT := ((Ts/Ti)/2) * (PROP + PROPOLD)  + INTOLD ;
SUM := INT + PROP ;
if  SUM > HLIM  then
    begin
    SUM := HLIM ;
    INT := HLIM - PROP ;
    end
if  SUM < LLIM  then
    begin
    SUM := LLIM ;
    INT := LLIM - PROP ;
    end
OUT := SUM ;
INTOLD := INT ;
PROPOLD := PROP ;
end
```

# PIPROG 4




$G(s) = K\left[1 + \frac{1}{T_i s}\right]$; Utvidgad bilinjär transformation: $s = \frac{3}{T_s} \cdot \frac{1 - z^{-2}}{1 + 4z^{-1} + z^{-2}}$

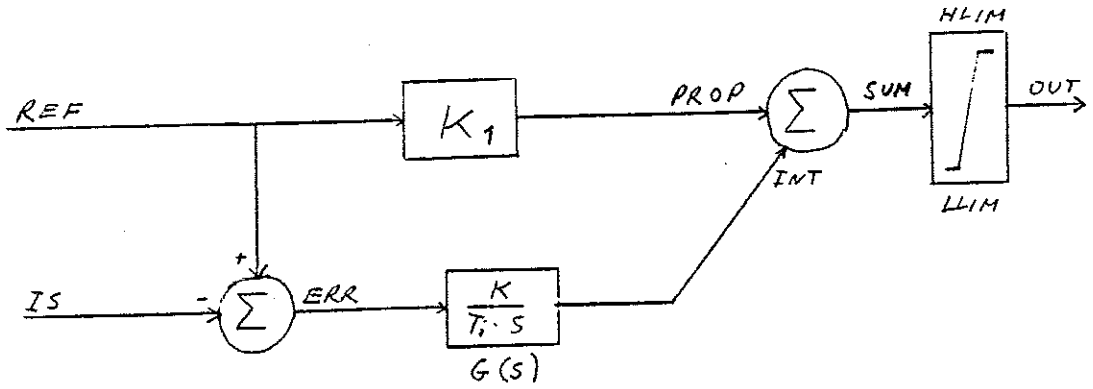
$$SUM = K\left[\left(1 + \frac{T_s}{3T_i}\right) \cdot ERRNY + \frac{4}{3}\frac{T_s}{T_i} \cdot ERR + \left(\frac{T_s}{3T_i} - 1\right) \cdot ERROLD\right] + OUTOLD =$$

$$= K\left[\frac{T_s}{3T_i}\left(ERRNY + 4\,ERR + ERROLD\right) + ERRNY - ERROLD\right] + OUTOLD =$$

$$= \underbrace{\frac{1}{3}\frac{T_s}{T_i}\left(PROPNY + 4 \cdot PROP + PROPOLD\right)}_{INTNY - INTOLD} + \underbrace{K \cdot ERRNY}_{PROP} + \underbrace{OUTOLD - K \cdot ERROLD}_{INTOLD}$$

```
begin
ERRNY:= REF - IS;
PROPNY:= K*ERRNY;
INTNY:= (((Ts/Ti)/3)*(PROPNY + (4*PROP) + PROPOLD)) - INTOLD;
SUM:= INTNY + PROPNY
if  SUM > HLIM  then
    begin
    SUM:= HLIM;
    INT:= HLIM - PROP;
    end
if  SUM < LLIM  then
    begin
    SUM:= LLIM;
    INT:= LLIM - PROP;
    end
OUT:= SUM;
INTOLD:= INT;
INT:= INTNY;
PROPOLD:= PROP;
PROP:= PROPNY;
end
```

# PIPROG 5




$G(s) = K \cdot \frac{1}{T_i \cdot s}$ ; Enkel differentiering : $s = \frac{1}{T_s}(1 - z^{-1})$

$$SUM = \underbrace{K_1 \cdot REF}_{PROP} + \underbrace{K \cdot \frac{T_s}{T_i} \cdot ERR + INTOLD}_{INT}$$

```
begin
ERR := REF - IS ;
PROP := K1 · REF ;
INT := K · Ts/Ti · ERR + INTOLD ;
SUM := PROP + INT ;
if SUM > HLIM then
    begin
    SUM := HLIM ;
    INT := HLIM - PROP ;
    end
if SUM < LLIM then
    begin
    SUM := LLIM
    INT := LLIM - PROP
    end
OUT := SUM
INTOLD := INT
end
```

# 5
# Test av de olika regulatorprogrammen

## 5.1
## Utvärdering av de olika PI-algoritmerna

Vid en referenssignal som är endast 2 % (= ± 0,100 V) av utsignalens begränsning (= ± 5,00 V) så märktes ingen skillnad på PIPROGØ och PIPROG1. Däremot om man ökade referenssignalen till 20 % (= ± 1,00 V), blev det en tydlig skillnad. Utan integratormättning (PIPROGØ) blev det betydligt kraftigare överslängar än med integratormättning (PIPROG1).

Integratormättning med villkorlig integration (PIPROG2) visade sig vara mindre lyckad vid en referenssignal på 2 % av utsignalens begränsning, ty integraldelen blev noll vid negativ referenssignal. Vid 20 % fungerade metoden tillfredsställande, men risken är uppenbar att felet inte blir tillräckligt litet så att integraldelen kopplas in.

Vid så låg referenssignal som 2 % av utsignalens begränsning räcker det med att en eller två bitar skall ändra sig för att vi skall få en betydande ändring av det statiska felet. Detta märktes speciellt på de två olika PI-regulatorerna med bilinjär transformation, men även de andra tre metoderna visade upp samma fenomen.

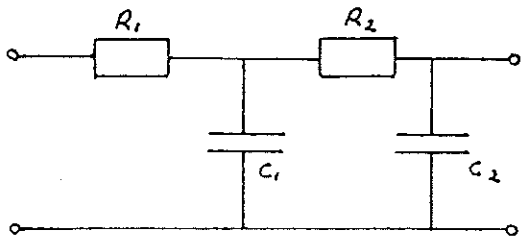
Vid test med 1:a ordningens system och med en referenssignal på 20 % av utsignalens begränsning syntes ingen märkbar skillnad på de tre olika metoderna, som alla har integratormättning, som härrör från utsignalens begränsning. Testade man däremot med ett 2:a ordningens system, så syntes en klar skillnad. Metoden med enkel differentiering (PIPROG1) fick den kortaste insvängningstiden, och metoden med s k utvidgad bilinjär transformation (PIPROG4) fick den klart längsta insvängningstiden.

Vid test med PIPROG5, dvs regulatorn som har P-del som endast arbetar med referenssignalen och I-del som endast arbetar med felet, så visade det sig att vi även här har en insvängningstid, som var betydligt större än PIPROG1. Jämförelsen blir här lite orättvis mot PIPROG5, eftersom här har regulatorn en annan struktur.

## 5.2
## Jämförelse mellan de olika PI-algoritmerna för ett 2:a ordningens system

Vidstående 2:a ordningens system användes vid testet.

Ett antal olika värden användes på R och C under testet.



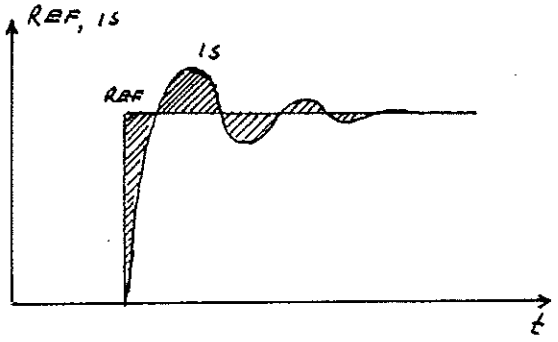
Sampeltider på mellan 0,5 ms till 15 ms användes. 0,5 ms var den lägsta tiden, eftersom exekveringstiden låg på 0,3-0,5 ms.

Nedanstående statiska fel beror på kvantiseringsfel. För PIPROG2 tillkommer risken att integraldelen aldrig träder till.

Några exempel på kurvor visas i 5.5.1 - 5.5.4.

| | PIPROGØ | PIPROG1 | PIPROG2 | PIPROG3 | PIPROG4 | PIPROG5 |
|---|---|---|---|---|---|---|
| Exekveringstid (inkl INSAML) | 0,32 ms | 0,33 ms | 0,38 ms | 0,43 ms | 0,46 ms | 0,37 ms |
| REF = 2 % av utsignalens begränsning | | | (Metoden fungerar här ej tillfredsställande) | | | |
| överslängar | stora | stora | små | stora | stora | stora |
| insvängningstid | lång | lång | kort | lång | mycket lång | mycket lång |
| statiskt fel | märkbart | märkbart | risk för mycket stort | betydande | betydande | märkbart |
| REF = 20 % av utsignalens begränsning | | | | | | |
| överslängar | stora | små | små | små | små | små |
| insvängningstid | kort | kort | kort | lång | mycket lång | mycket lång |
| statiskt fel | obetydligt | obetydligt | risk för mycket stort | obetydligt | obetydligt | obetydligt |

5.3
Kriterium för att avgöra vilken PI-algoritm som är bäst




Den algoritm som ger minimum av $\int_0^\infty |e(t)|dt$ är bäst.

Subroutinen MINFEL beräknar integralen av felet under 5 s efter att referenssignalen har stegats från negativt till positivt värde. Det beräknade värdet D/A-omvandlas och kan sen avläsas på en digitalvoltmeter.

5.4
Val av den bästa PI-algoritmen

Programmet med enkel differentiering och integralmättning, som beror på utsignalens begränsning, PIPROG1, visar genomgående de bästa resultaten. Mätningen som beräknar integralen av felet ger det minsta resultatet. Dessutom ger de övriga jämförelserna de bästa resultaten för PIPROG1. Slutsatsen blir att PIPROG1 är bäst.

## 5.5 Kurvor från de olika testen

### 5.5.1 Inverkan av integratormättming

I detta test visas inverkan av integratormättning. Samtliga regulatorprogram har enkel differentiering som överföring mellan Laplace- och z-transformen. Referenssignalen är 20% av utsignalens begränsning.

Vidstående 2:a ordningens system användes vid testet.

Sampeltiden Ts är 7ms

Förstärkningsfaktorn K är 1.5 ggr

Kvoten Ts/Ti är 0.875
dvs integrationstidskonstanten Ti är 8ms

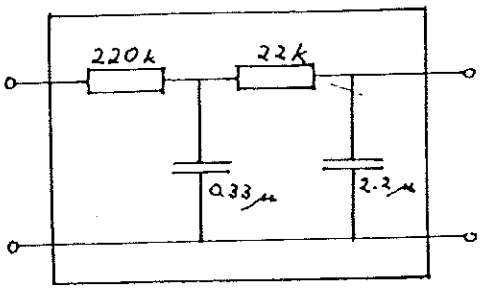
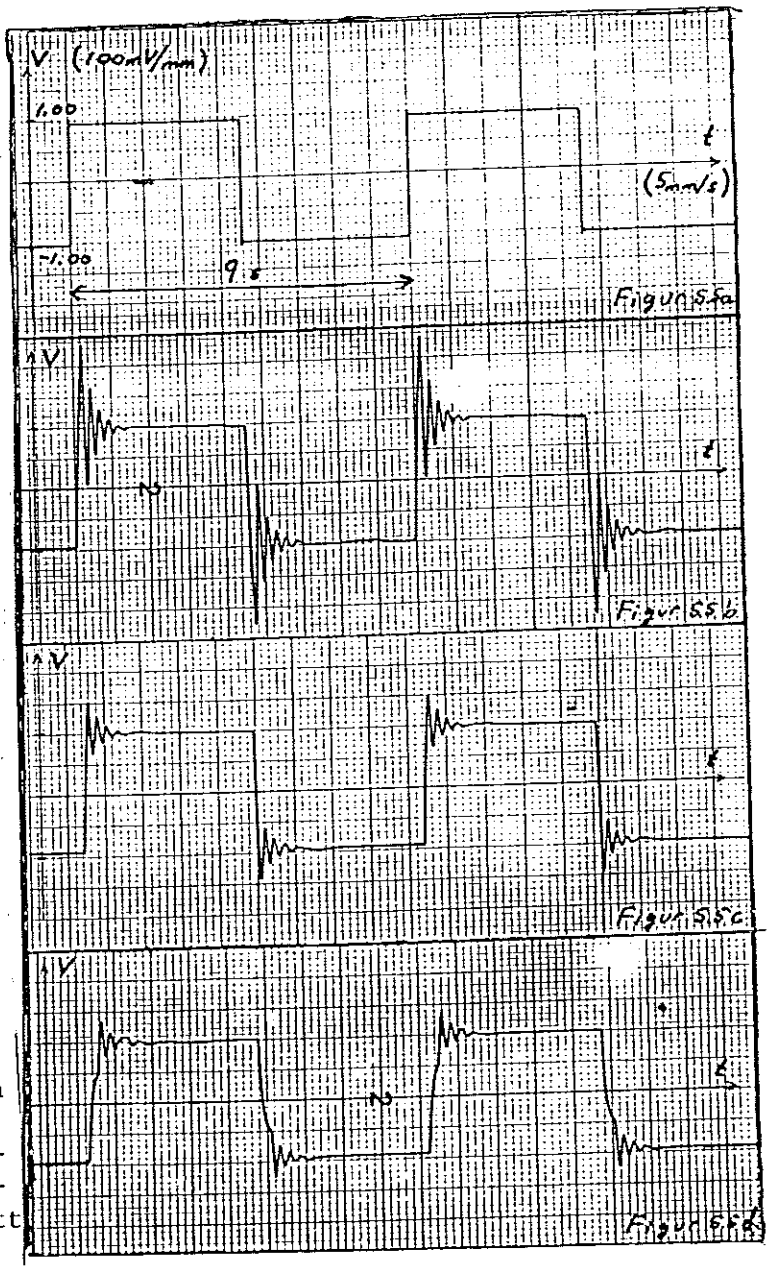



Fig. 5.5a Här är det REFERENS-SIGNALEN som ändrar sig från +1.00 V till -1.00 V

Fig. 5.5b PIPROG0 testas. Här ser man att utan integratormättning fås kraftiga överslängar.

Fig. 5.5c Med integratormättning som härrör från utsignalens begränsning som här i PIPROG1 fås en betydande dämpning i överslängarna.

Fig. 5.5d Detta test visar PIPROG2 dvs integratormättning med villkorlig integration. Visserligen fås något större dämpning än i förutgående test, men det är stor risk att integraldelen aldrig kopplas in, så att vi får ett betydande statiskt fel.



5.5.2 Jämförelse för olika överföringar mellan Laplace- och z-transformen

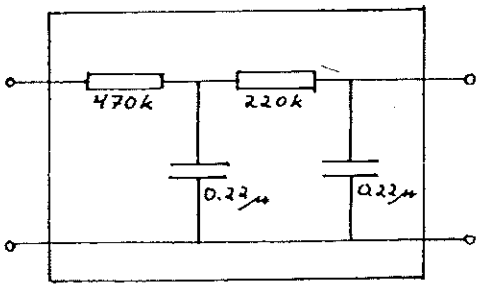
Här jämförs enkel differentiering, bilinjär transformation och utvidgad bilinjär transformation. Integratormättning som härrör från utsignalens begränsning används i alla tre fallen. Referenssignalen är 20% av utsignalens begränsning.

Vidstående 2:a ordningens system användes vid testet

Sampeltiden Ts är 10ms

Förstärkningsfaktorn K är 4 ggr

Kvoten Ts/Ti är 3/16
dvs integrationstidskonstanten Ti är 53ms



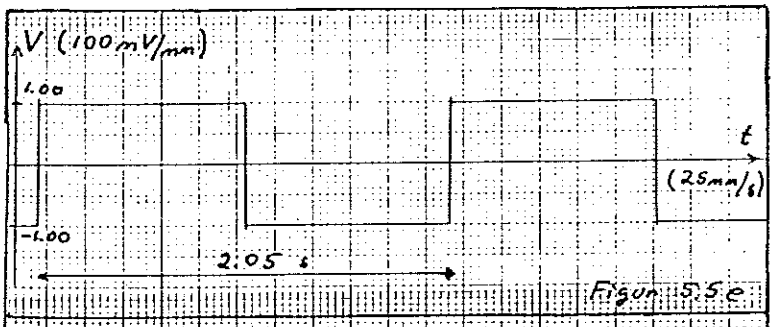




Fig. 5.5e REFERENSSIGNALEN ändrar sig från +1.00 V till -1.00 V

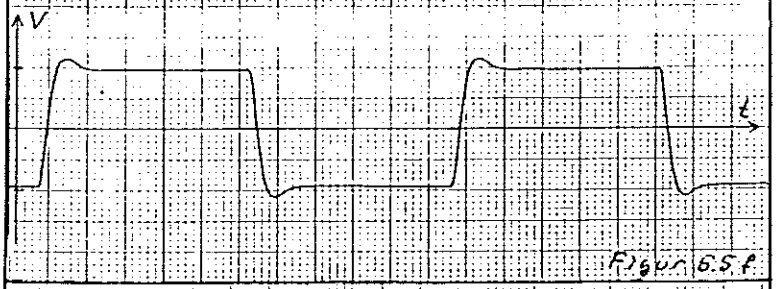



Fig. 5.5f Här visas insvängningen hos PIPROG1 dvs regulatorn med enkel differentiering.


Fig. 5.5g Vid bilinjär transformation, PIPROG3, så är det lite mer svängningar.

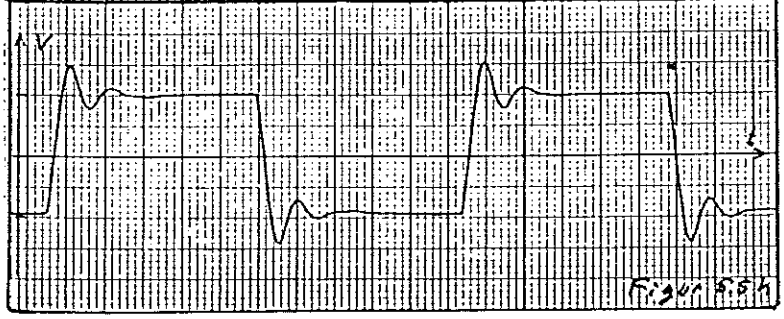



Fig. 5.5h PIPROG4 som är utvidgad bilinjär transformation ger störst antal svängningar.



6917 0900-BE (99) 80-02 3000 AO Ansvarig: POR

5.5.3 Jämförelse mellan PIPROG1 och PIPROG5

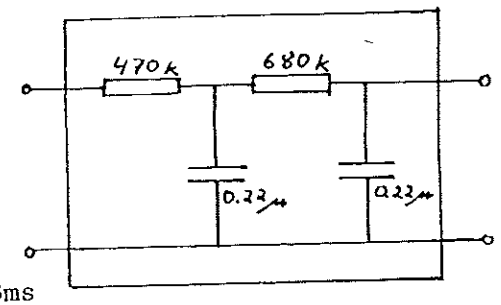
Vid detta test jämförs två regulatorer där den ena har P- och I-del som båda jobbar på felet (PIPROG1), och där den andra har en P-del som endast jobbar på referenssignalen och en I-del som endast jobbar på felet (PIPROG5). Båda regulatorerna har enkel differentiering som överföring mellan Laplace- och z-transformen. Referenssignalen är 20% av utsignalens begränsning.

Vidstående 2:a ordningens system användes vid testet.

Sampeltiden Ts är 14ms

Förstärkningsfaktorn K=K1 är 1 ggr

Kvoten Ts/Ti är 0.60
dvs integrationstidskonstanten Ti är 23ms



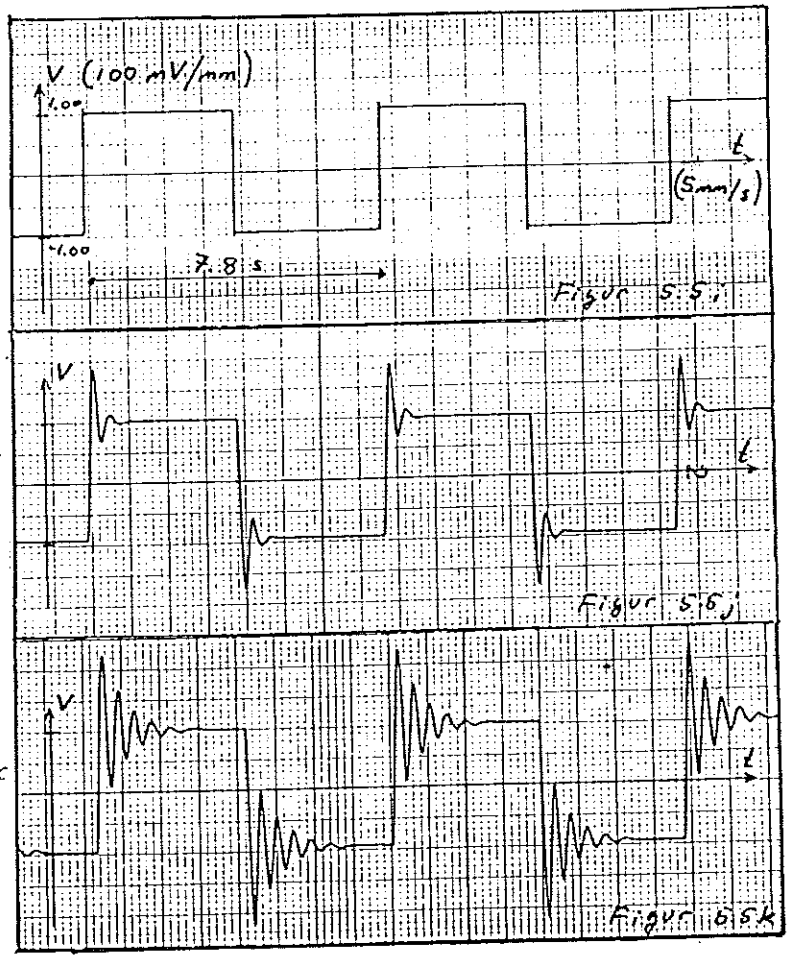




Fig. 5.5i REFERENSSIGNALEN ändrar sig från +1.00 V till -1.00 V

Fig. 5.5j Här är det signalen med PIPROG1 som regulator som visas.

Fig. 5.5k Här ses att PIPROG5 ger betydligt längre insvängningstid.



Ansvarig: POR

917 0900-BE (99) 80-02 3000 AO

5.5.4 Test med en referenssignal som är endast 2% av utsignalens begränsning

Vid denna test så varierar referenssignalen från +0.100 V till -0.100 V Alla övriga värden på konstanter och på systemet är exakt identiska med 5.5.1

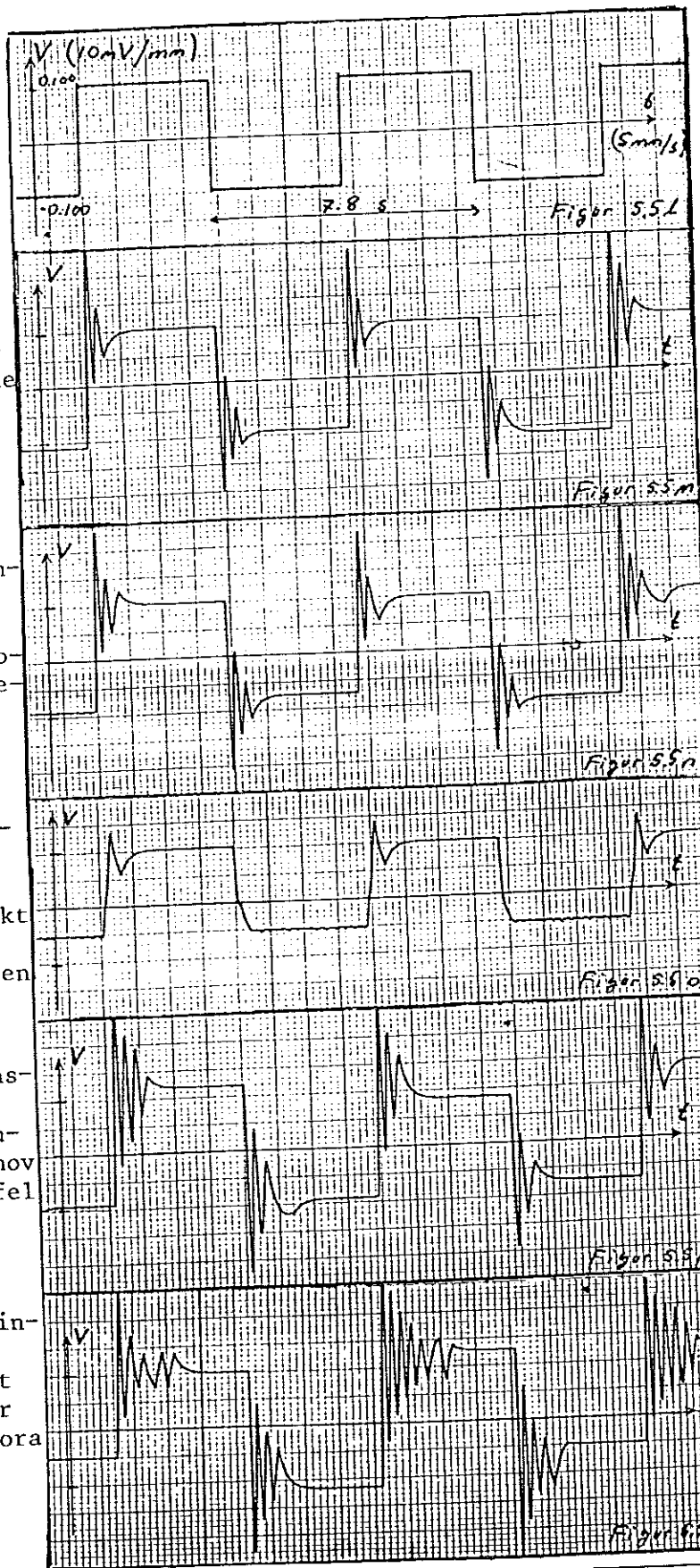



Fig. 5.51 REFERENSSIGNALEN ändrar sig från +0.100 V till -0.100 V

Fig. 5.5m PIPROG0 har även här stora överslängar. Märkbara statiska fel erhölls, beroende på kvantiseringsfel.

Fig. 5.5n PIPROG1 är här nästan identisk med ovanstående. Vi har inte fått någon dämpning av överslängarna beroende på att utsignalens begränsning på ± 5.00 V är för stor.

Fig. 5.5o PIPROG2 med villkorlig integration visar här sin uppenbara nackdel. Vi har här fått ett stort statiskt fel på negativa sidan beroende på att integraldelen aldrig trädde till.

Fig. 5.5p PIPROG3 med bilinjär transformation visar sig även den vara känslig för kvantiseringsfel som ger upphov till betydande statiska fel

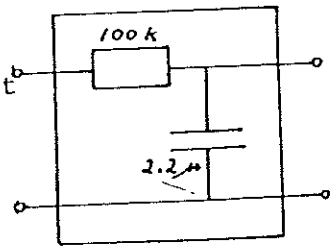
Fig. 5.5q PIPROG4 med utvidgad bilinjär transformation visar sig vara allra känsligast för kvantiseringsfel. Här uppstår ibland mycket stora statiska fel.



17 0900-BE (99) 80-02 3000 AO Ansvarig: POR

5.5.5 Test med ett 1:a ordningens system

Här jämförs alla PI-regulatorerna samtidigt. Referenssignalen ändrar sig från +1.00 V till -1.00 V




Vidstående 1:a ordningens system användes vid testet

Sampeltiden Ts är 7ms

Förstärkningsfaktorn K=K1 är 1.5 ggr

Kvoten Ts/Ti är 0.875
dvs integrationstidskonstanten Ti är 8ms

REFERENSSIGNALEN

PIPROG0

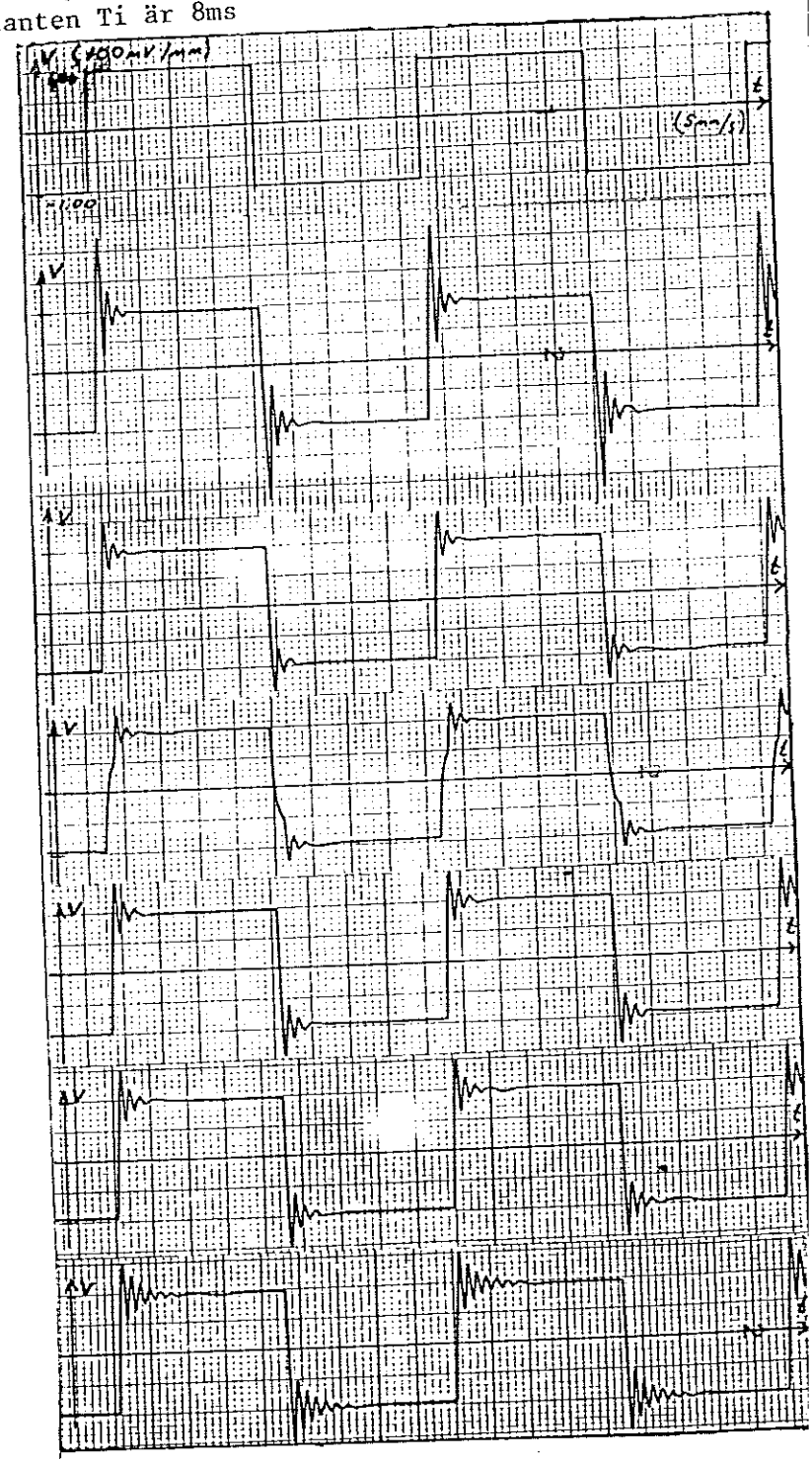
PIPROG1

PIPROG2

PIPROG3

PIPROG4

PIPROG5

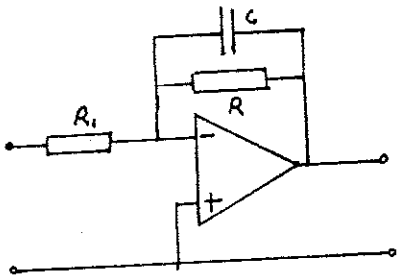


# 6
# Test av den bästa PI-algoritmen

## 6.1
## Teori för test vid varierande förhållande mellan sampelfrekvens och systemets överkorsningsfrekvens

1:a ordn. system:
(LP-filter med 10 ggr förstärkning)

$R = 10 \cdot R_1$




$$G(s) = -\frac{10}{1+sRC}$$

$$\log|G(i\omega)| = \log 10 \cdot \log\sqrt{1+(\omega RC)^2} = 1 - \log\sqrt{1+(\omega RC)^2}$$

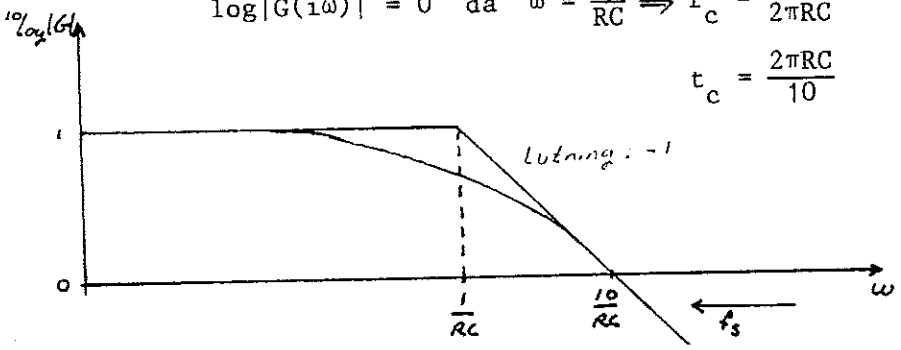
Bodediagram: lågpassasymptoten ($\omega RC \ll 1$) : $\log|G(i\omega)| = 1$

högpassasymptoten ($\omega RC \gg 1$) : $\log|G(i\omega)| = 1 - \log \omega RC$

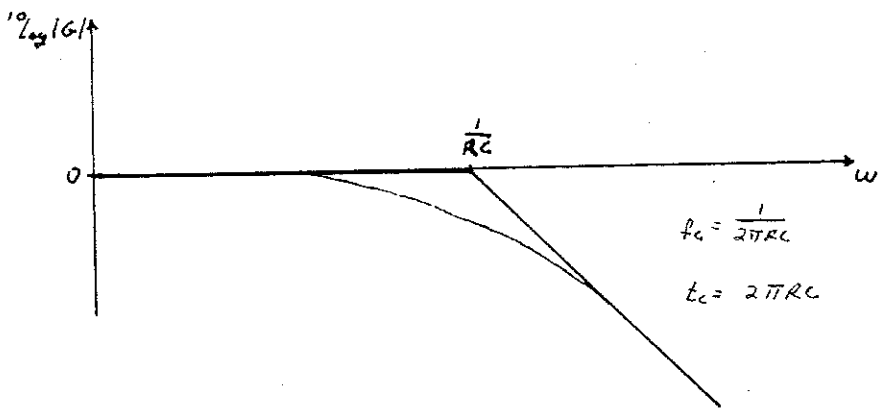
brytpunkt då $\omega = \frac{1}{RC}$

$\log|G(i\omega)| = 0$ då $\omega = \frac{10}{RC} \Rightarrow f_c = \frac{10}{2\pi RC}$

$$t_c = \frac{2\pi RC}{10}$$



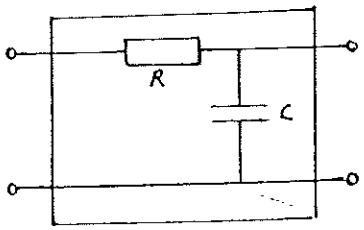

Testet går ut på att se hur nära vi kan gå med sampelfrekvensen $f_s$ mot $f_c$.



## 6.2 Test av PIPROG1 för olika sampeltider

### 6.2.1 Test med 1:a ordningens system (ett LP-filter)

Vidstående system användes vid testet.




Förstärkningsfaktorn K=1 ggr

Kvoten Ts/Ti=0.875

R=47kohm

C=0.1$\mu$F

$t_c = 2*\pi*R*C = 29.5$ms

Instabilitet när Ts=10ms, dvs då $t_c$ /Ts=2.95 ggr

REF

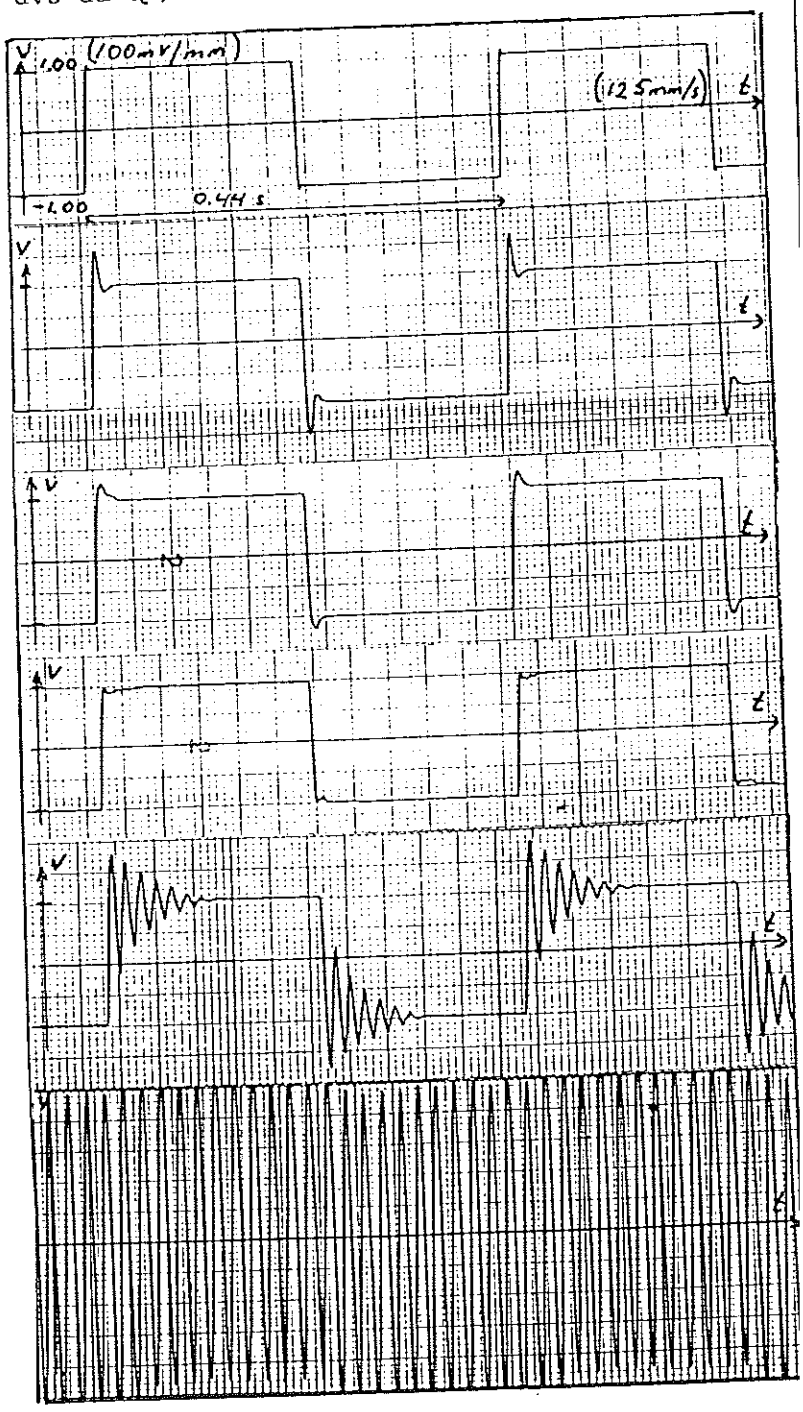
Ts=1ms

Ts=2ms

Ts=4ms

Ts=8ms

Ts=10ms



6.2.2 Test med LP-filter och 10 ggr förstärkning

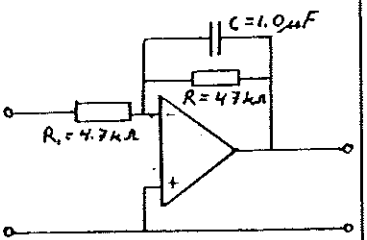
Vidstående system användes vid testet.

Förstärkningsfaktorn K=0.5 ggr

Kvoten Ts/Ti=0.5

$t_c = (2\pi RC)/10 = 29.5ms$

Instabilitet när Ts=15.5ms, dvs när $t_c/Ts = 1.9$ ggr




REF

Ts=1ms

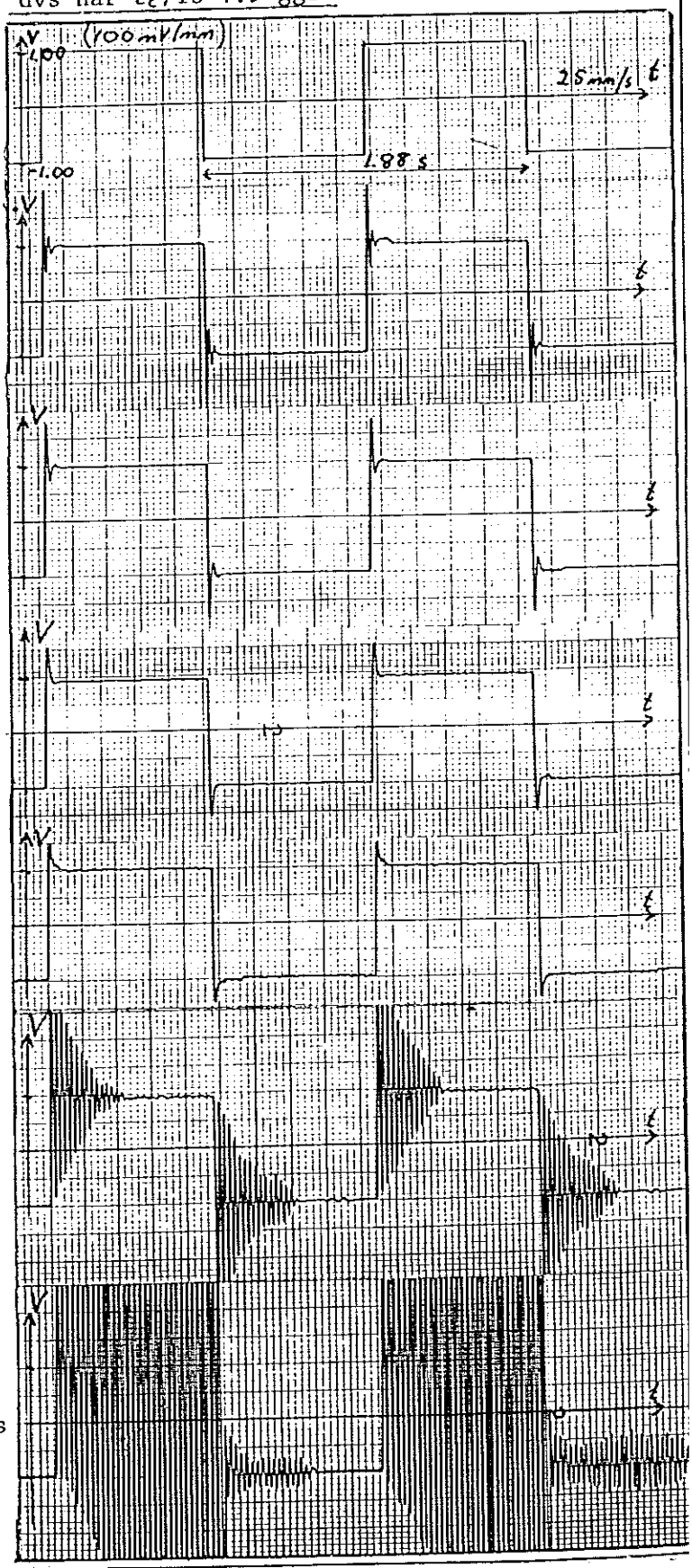
Ts=2ms

Ts=4ms

Ts=8ms

Ts=15ms

Ts=15.5ms



## 7 Slutlig kommentar

Det visade sig vara lämpligt att i regulatorprogrammet särskilja på P-del och I-del. Om man i stället använde sig av den gamla utsignalen och de gamla felen, så erhölls ej en stabil signal, utan den svängde mer eller mindre regelbundet. Orsaken till detta var troligen att genom att ta skillnaden mellan gamla och nya felet, så erhöll man ett litet värde, som är starkt beroende av kvantiseringsfel. Genom att sedan multiplicera detta värde med förstärkningsfaktorn K påverkar detta kvantiseringsfel utsignalen tillräckligt mycket för att det skall märkas.

6917 0907-AB (999) 81.12 100000 Ansvarig: CKRB

```
PAGE  001  INIT     .SA:1  INIT   ♦♦♦♦♦ INITIATION ROUTINE ♦♦♦♦♦

00010 00001                          OPT    Z01,REL,LLE=80,P=62
00020 00002                          NAM    INIT
00030 00003                          TTL    ♦♦♦♦♦ INITIATION ROUTINE ♦♦♦♦♦
00040 00004                    ♦
00050 00005                    ♦
00060 00006                    ♦
00070 00007                    ♦
00080 00008                    ♦♦♦♦♦♦♦♦ PROGRAM DESCRIPTION ♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦
00090 00009                    ♦                                                    ♦
00100 00010                    ♦ THIS ROUTINE INITIATES ALL GLOBAL PARAMETERS       ♦
00110 00011                    ♦ HARDWARE-SECTORS AND MACRODEFINITIONS.             ♦
00120 00012                    ♦                                                    ♦
00130 00013                    ♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦
00140 00014                    ♦
00150 00015                    ♦
00160 00016                    ♦
00170 00017                    ♦
00180 00018                    ♦♦♦♦♦♦♦♦ DATABASE DEFINITION ♦♦♦♦♦♦♦♦
00190 00019                    ♦
00200 00020D 0000                    DSCT
00210 00021                    ♦
00220 00022                    ♦
00230 00023                          XDEF   ADIN1,ADOUT,DA1,TSAMP,INDAT
00240 00024                          XDEF   DIP2,DIP3,K,TS.TI
00250 00025                          XDEF   REF,IS,DA2
00260 00026                          XDEF   TCSR,FRC,OCR,RESET,SAMPT
00270 00027
00280 00028                          XREF   REST,RTC,INTER
00290 00029
00300 00030                    ♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦
00310 00031                    ♦
00320 00032                    ♦
00330 00033                    ♦
00340 00034                    ♦
00350 00035                    ♦
00360 00036                    ♦♦♦♦♦♦♦♦♦ MACRO DEFINITIONS ♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦
00370 00037                    ♦
00380 00038                    ♦
00390 00039                    LIMIT  MACR
      00040                    00400          BVC      ö0
      00041                    00410          BGT      ö.0
      00042                    00420          LDD      #$8000
      00043                    00430          BRA      ö0
      00044                    00440 ö.0      LDD      #$7FFF
      00045                    00450          ENDM
00460 00046                    ♦
00470 00047                    ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦ ♦
00480 00048                    ♦
00490 00049                    SIGNCH MACR                 CHANGE SIGN
      00050                    00500          BMI     ö.1
      00051                    00510          COMA
      00052                    00520          COMB
      00053                    00530          ADDD    #$0001
      00054                    00540          BRA     ö0
      00055                    00550 ö.1      COMA
      00056                    00560          COMB
      00057                    00570          SUBD    #$0001
      00058                    00580          ENDM
00590 00059                    ♦
00600 00060                    ♦♦♦♦♦♦♦♦ HARDWARE-ADDRESS DEFINITION ♦♦♦♦♦♦♦♦♦♦♦♦
00610 00061                    ♦
00620 00062            2000  A ADIN1  EQU    $2000    A/D START CHANNEL 1
```

```
PAGE  002  INIT     .SA:1  INIT   ♦♦♦♦♦ INITIATION ROUTINE ♦♦♦♦♦

00630 00063           2001  A ADIN2  EQU     $2001     ------ " -------- 2
00640 00064                   ♦
00650 00065           200C  A ADOUT  EQU     $200C     DIGITAL RESULT FROM A/D
00660 00066                   ♦
00670 00067           2008  A DA1    EQU     $2008     D/A -LOADADDRESS 1
00680 00068           200A  A DA2    EQU     $200A     - " -         2
00690 00069           3008  A RESET  EQU     $3008     RESET IRQ1-N
00700 00070           300C  A DIP1   EQU     $300C     CONSTANT NO.    1   (DIP-SWITC
00710 00071           300D  A DIP2   EQU     $300D     -------- " --- 2   ---- " ---
00720 00072           300E  A DIP3   EQU     $300E     -------- " --- 3   ---- " ---
00730 00073                   ♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦
00740 00074D 0000     0001  A K      RMB     1
00750 00075D 0001     0001  A TS.TI  RMB     1
00760 00076D 0002     0002  A SLASK  RMB     2
00770 00077D 0004     0004  A INDAT  RMB     4
00780 00078                   ♦
00790 00079           300C  A TSAMP  EQU     DIP1
00800 00080           0004  D REF    EQU     INDAT
00810 00081           0006  D IS     EQU     INDAT+2
00820 00082                   ♦
00830 00083                   ♦
00840 00084                   ♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦
00850 00085                   ♦
00860 00086                   ♦♦♦♦♦♦♦♦♦♦ INITIATION ROUTINE FOR REALTIME CLOCK ♦♦
00870 00087                   ♦
00880 00088                   ♦INTERRUPT VECTOR DEFINITION
00890 00089                   ♦
00900 00090D 0008     FCE1  A INTVCT FDB     $FCE1     SERIAL COM INT    (MEX)
00910 00091D 000A     FCE1  A TIMOWF FDB     $FCE1     TIMER OWERFLOW    (USER)
00920 00092D 000C     0000  A TIMOUT FDB     RTC       ADDRESS REALTIME CLOCK
00930 00093D 000E     FCE1  A TIMIN  FDB     $FCE1     INPUT COMPARE  (USER)
00940 00094D 0010     0000  A IRQ1   FDB     INTER     ALARM FROM PROCESS  (USER)
00950 00095D 0012     F821  A SWI    FDB     $F821     SW1  (MEX)
00960 00096D 0014     F803  A NMI    FDB     $F803     NMI
00970 00097                   ♦
00980 00098           0008  A TCSR   EQU     $08       INTERNAL COUNTER CONTROL WOR
00990 00099           0009  A FRC    EQU     $09       FREE RUNNING COUNTER ADRESS
01000 00100           000B  A OCR    EQU     $0B       OUTPUT COMPARE REG
01010 00101                   ♦
01020 00102D 0016     0002  A SAMPT  RMB     2         SAMPLETIME FOR TIMER
01030 00103                   ♦
01040 00104                   ♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦♦
01050 00105                   ♦
```

```
PAGE  003  INIT     .SA:1  INIT    ***** INITIATION ROUTINE *****

01070 00107                        ************ PROGRAM SECTION **********************
01080 00108                        *
01090 00109P 0000                         PSCT
01100 00110                        *
01110 00111P 0000 0F               START  SEI              DISABLE INTERRUPT
01120 00112                        *
01130 00113                        *SET INTERRUPT VECTOR
01140 00114                        *
01150 00115P 0001 CC 0008  D              LDD    #INTVCT  PTR TO D-REG
01160 00116P 0004 DD FE    A              STD    $FE      STORE PTR TO MEX PTR
01170 00117                        *
01180 00118                        *SET STATUS TO ENABLE IRQ2-N TO INT TIMER
01190 00119                        *
01200 00120P 0006 86 08    A              LDAA   #$08     STATUS TO A-REG
01210 00121P 0008 97 08    A              STAA   TCSR
01220 00122                        *
01230 00123                        *JUMP TO RESTART ROUTINE
01240 00124                        *
01250 00125P 000A 7E 0000  A              JMP    REST
01260 00126                        *
01270 00127                        ************************************************************
01280 00128                        *
01290 00129                               END
TOTAL ERRORS 00000--00000
```

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| ADIN1 | 2000 | ADIN2 | 2001 | ADOUT | 200C | DA1 | 2008 | DA2 | 200A | DIP1 | 300C |
| DIP2 | 300D | DIP3 | 300E | FRC | 0009 | INDAT | 0004 | INTER | 000D | INTVCT | 0008 |
| IRQ1 | 0010 | IS | 0006 | K | 0000 | NMI | 0014 | OCR | 000B | REF | 0004 |
| RESET | 3008 | REST | 0013 | RTC | 0014 | SAMPT | 0016 | SLASK | 0002 | START | 0000 |
| SWI | 0012 | TCSR | 0008 | TIMIN | 000E | TIMOUT | 000C | TIMOWF | 000A | TS.TI | 0001 |
| TSAMP | 300C | | | | | | | | | | |

```
00010 00001                             OPT    Z01,REL,LLE=80,P=62
00020 00002                             NAM    RESTART
00030 00003                             TTL    ******* RESTART ROUTINE ********
00040 00004                     *
00050 00005                     *
00060 00006                     *
00070 00007                     *
00080 00008                     ************** PROGRAM DESCRIPTION ****************
00090 00009                     *
00100 00010                     * THE RESTART ROUTINE CLEARES NECESSARY VARIABLES
00110 00011                     * AND CONVERTS SAMPLETIME TO RTC CLOCK PULSES
00120 00012                     * IT ALSO INITIATES REALTIME CLOCK FOR NEXT CLOCK
00130 00013                     *
00140 00014                     ***********************************************************
00150 00015                     *
00160 00016                     *
 170 00017                      *
 180 00018                      *
 190 00019                      *
00200 00020                     *
00210 00021D 0000                       DSCT
 220 00022                      *
 230 00023                              XREF   DA1,KONST,INT,ERR
 240 00024                              XREF   TSAMP,SAMPT
00250 00025                             XREF   TCSR,FRC,OCR
00260 00026                     *
00270 00027                             XDEF   REST
00280 00028                     *
00290 00029                     ***********************************************************
```

```
00310 00031                          *
00320 00032                          *
00330 00033                          ********** PROGRAM SECTION **********
00340 00034                          *
00350 00035P 0000                           PSCT
00360 00036                          *
00370 00037                          *LOAD REG A,B WITH 0000
00380 00038P 0000 CC 0000  A REST    LDD    #0000     D-REG:=0
00390 00039                          *
00400 00040                          *PUT OUT ZERO ON D/A
00410 00041                          *
00420 00042P 0003 FD 0000  A         STD    DA1       CLEAR D/A 1
00430 00043                          *
00440 00044                          *CLEAR MEMORY CELLS
00450 00045                          *
00460 00046P 0006 FD 0000  A         STD    INT       INT:=D-REG
00470 00047P 0009 FD 0000  A         STD    ERR       ERR:=D-REG
00480 00048                          *
00490 00049                          *
00500 00050                          *************** RESTART ROUTINE **********
00510 00051P 000C BD 0000  A         JSR    KONST     JUMP TO SUBROUTINE KONST
00520 00052                          *
00530 00053                          *INITIATE NEXT CLOCK INTERRUPT
00540 00054                          *
00550 00055P 000F B6 0000  A         LDAA   TCSR      READ TCSR TO RESET OCF
00560 00056P 0012 FC 0000  A         LDD    FRC       READ FRC
00570 00057P 0015 F3 0000  A         ADDD   SAMPT     ADD SAMPT TO FRC VALUE
00580 00058P 0018 FD 0000  A         STD    OCR       STORE IN OCR
00590 00059                          *
00600 00060P 001B 0E                 CLI              ENABLE INTERRUPT
00610 00061                          *
00620 00062                          *WAIT FOR TIMER INTERRUPT
00630 00063                          *
00640 00064P 001C 01         WAIT    NOP              WAIT LOOP
00650 00065P 001D 7E 001C  P         JMP    WAIT      JUMP TO WAIT
00660 00066                          *
00670 00067                          END
TOTAL ERRORS 00000--00000
```

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| DA1 | 0005 | ERR | 0006 | FRC | 0007 | INT | 0008 | KONST | 0009 | OCR | 000A |
| REST | 0000 | SAMPT | 000C | TCSR | 000D | TSAMP | 000E | WAIT | 001C | | |

```
PAGE  001  KONST    .SA:1  KONST  ********* KONSTANTBERAEKNINGAR **********

00010 00001                            OPT     Z01,REL,LLE=80,P=62
00020 00002                            NAM     KONST
00030 00003                            TTL     ********* KONSTANTBERAEKNINGAR *****
00040 00004                    *
00050 00005                    *
00060 00006                    *
00070 00007                    *
00080 00008                    ********** PROGRAM DESCRIPTION ********************
00090 00009                    *
00100 00010                    * THIS PROGRAM READS THE 3 DIP-SWITCHES WHEN RESTAI
00110 00011                    * THE PROGRAM AND AT IRQ1-N INTERRUPT, THEN CONVER
00120 00012                    * TSAMP TO SAMPT
00130 00013                    *
00140 00014                    ************************************************
00150 00015                    *
00160 00016                    *
00170 00017                    *
00180 00018                    *********** DATABASE DEFINITION ******************
00190 00019                    *
00200 00020D 0000                      DSCT
00210 00021                    *
00220 00022                            XREF    TSAMP,DIP2,DIP3,K
00230 00023                            XREF    RESET,SAMPT,TS.TI
00240 00024                    *
00250 00025                            XDEF    KONST,ERR,PROP,PROPOL,INT,INTOLD
00260 00026                            XDEF    K1
00270 00027                    *
00280 00028                    *
00290 00029                    ************************************************
00291 00030                    *
00292 00031            300F  A DIP4    EQU     $300F
00300 00032                    *
00310 00033D 0000      0001  A K1      RMB     1
00320 00034D 0001      0002  A ERR     RMB     2
00330 00035D 0003      0002  A PROP    RMB     2
00340 00036D 0005      0002  A PROPOL  RMB     2
00350 00037D 0007      0002  A INT     RMB     2
00360 00038D 0009      0002  A INTOLD  RMB     2
00370 00039                    *
00380 00040                    *********** PROGRAM SECTION ******************
00390 00041                    *
00400 00042P 0000                      PSCT
00410 00043                    *
00420 00044                    *DISABLE INTERRUPT
00430 00045                    *
00440 00046P 0000 0F           KONST   SEI                     DISABLE INTERRUPT
00450 00047                    *
00460 00048                    * READ KONSTANTS FROM DIP-SWITCHES
00470 00049                    *
00480 00050P 0001 B6 0000  A           LDAA    DIP2            READ DIP2
00490 00051P 0004 B7 0000  A           STAA    K               STORE IN K
00500 00052                    *
00510 00053P 0007 B6 0000  A           LDAA    DIP3            READ DIP3
00520 00054P 000A B7 0000  A           STAA    TS.TI           STORE IN TS.TI
00530 00055                    *
00540 00056P 000D B6 300F  A           LDAA    DIP4            READ DIP4
00550 00057P 0010 B7 0000  D           STAA    K1              STORE IN K1
00560 00058                    *
00570 00059                    *CLEAR MEMORY CELL FOR INT,ERR AND OUT
00580 00060P 0013 CC 0000  A           LDD     #$0000          D-REG:=0000
00590 00061P 0016 FD 0001  D           STD     ERR             ERR:= 0000
00600 00062P 0019 FD 0003  D           STD     PROP            PROP:= 0000
```

```
PAGE  002  KONST    .SA:1  KONST  ********* KONSTANTBERAEKNINGAR **********

00610 00063P 001C FD 0005  D          STD    PROPOL   PROPOL:= 0000
00620 00064P 001F FD 0007  D          STD    INT      INT:= 0000
00630 00065P 0022 FD 0009  D          STD    INTOLD   INTOLD:= 0000
00640 00066                        *
00650 00067                        ***********************************************
00660 00068                        * CONVERT TSAMP TO SAMPT
00670 00069                        *
00680 00070P 0025 B6 0000  A          LDAA   TSAMP    A-REG:=TSAMP
00690 00071P 0028 C6 4B    A          LDAB   #75      B-REG:=75
00700 00072P 002A 3D                  MUL
00710 00073P 002B FD 0000  A          STD    SAMPT    SAMPT:=D-REG
00720 00074                        *
00730 00075                        *
00740 00076P 002E B6 0000  A          LDAA   RESET    RESET IRQ1-N
00750 00077                        *
00760 00078P 0031 39                  RTS
00770 00079                        *
0  30 00080                           END
TOTAL ERRORS 00000--00000
```

```
D  2   0005  DIP3   0006  DIP4   300F  ERR    0001  INT    0007  INTOLD 0009
K      000A  K1     0000  KONST  0000  PROP   0003  PROPOL 0005  RESET  000F
SAMPT  0010  TS.TI  0011  TSAMP  0012
```

```
PAGE  001  RTC      .SA:1  RTC     ********** REAL TIME CLOCK ROUITINE *********

00010 00001                            OPT     Z01,REL,LLE=80,P=62
00020 00002                            NAM     RTC
00030 00003                            TTL     ********** REAL TIME CLOCK ROUITINE *
00040 00004                       *
00050 00005                       *
00060 00006                       *
00070 00007                       *
00080 00008                       ********** PROGRAM DESCRIPTION **********
00090 00009                       *
00100 00010                       * THIS MODULE CONTENTS TWO DIFFERENT INTERRUPT-
00110 00011                       * ROUTINES:
00120 00012                       *
00130 00013                       * RTC - AN INTERRUPTROUTINE THAT INITIATES NEXT
00140 00014                       * CLOCK INTERRUPT AND JUMPS TO REGULATORPROGRAM
00150 00015                       *
00160 00016                       * INTER - AN INTERRUPTROUTINE THAT TAKES CARE OF
00170 00017                       * IRQ1-N -INTERRUPTS AND THEN CALLS THE SUBROUTINE
00180 00018                       * KONST, FOR NEW CONSTANTS.
00190 00019                       *
00200 00020                       ************************************************
00210 00021                       *
00220 00022                       *
00230 00023                       *
00240 00024                       *
00250 00025                       ********** DATABASE DEFINITION **********
00260 00026                       *
00270 00027                             DSCT
00280 00028D 0000                 *
00290 00029                             XDEF    RTC,INTER
00300 00030                       *
00310 00031                             XREF    TCSR,OCR,SAMPT,KONST,PIREG
00320 00032                       *
00330 00033                       *
00340 00034                       *
00350 00035                       ************************************************
00360 00036                       *
00370 00037                       *
00380 00038                       ********** PROGRAM SECTION **********
00390 00039                       *
00400 00040                             PSCT
00410 00041P 0000                 *
00420 00042                       ************************************************
00430 00043                       *
00440 00044                       * INITIATE NEXT CLOCK INTERRUPT
00450 00045                       * AND START REGULATOR
00460 00046                       *
00470 00047                       RTC     SEI
00480 00048P 0000 0F              A       LDAA    TCSR        READ TCSR TO RESET OCF
00490 00049P 0001 B6 0000         A       LDD     OCR         READ OCR
00500 00050P 0004 FC 0000         A       ADDD    SAMPT       ADD SAMPT TO FRC VALUE
00510 00051P 0007 F3 0000         A       STD     OCR         STORE IN OCR
00520 00052P 000A FD 0000         A       *
00530 00053                       *START REGULATOR PROGRAM AS SUBROUTINE
00540 00054                       *
00550 00055                             JMP     PIREG       JUMP TO REGULATORPROGRAM
00560 00056P 000D 7E 0000         A     *
00570 00057                       ************************************************
00580 00058                       *
00590 00059                       * IRQ1-N INTERRUPT ROUTINE
00600 00060                       *
00610 00061                       *
00620 00062
```

```
00630 00063P 0010 0F          INTER  SEI              DISABLE INTERRUPT
00640 00064P 0011 BD 0000  A         JSR    KONST     JUMP TO SUBROUTINE KONST
00650 00065P 0014 0E                 CLI              ENABLE INTERRUPT
00660 00066P 0015 3B                 RTI              RETURN FROM INTERRUPT
00670 00067                  *
00680 00068                  ********************************************************
00690 00069                          END
TOTAL ERRORS 00000--00000


INTER  0010  KONST  0006  OCR    0007  PIREG  0008  RTC    0000  SAMPT  000A
TCSR   000B
```

```
PAGE  001  INSAML  .SA:1  INSAML ********** INSAMLINGSRUTIN **********

00010 00001                         OPT    Z01,REL,LLE=80,P=62
00020 00002                         NAM    INSAML
00030 00003                         TTL    ********** INSAMLINGSRUTIN ********
00040 00004                 *
00050 00005                 *
00060 00006                 *
00070 00007                 ******** PROGRAM DESCRIPTION ***********************
00080 00008                 *
00090 00009                 * THIS PROGRAM READS THE TWO ANALOG INPUTS
00100 00010                 * AND STORES THE DIGITAL VALUES IN MEMORY
00110 00011                 *
00120 00012                 ************************************************
00130 00013                 *
00140 00014                 *
00150 00015                 ************ DATABASE DEFINITION ******************
00160 00016                 *
( )170 00017D 0000                  DSCT
  )80 00018                 *
00190 00019                         XREF   ADIN1,ADOUT,INDAT
00200 00020                 *
00210 00021                         XDEF   INSAML
( )220 00022                *
  )230 00023                *
00240 00024D 0000     0002  A POINT1 RMB    2          POINTER TO ANALOG INPUTS
00250 00025D 0002     0002  A POINT2 RMB    2          POINTER TO MEMORY CELLS
00260 00026D 0004     0001  A CNT    RMB    1          LOOP COUNTER
00270 00027                 *
00280 00028                 *
00290 00029                 ************************************************
```

```
PAGE  002  INSAML  .SA:1  INSAML ********** INSAMLINGSRUTIN ****************

00310 00031                          ********** PROGRAM SECTION **********
00320 00032                          *
00330 00033P 0000                            PSCT
00340 00034                          *
00350 00035P 0000 CC 0000  A INSAML  LDD     #ADIN1
00360 00036P 0003 FD 0000  D         STD     POINT1    POINT1:=2000
00370 00037                          *
00380 00038P 0006 CC FFFE  A         LDD     #INDAT-2
00390 00039P 0009 FD 0002  D         STD     POINT2
00400 00040                          *
00410 00041P 000C 86 02    A         LDAA    #2
00420 00042P 000E B7 0004  D         STAA    CNT       CNT:=2
00430 00043                          *
00440 00044                          ********
00450 00045P 0011 FE 0000  D NEXTCH  LDX     POINT1    X:=POINT1
00460 00046P 0014 FC 0000  A         LDD     ADOUT     D:=A/D
  470 00047P 0017 A7 00    A         STAA    0,X       START A/D, CHANNEL X
  480 00048                          *
00490 00049P 0019 08                 INX               X:=X+1
00500 00050P 001A FF 0000  D         STX     POINT1    POINT1:=X
00510 00051                          *
  520 00052P 001D FE 0002  D         LDX     POINT2    X:=POINT2
  530 00053P 0020 ED 00    A         STD     0,X       INDAT(X):=D-REG
00540 00054                          *
00550 00055P 0022 08                 INX
00560 00056P 0023 08                 INX               X:=X+2
00570 00057P 0024 FF 0002  D         STX     POINT2    POINT2:=X
00580 00058                          *
00590 00059P 0027 7A 0004  D         DEC     CNT       CNT:=CNT-1
00600 00060P 002A 26 E5 0011         BNE     NEXTCH    JUMP IF <>0
00610 00061                          ********
00620 00062P 002C FC 0000  A         LDD     ADOUT     D:=A/D
00630 00063P 002F ED 00    A         STD     0,X       INDAT(X):=D-REG
00640 00064                          *
00650 00065P 0031 39                 RTS               RETURN FROM SUBROUTINE
00660 00066                          *
00670 00067                          END
TOTAL ERRORS 00000--00000


ADIN1  0005  ADOUT  0006  CNT    0004  INDAT  0007  INSAML 0000  NEXTCH 0011
POINT1 0000  POINT2 0002
```

```
PAGE  001  PIPROG0 .SA:1  PIPROG ************ PI-PROGRAM Nr. 0 ************

00010 00001                         OPT    Z01,REL,LLE=80,P=62
00020 00002                         NAM    PIPROG0
00030 00003                         TTL    ************ PI-PROGRAM Nr. 0 ******
00040 00004                       *
00050 00005                       *
00060 00006                       ************ PROGRAM DESCRIPTION ****************
00070 00007                       *
00080 00008                       * THIS PROGRAM IS PI-REGULATOR Nr. 0
00090 00009                       *
00100 00010                       ***************************************************
00110 00011                       *
00120 00012                       *
00130 00013                       ********** MACRO DEFINITIONS ********************
00140 00014                       *
00150 00015                       LIMIT  MACR                CHECKS IF OVERFLOW
      00016                       00160          BVC     ö0
      00017                       00170          BGT     ö.0
      00018                       00180          LDD     #$8000
      00019                       00190          BRA     ö0
      00020                       00200 ö.0      LDD     #$7FFF
      00021                       00210          ENDM
0[illegible]00022                 *
0[illegible]00023                 *
00240 00024                       ********* DATABASE DEFINITION *********************
00250 00025                       *
00260 00026D 0000                         DSCT
00270 00027                       *
00280 00028                               XREF   INSAML,REF,IS,K,TS.TI,DA1,DA2
00290 00029                               XREF   ERR,PROP,INT
00300 00030                       *
00310 00031                               XDEF   PIREG
00320 00032                       *
00330 00033             00FF  A HLIM      EQU    $00FF
00340 00034             FF00  A LLIM      EQU    $FF00
00350 00035                       *
00360 00036D 0000       0002  A SUM       RMB    2
00370 00037D 0002       0002  A OUT       RMB    2
00380 00038D 0004       0002  A MLAG      RMB    2
0[illegible]390 00039D 0006     0002  A NLAG      RMB    2
00400 00040                       *
00410 00041                       ***************************************************
```

```
PAGE  001  PIPROG1 .SA:1  PIPROG ************ PI-PROGRAM Nr. 1 ************

00010 00001                              OPT    Z01,REL,LLE=80,P=62
00020 00002                              NAM    PIPROG1
00030 00003                              TTL    ************ PI-PROGRAM Nr. 1 *****
00040 00004                         *
00050 00005                         *
00060 00006                         ************ PROGRAM DESCRIPTION ***************
00070 00007                         *
00080 00008                         * THIS PROGRAM IS PI-REGULATOR Nr. 1
00090 00009                         *
00100 00010                         ***************************************************
00110 00011                         *
00120 00012                         *
00130 00013                         ********** MACRO DEFINITIONS ********************
00140 00014                         *
00150 00015                         LIMIT  MACR            CHECKS IF OVERFLOW
      00016                         00160          BVC     ö0
      00017                         00170          BGT     ö.0
      00018                         00180          LDD     #$8000
      00019                         00190          BRA     ö0
      00020                         00200 ö.0      LDD     #$7FFF
      00021                         00210          ENDM
0^220 00022                         *
 :30 00023                          *
00240 00024                         ********* DATABASE DEFINITION *********************
00250 00025                         *
00260 00026D 0000                          DSCT
00270 00027                         *
00280 00028                                XREF   INSAML,REF,IS,K,TS.TI,DA1,DA2
00290 00029                                XREF   ERR,PROP,INT
00300 00030                         *
00310 00031                                XDEF   PIREG
00320 00032                         *
00330 00033            00FF  A HLIM    EQU    $00FF
00340 00034            FF00  A LLIM    EQU    $FF00
00350 00035                         *
00360 00036D 0000      0002  A SUM     RMB    2
00370 00037D 0002      0002  A OUT     RMB    2
00380 00038D 0004      0002  A MLAG    RMB    2
( 390 00039D 0006      0002  A NLAG    RMB    2
0.400 00040                         *
00410 00041                         ***************************************************
```

```
00430 00043                          *
00440 00044                          ********** PROGRAM SECTION **********
00450 00045                          *
00460 00046P 0000                          PSCT
00470 00047                          *
00480 00048                          * GET ANALOG DATA FROM SUBROUTINE INSAML
00490 00049P 0000 BD 0000  A PIREG   JSR     INSAML
00500 00050                          *
00510 00051P 0003 FC 0000  A         LDD     REF        D-REG:=REF
00520 00052P 0006 B3 0000  A         SUBD    IS         D-REG:=REF-IS
00530 00053P 0009                    LIMIT   OK1        LIMIT RESULT
00540 00054P 0015 FD 0000  A OK1     STD     ERR        ERR:=D-REG
00550 00055P 0018 FD 0000  A         STD     DA2        DA2:=D-REG
00560 00056P 001B 2A 04 0021         BPL     UT0        JUMP TO UT0 IF ERR POSITIVE
00570 00057P 001D 40                 NEGA               A:=00 - A
00580 00058P 001E 50                 NEGB               B:=00 - B
00590 00059P 001F 80 01     A        SUBA    #01        CONVERT TO POSITIVE
00600 00060                          *
00610 00061P 0021 05         UT0     LSLD               EXCLUDE SIGN BIT
00620 00062P 0022 F6 0000  A         LDAB    K          B-REG:=K
00630 00063P 0025 3D                 MUL                D-REG:=A-REG*B-REG
00640 00064P 0026 04                 LSRD
00650 00065P 0027 04                 LSRD
00660 00066P 0028 04                 LSRD
00670 00067P 0029 04                 LSRD               SHIFT RIGHT FOUR TIMES
00680 00068P 002A FD 0000  A         STD     PROP       PROP:=D-REG
00690 00069                          *
00700 00070P 002D F6 0000  A         LDAB    TS.TI      B-REG:=TS.TI
00710 00071P 0030 3D                 MUL                D-REG:=PROP*TS.TI
00720 00072P 0031 FD 0004  D         STD     MLAG       MLAG:=D-REG
00730 00073P 0034 B6 0001  A         LDAA    PROP+1     A-REG:=PROP+1
00740 00074P 0037 F6 0000  A         LDAB    TS.TI      B-REG:=TS.TI
00750 00075P 003A 3D                 MUL                D-REG:=PROP+1*TS.TI
00760 00076P 003B 16                 TAB                B-REG:=A-REG
00770 00077P 003C 86 00     A        LDAA    #$00       A-REG:=00
00780 00078P 003E F3 0004  D         ADDD    MLAG       D-REG:=D-REG+MLAG
00790 00079P 0041                    LIMIT   OK2        LIMIT RESULT
00800 00080P 004D FD 0004  D OK2     STD     MLAG       MLAG:=D-REG
00810 00081                          *
00820 00082P 0050 FC 0000  A         LDD     ERR        D-REG:=ERR
00830 00083P 0053 2B 2A 007F         BMI     MINUS      TEST IF PROP NEGATIVE
00840 00084                          *
00850 00085P 0055 FC 0004  D         LDD     MLAG       D-REG:=MLAG
00860 00086P 0058 F3 0000  A         ADDD    INT        D-REG:=MLAG+INT
00870 00087P 005B                    LIMIT   OK3        LIMIT RESULT
00880 00088P 0067 FD 0000  A OK3     STD     INT        INT:=D-REG
00890 00089P 006A F3 0000  A         ADDD    PROP       D-REG:=INT+PROP
00900 00090P 006D                    LIMIT   OK4        LIMIT RESULT
00910 00091P 0079 FD 0000  D OK4     STD     SUM        SUM:=D-REG
00920 00092P 007C 7E 00A6  P         JMP     UT1        JUMP TO UT1
00930 00093                          *
00940 00094P 007F FC 0000  A MINUS   LDD     INT        D-REG:=INT
00950 00095P 0082 B3 0004  D         SUBD    MLAG       D-REG:=INT - MLAG
00960 00096P 0085                    LIMIT   OK5        LIMIT RESULT
00970 00097P 0091 FD 0000  A OK5     STD     INT        INT:=D-REG
00980 00098P 0094 B3 0000  A         SUBD    PROP       D-REG:=INT - PROP
00990 00099P 0097                    LIMIT   OK6        LIMIT RESULT
01000 00100P 00A3 FD 0000  D OK6     STD     SUM        SUM:=D-REG
01010 00101                          *
01020 00102P 00A6 2B 23 00CB UT1     BMI     UT2        JUMP TO UT2 IF SUM NEGATIVE
01030 00103P 00A8 CC 00FF  A         LDD     #HLIM      A-REG:=HLIM
01040 00104P 00AB B3 0000  D         SUBD    SUM        D-REG:=D-REG - SUM
```

```
01050 00105P 00AE 2A 55 0105          BPL     UT4       JUMP TO UT4 IF D-REG POSITIV
01060 00106                           ◆
01070 00107P 00B0 CC 00FF  A          LDD     #HLIM     A-REG:=HLIM
01080 00108P 00B3 FD 0000  D          STD     SUM       SUM:=D-REG
01090 00109P 00B6 B3 0000  A          SUBD    PROP      D-REG:=SUM - PROP
01100 00110P 00B9                     LIMIT   OK7       LIMIT RESULT
01110 00111P 00C5 FD 0000  A OK7      STD     INT       INT:=D-REG
01120 00112P 00C8 7E 0105  P          JMP     UT4       JUMP TO UT4
01130 00113                           ◆
01140 00114P 00CB CC FF00  A UT2      LDD     #LLIM     A-REG:=LLIM
01150 00115P 00CE FD 0006  D          STD     NLAG      NLAG:=D-REG
01160 00116P 00D1 FC 0000  D          LDD     SUM       D-REG:=SUM
01170 00117P 00D4 B3 0006  D          SUBD    NLAG      D-REG:=SUM - LLIM
01180 00118P 00D7 2A 18 00F1          BPL     UT3       JUMP TO UT3 IF SUM > LLIM
01190 00119                           ◆
01200 00120P 00D9 CC FF00  A          LDD     #LLIM     A-REG:=LLIM
01210 00121P 00DC FD 0000  D          STD     SUM       SUM:=D-REG
01220 00122P 00DF F3 0000  A          ADDD    PROP      D-REG:SUM - PROP
01230 00123P 00E2                     LIMIT   OK8       LIMIT RESULT
01240 00124P 00EE FD 0000  A OK8      STD     INT       INT:= D-REG
01250 00125                           ◆
01260 00126P 00F1 FC 0000  D UT3      LDD     SUM       D-REG:=SUM
01270 00127P 00F4 40                  NEGA              A:=00 - A
01280 00128P 00F5 50                  NEGB              B:=00 - B
01290 00129P 00F6 80 01    A          SUBA    #$01      CONVERT TO POSITIVE
01300 00130P 00F8 81 02    A          CMPA    #$02      COMPARE A WITH $02
01310 00131P 00FA 2B 19 0115          BMI     UT5       JUMP TO UT5 IF A - $02 NEGAT
01320 00132P 00FC CC 8000  A          LDD     #$8000    D-REG:=$8000
01330 00133P 00FF FD 0002  D          STD     OUT       OUT:=D-REG
01340 00134P 0102 7E 0121  P          JMP     UT6       JUMP TO UT6
01350 00135P 0105 FC 0000  D UT4      LDD     SUM       D-REG:=SUM
01360 00136P 0108 81 02    A          CMPA    #$02      COMPARE A WITH $02
01370 00137P 010A 2B 09 0115          BMI     UT5       JUMP TO UT5 IF A - $02 NEGAT
01380 00138P 010C CC 7FC0  A          LDD     #$7FC0    D-REG:=$7FC0
01390 00139P 010F FD 0002  D          STD     OUT       OUT:=D-REG
01400 00140P 0112 7E 0121  P          JMP     UT6       JUMP TO UT6
01410 00141P 0115 FC 0000  D UT5      LDD     SUM       D-REG:=SUM
01420 00142P 0118 05                  ASLD
01430 00143P 0119 05                  ASLD
01440 00144P 011A 05                  ASLD
01450 00145P 011B 05                  ASLD
01460 00146P 011C 05                  ASLD
01470 00147P 011D 05                  ASLD              SHIFT LEFT SIX TIMES
01480 00148P 011E FD 0002  D          STD     OUT       OUT:=D-REG
01490 00149                           ◆
01500 00150P 0121 FC 0002  D UT6      LDD     OUT       D-REG:=OUT
01510 00151P 0124 FD 0000  A          STD     DA1       DA1:=D-REG
01520 00152                           ◆
01530 00153P 0127 0E                  CLI
01540 00154                           ◆
01550 00155P 0128 3B                  RTI
01560 00156                           ◆
01570 00157                           END
TOTAL ERRORS 00000--00000
```

```
.00000 0012   .00001 004A   .00002 0064   .00003 0076   .00004 008E   .00005 00A0
.00006 00C2   .00007 00EB   DA1     0005   DA2     0006   ERR     0007   HLIM    00FF
INSAML 0008   INT     0009   IS      000A   K       000B   LLIM    FF00   MINUS   007F
MLAG   0004   NLAG    0006   OK1     0015   OK2     004D   OK3     0067   OK4     0079
OK5    0091   OK6     00A3   OK7     00C5   OK8     00EE   OUT     0002   PIREG   0000
```

```
PROP   0008  REF    000E  SUM    0000  TS.TI  000F  UT0    0021  UT1    00A6
UT2    00CB  UT3    00F1  UT4    0105  UT5    0115  UT6    0121
```

```
PAGE  001  MAPFI1  .SA:1

  NO UNDEFINED SYMBOLS

MEMORY MAP

  S SIZE  STR  END COMN
  B 0000 0020 0020 0000
  C 0000 0020 0020 0000
  D 002F 1800 182E 0000
  P 0629 1000 1628 0000

MODULE NAME BSCT DSCT PSCT
  INIT      0020 1800 1000
  RESTAR    0020 1818 1100
  KONST     0020 1818 1200
  RTC       0020 1822 1300
  INSAML    0020 1822 1400
  PIPROG    0020 1827 1500

DEFINED SYMBOLS


 'DULE NAME: INIT
  )DIN1  A 2000    ADOUT  A 200C    DA1    A 2008    DA2    A 200A
  DIP2   A 300D    DIP3   A 300E    FRC    A 0009    INDAT  D 1804
  IS     D 1806    K      D 1800    OCR    A 000B    REF    D 1804
  RESET  A 3008    SAMPT  D 1816    TCSR   A 0008    TS.TI  D 1801
  TSAMP  A 300C

MODULE NAME: RESTAR
  REST   P 1100

MODULE NAME: KONST
  ERR    D 1818    INT    D 181E    INTOLD D 1820    KONST  P 1200
  PROP   D 181A    PROPOL D 181C

MODULE NAME: RTC
  INTER  P 1310    RTC    P 1300

 DULE NAME: INSAML
  INSAML P 1400

MODULE NAME: PIPROG
  PIREG  P 1500
```

```
PAGE  001  MINFEL  .SA:1  MINFEL ******** MINFEL SUBROUTINE **********

00010 00001                              OPT     Z01,REL,LLE=80,P=62
00020 00002                              NAM     MINFEL
00030 00003                              TTL     ******** MINFEL SUBROUTINE *********
00040 00004                         *
00050 00005                         *
00060 00006                         *
00070 00007                         ************ PROGRAM DESCRIPTION ******************
00080 00008                         *
00090 00009                         * THIS SUBROUTINE CALCULATES THE INTEGRAL OF THE E
00100 00010                         * DURING 5 s AFTER A CHANGE OF THE REFERENS SIGNAL
00110 00011                         * NEGATIVE TO POSITIVE.
00120 00012                         * TSAMP SHOULD BE SET TO 10ms BEFORE USING THIS SU
00130 00013                         * RUOTINE
00140 00014                         *
0@50 00015                          ***********************************************************
00160 00016                         *
0^170 00017                         *
  ,80 00018                         *
00190 00019                         ************ DATABASE DEFINITION ********************
00200 00020                         *
00210 00021D 0000                           DSCT
0^220 00022                         *
  230 00023                         *
00240 00024                                 XDEF    MINFEL
00250 00025                         *
00260 00026                                 XREF    REF,ERRNY,DA2,TS.TI
00270 00027                         *
00280 00028                         *
00290 00029D 0000     0002  A REFOLD RMB    2
00300 00030D 0002     0002  A RAKNA  RMB    2
00310 00031D 0004     0002  A FEL    RMB    2
00320 00032                         *
00330 00033                         *
00340 00034                         ************** MACRO DEFINITIONS ******************
00350 00035                         *
00360 00036                         SIGNCH MACR             CHANGE SIGN
      00037                         00370         BMI     ö.X
      00038                         00380         COMA
      00039                         00390         COMB
      00040                         00400         ADDD    #$0001
      00041                         00410         BRA     ö0
      00042                         00420 ö.X     COMA
      00043                         00430         COMB
      00044                         00440         SUBD    #$0001
      00045                         00450         ENDM
00460 00046                         *
00470 00047                         ***********************************************************
```

```
00490 00049                          *
00500 00050D 0006 FC 0000  A MINFEL  LDD    REF      D-REG:= REF
00510 00051D 0009 2A 0C 0017         BPL    KLAR1    JUMP TO KLAR1 IF REF POSITI'
00520 00052D 000B CC 0000  A         LDD    #$0000   D-REG:= 0000
00530 00053D 000E FD 0004  D         STD    FEL      FEL:= 0000
00540 00054D 0011 FD 0002  D         STD    RAKNA    RAKNA:= 0000
00550 00055D 0014 7E 005A  D         JMP    KLAR4    JUMP TO KLAR4
00560 00056                          *
00570 00057D 0017 FC 0000  D KLAR1   LDD    REFOLD   D-REG:= REFOLD
00580 00058D 001A 2A 09 0025         BPL    KLAR2    JUMP TO KLAR2 IF REFOLD POS
00590 00059D 001C CC 0000  A         LDD    #$0000   D-REG:= 0000
00600 00060D 001F FD 0002  D         STD    RAKNA    RAKNA:= 0000
00610 00061D 0022 FD 0004  D         STD    FEL      FEL:= 0000
00620 00062                          *
00630 00063D 0025 FC 0002  D KLAR2   LDD    RAKNA    D-REG:= RAKNA
00640 00064D 0028 83 01F4  A         SUBD   #500     D-REG:= RAKNA - 500
00650 00065D 002B 2A 2D 005A         BPL    KLAR4    JUMP TO KLAR4 IF RAKNA - 50
00660 00066                          *
00670 00067D 002D FC 0000  A         LDD    ERRNY    D-REG:= ERRNY
00680 00068D 0030 2A 0E 0040         BPL    KLAR3    JUMP TO KLAR3 IF ERRNY POSI
00690 00069D 0032                    SIGNCH KLAR3    CONVERT ERRNY TO POSITIVE
00700 00070D 0040 05           KLAR3 LSLD             EXCLUDE SIGN BIT
00710 00071D 0041 F6 0000  A         LDAB   TS.TI     B-REG:= TS.TI
00720 00072D 0044 3D                 MUL              D-REG:= ABS ERRNY * TS.TI
00730 00073D 0045 04                 LSRD
00740 00074D 0046 04                 LSRD
00750 00075D 0047 04                 LSRD
00760 00076D 0048 04                 LSRD
00770 00077D 0049 04                 LSRD
00780 00078D 004A 04                 LSRD             SHIFT RIGHT 6 TIMES
00790 00079D 004B F3 0004  D         ADDD   FEL      D-REG:= D-REG + FEL
00800 00080D 004E FD 0004  D         STD    FEL      FEL:= D-REG + FEL
00810 00081D 0051 FC 0002  D         LDD    RAKNA    D-REG:= RAKNA
00820 00082D 0054 C3 0001  A         ADDD   #$0001   D-REG:= RAKNA + 1
00830 00083D 0057 FD 0002  D         STD    RAKNA    RAKNA:= RAKNA + 1
00840 00084                          *
00850 00085                          *
00860 00086                          * RENEW VALUES OF REF
00870 00087                          *
00880 00088D 005A FC 0000  A KLAR4   LDD    REF      D-REG:= REF
00890 00089D 005D FD 0000  D         STD    REFOLD   REFOLD:= REF
00900 00090                          *
00910 00091                          *
00920 00092                          * CONVERT FEL FROM 16 BITS
00930 00093                          *                 TO 10 BITS
00940 00094                          *                 AND STORE FEL IN DA2
00950 00095                          *
00960 00096D 0060 FC 0004  D         LDD    FEL      D-REG:= FEL
00970 00097D 0063 81 02    A         CMPA   #$02     COMPARE A-REG WITH 02
00980 00098D 0065 2B 09 0070         BMI    KLAR5    JUMP TO KLAR5 IF NEGATIVE
00990 00099D 0067 CC 7FC0  A         LDD    #$7FC0   D-REG:= 7FC0
01000 00100D 006A FD 0000  A         STD    DA2      DA2:= 7FC0
01010 00101D 006D 7E 007C  D         JMP    KLAR6    JUMP TO KLAR6
01020 00102                          *
01030 00103D 0070 FC 0004  D KLAR5   LDD    FEL      D-REG:= FEL
01040 00104D 0073 05                 ASLD
01050 00105D 0074 05                 ASLD
01060 00106D 0075 05                 ASLD
01070 00107D 0076 05                 ASLD
01080 00108D 0077 05                 ASLD
01090 00109D 0078 05                 ASLD             SHIFT LEFT 6 TIMES
01100 00110D 0079 FD 0000  A         STD    DA2      DA2:= D-REG
```

```
01110 00111                          ♦
01120 00112D 007C 39          KLAR6  RTS              RETURN FROM SUBROUTINE
01130 00113                          ♦
01140 00114                          END
TOTAL ERRORS 00000--00000


.00000 003B  DR2     0005  ERRNY  0006  FEL    0004  KLAR1  0017  KLAR2  0025
KLAR3  0040  KLAR4   005A  KLAR5  0070  KLAR6  007C  MINFEL 0006  RAKNA  0002
REF    0008  REFOLD 0000  TS.TI  0009
```