Institut für Technische Physik an der ETH-Zürich

BEITRAEGE ZUR ENTWICKLUNG EINES DIGITALEN ECHTZEIT - SPEKTRALANALYSATORS FUER DAS ELEKTROENZEPHALOGRAMM

DIPLOMARBEIT

von

Bruno Fricker
Abt. IX B
Kehlhof 211
8572 Berg / TG

Zürich, den 1. September 1974

Vorwort

Im Frühjahr 1972 stiess das an der EEG-Abteilung des Kinderspitals von Herrn Dr. Dumermuth entwickelte Gedankengut eines festverdrahteten Fourierprozessors für das Elektroenzephalogramm auf die Gegenliebe unseres Institutes, an welchem sich schon seit längerer Zeit eine Gruppe mit signalanalytischen Projekten befasste. In darauffolgenden Semesterferien wurde ich beauftragt, die Vorbereitungen zur Formulierung eines detaillierten Projektes zu treffen.

Meine Diplomarbeit rundet all diese Bemühungen ab. Im ersten Kapitel enthält sie die Resultate der Simulation eines hardwareorientierten FFT-Algorithmus. Die Brauchbarkeit desselben als Kernstück des geplanten Prozessors hat sich bestätigt. Im zweiten Kapitel sind die verschiedenen Fehlerquellen, die der diskreten Fouriertransformation in Theorie und Praxis anhaften, zusammengefasst. Kapitel drei enthält meine Vorschläge zur Einbettungsfrage und Gesamtarchitektur. Und im vierten Kapitel schliesslich ist ein interessantes Verfahren zur nichtlinearen Verformung der Frequenzachse diskutiert, welches unser Projekt wesentlich bereichern wird. Im Anhang habe ich alle Arbeiten beigefügt, welche am Institut nicht in Form von Aktenvermerken greifbar sind.

Herrn PD Dr. Dumermuth möchte ich herzlich danken für die erfreuliche und stets stimulierende Zusammenarbeit. Mein Dank geht auch an meinen Betreuer, Herrn Roger Lagadec, der mir durch fruchtbare Diskussionen und Impulse aus etlichen Engpässen geholfen hat. Besonders danken möchte ich meiner lieben Frau für die sorgfältige Reinschrift des Manuskriptes. Herrn Prof. Baumann danke ich für sein immerwährendes Interesse, und auch dafür, dass die Arbeit unter seinem Dache nach und nach heranreifen durfte.

Zürich, den 1. September 1974

sig. B. Fricker

Inhaltsübersicht

1. DIGITALE SIMULATION EINES VERDRAHTETEN FFT-PROZESSORS

Meine Diplomarbeit stützt sich auf eine kleinere Zahl vorangehender Studien (12, 13, 14, 16), die hauptsächlich dem Zweck dienten, unter den zahlreichen Publikationen über FFT-Algorithmen und -Prozessoren ein für den EEG-Spektralanalysator geeignetes Verfahren zu finden.

Meine vorletzte derartige Untersuchung (14) enthält eine Darstellung des Zeitreihen-Analysators von Corinthios, den ich als besonders günstige Realisierung des Cooley-Tuckey-Algorithmus ansehe. Diese Arbeit existiert nicht als Aktenvermerk, ist aber Voraussetzung dieses Kapitels. Das hat mich bewogen, sie in überarbeiteter Form als Anhang A in die Diplomarbeit aufzunehmen.

1.1. Der Algorithmus von Pease

Pease hat eine matrizielle Darstellung des FFT-Algorithmus publiziert, die nicht bloss durch ihre Prägnanz auffällt (21). Die Verfahrensweisen des Matrizenkalküls, besonders die Faktorisierung mit Hilfe des Kroneckerprodukts, führen nämlich auf ganz interessante Varianten des iterativen Ablaufschemas der schnellen Fouriertransformation. Pease zerlegt die lineare Vektorabbildung,

$$\vec{F} = \frac{1}{N} T_N \vec{f} \qquad (1)$$

mit $\vec{f} = \mathrm{col}(f_0, f_1, \ldots, f_{N-1})$, $\vec{F} = \mathrm{col}(F_0, F_1, \ldots, F_{N-1})$

und $(T_N)_{rs} = \exp(2\pi jrs/N)$

als welche die diskrete Fouriertransformation auch angesehen werden kann, in eine Abfolge von Operatoren (Matrizen), die in drei Klassen zerfallen:

- der Summen/Differenz-Operator S für Wertepaare
- der diagonale Multiplikations-Operator M_i, welcher die zweite Hälfte des Vektors mit den komplexen Gewichten multipliziert
- der Permutations-Operator P_i' zur Entmischung der Vektorelemente

Die Transformationsmatrix kann, wie im Anhang gezeigt ist, folgendermassen faktorisiert werden:

$$T_N = \prod_{i=m}^{1} (P_i' M_i S) \quad , \quad N=2^m \tag{2}$$

Da $P_m' = M_m = I_N$ Einheitsmatrizen sind, wird die Transformationsmatrix für das Beispiel N = 8

$$T_8 = S P_2' M_2 S P_1' M_1 S \quad , \tag{3}$$

und die ganze Transformation ist ausgeschrieben

$$\vec{F} = \frac{1}{2} \begin{bmatrix} 1 & & & & 1 & & & \\ & 1 & & & & 1 & & \\ & & 1 & & & & 1 & \\ & & & 1 & & & & 1 \\ 1 & & & & -1 & & & \\ & 1 & & & & -1 & & \\ & & 1 & & & & -1 & \\ & & & 1 & & & & -1 \end{bmatrix} \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & w^0 & \\ & & & w^2 \end{bmatrix} \frac{1}{2} \begin{bmatrix} 1 & & 1 & \\ & 1 & & 1 \\ 1 & & -1 & \\ & 1 & & -1 \end{bmatrix} \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & w^0 & \\ & & & w^1 \end{bmatrix} \frac{1}{2} \begin{bmatrix} 1 & & 1 & \\ & 1 & & 1 \\ 1 & & -1 & \\ & 1 & & -1 \end{bmatrix} \vec{f} \tag{4}$$

Dabei ist festzuhalten, dass die Normierungskonstante 1/N in Faktoren 1/2 aufgeteilt wird und dass die Multiplikations- und Permutationsmatrizen in jedem Zyklus leicht verändert auftreten.

Diese eher ungebräuchliche FFT-Notation fügt sich - und das ist bemerkenswert - in die üblichen Arten digitaler Verschaltung ausgezeichnet ein. Die Schlussformel (4) suggeriert sozusagen eine Systemarchitektur, die äusserst einfach und übersichtlich ist und mit einem Minimum an Steuerlogik auskommt. Corinthios hat aufgrund des Algorithmus von Pease eine solche Maschine entwickelt und publiziert (4).

1.2. Die Maschine von Corinthios

Michael S. Corinthios' Spezialprozessor für Zeitreihen-Analyse ist vorzüglich geeignet, die inverse und direkte Fouriertransformation mit geringem Hardwareaufwand und grosser Effizienz durchzuführen. In skelettierter Minimalkonfiguration besteht er aus je einem dynamischen Eingangs- und Ausgangsschieberegister für komplexe Datenvektoren der Länge N. Die Register sind je geteilt in zwei Hälften der Länge N/2. Sie gestatten so, den Datenstrang in der Mitte abzugreifen. Dies ist für den Radix-2-FFT-Algorithmus unerlässlich. Weitere Anzapfungen sind nicht vorgesehen. Auf dem Weg vom Eingangs- zum Ausgangsregister werden die Wertepaare arithmetischen Verknüpfungen unterworfen. In der Regel arbeitet der Prozessor rekursiv, d.h. die Ausgangsvektoren stellen Zwischenresultate dar, welche ins Eingangsregister zurückgeschleust werden müssen. In dieser Rückführphase kann eine Permutation vorgenommen werden. Ein weiterer Vorteil: Die Grundstruktur ist ausser für schnelle Fouriertransformation auch für andere Zeitreihen-Prozeduren geeignet, z.B. für Windowing, Glättung, Faltung, Korrelation und digitale Filtrierung. Spätere Untersuchungen werden erweisen, ob sich dieses Auftauschema auch für Rechenabläufe in Richtung Transientenanalyse eignen könnte.

Typische Merkmale sind die dynamischen Schieberegister und das komplexe Multiplizierwerk.

Dynamische Schieberegister stellen die zur Zeit preisgünstigsten Speicherelemente mit vollelektronischem Zugriff dar. Der Tandembetrieb zwischen Ein- und Ausgangsregister und die Möglichkeit einer permutierenden Feedbacks reduzieren den Nachteil des auf den Ausgang beschränkten Wertezugriffs. Im Falle der schnellen Fouriertransformation ist die iterative Entmischung in Feedbackzyklen sogar eine ganz überlegene Lösung.

Das hardwaremässig anspruchsvollste "Organ" des Prozessors ist wohl das komplexe Multiplizierwerk. Die komplexe Multiplikation besteht aus 4 reellen Multiplikationen, einer Addition und einer Subtraktion. Neben den 3 komplexen Buffern für Operanden und Resultat ist noch mindestens ein reeller Buffer für Zwischenwerte erforderlich. Da die Multiplikation eine zeitraubende aber höchst wichtige Operation ist, und zwar bei allen Zeitreihen-Prozeduren, möchte ich empfehlen, ein völlig paralleles Festkomma-Multiplizierwerk für volle Wortlänge zu bauen. Eine Uebersicht über Strukturvarianten findet sich bei A. Shah (23). Es sollte damit möglich sein, mit einer äusserst einfachen Kontrollogik eine komplexe 12-Bit-Multiplikation in ca. 1 usec zu bewerkstelligen. Diese Geschwindigkeit ist zwar für FFT an sich in unserem Anwendungsbereich nicht notwendig. Sie wird aber unentbehrlich, falls im Echtzeitbetrieb digitale Frequenzverformung nach Oppenheim vorgeschaltet wird (siehe Kapitel 4).

Ich bin jedoch ganz grundsätzlich der Ansicht, dass für einen aussichtsreichen Prototyp, dessen Pflichtenheft noch keineswegs geschlossen ist, ein allzu haushälterisches Vorgehen sich eines Tages als kurzsichtig erweisen könnte. Man bedenke, wie interessant, entwicklungsfähig und zukunftsträchtig unser Arbeitsgebiet ist! Und noch ein Gedanke: Wer sagt, dass unser FFT-Prozessor ausschliesslich auf niederfrequente Biosignale Anwendung finden soll? -

Ueber die Organisation seiner Maschine verrät Corinthios einiges, aber nicht ausreichend viel, um den Nachbau ohne eigene Ueberlegung zu gestatten. Die Synthese von Corinthios' Skelett mit meinen Detailentwürfen, die sich namentlich auf die Steuerlogik beziehen, findet der Leser in Fig. 1 bis 3 skizziert. Sie zeigen die Rechnerorganisation zur schnellen Fouriertransformation; andere Prozeduren erscheinen dort nicht. Die Transformation zerfällt in drei Rechenschritte, die zeitlich ge-

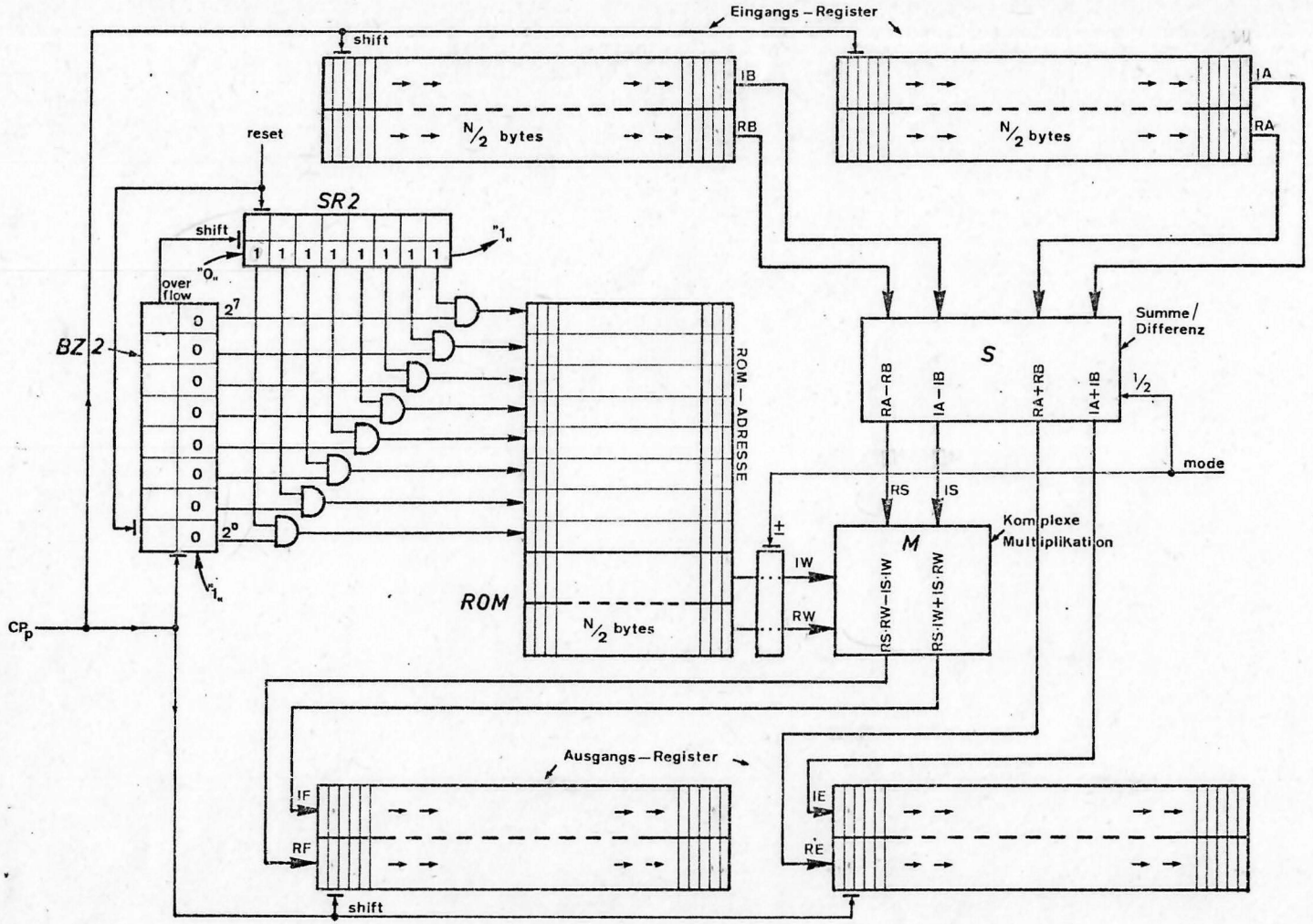


Fig. 10 "Processphase"des Fouriertransformators. Mit zwei komplexen Eingangswerten wird Summe und Differenz berechnet und bei Vorwärtstransformation durch zwei dividiert. Die Differenzen werden mit den im ROM gespeicherten komplexen Gewichtsfaktoren muliplizierd. Alle vier Werte werden im Ausgangsregister abgespeichert. R bedeutet Realteil; I bedeutet Imaginärteil.

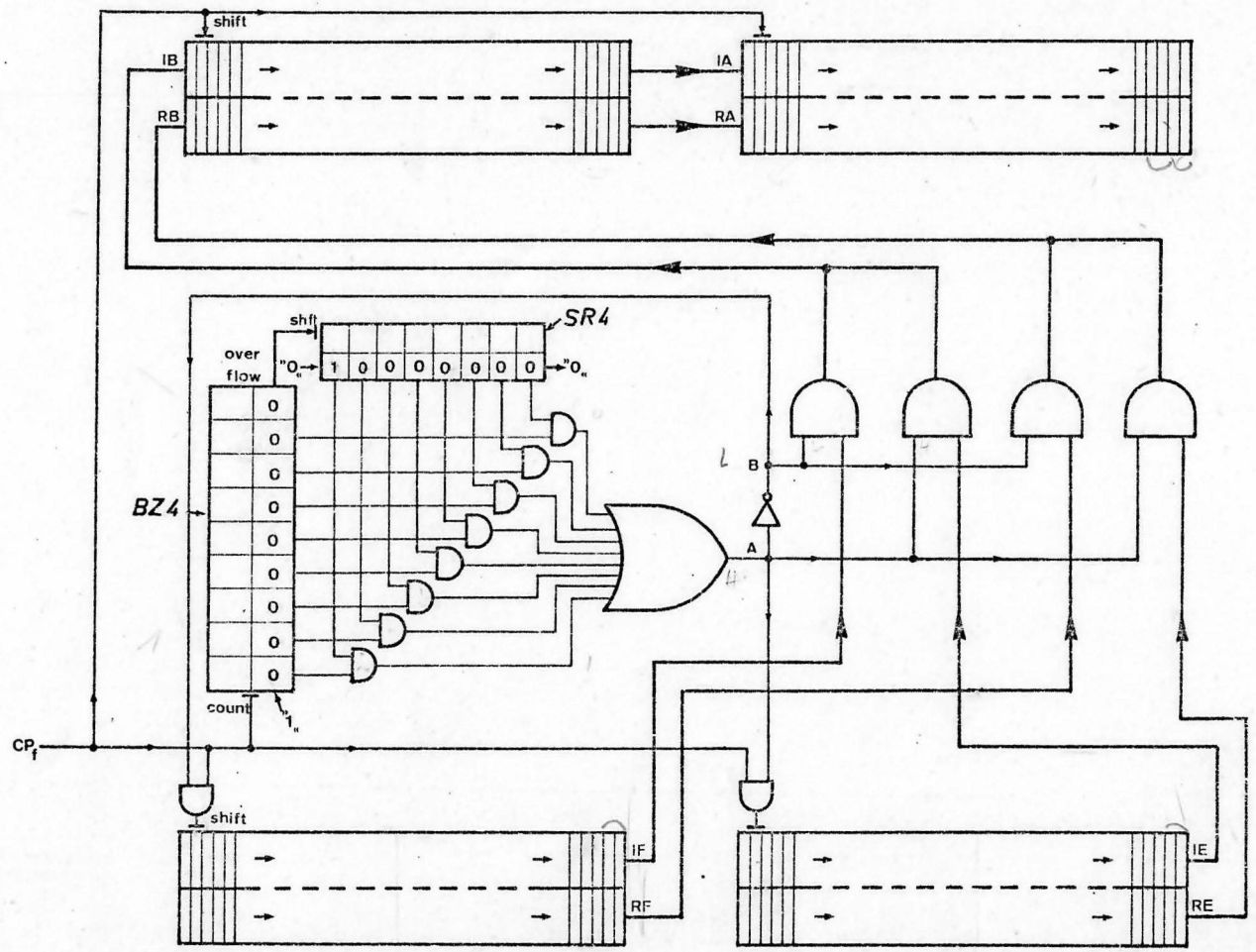


Fig. 11 "Feedbackphase" des Fouriertransformators. Die Werte im Ausgangsregister werden unter dem Einfluss der Steuerlogik BZ4 und SR4 in korrekter Reihenfolge ins Eingangsregister rückgeführt. Die beiden Hälften des Eingangsregisters sind in Serie geschaltet.

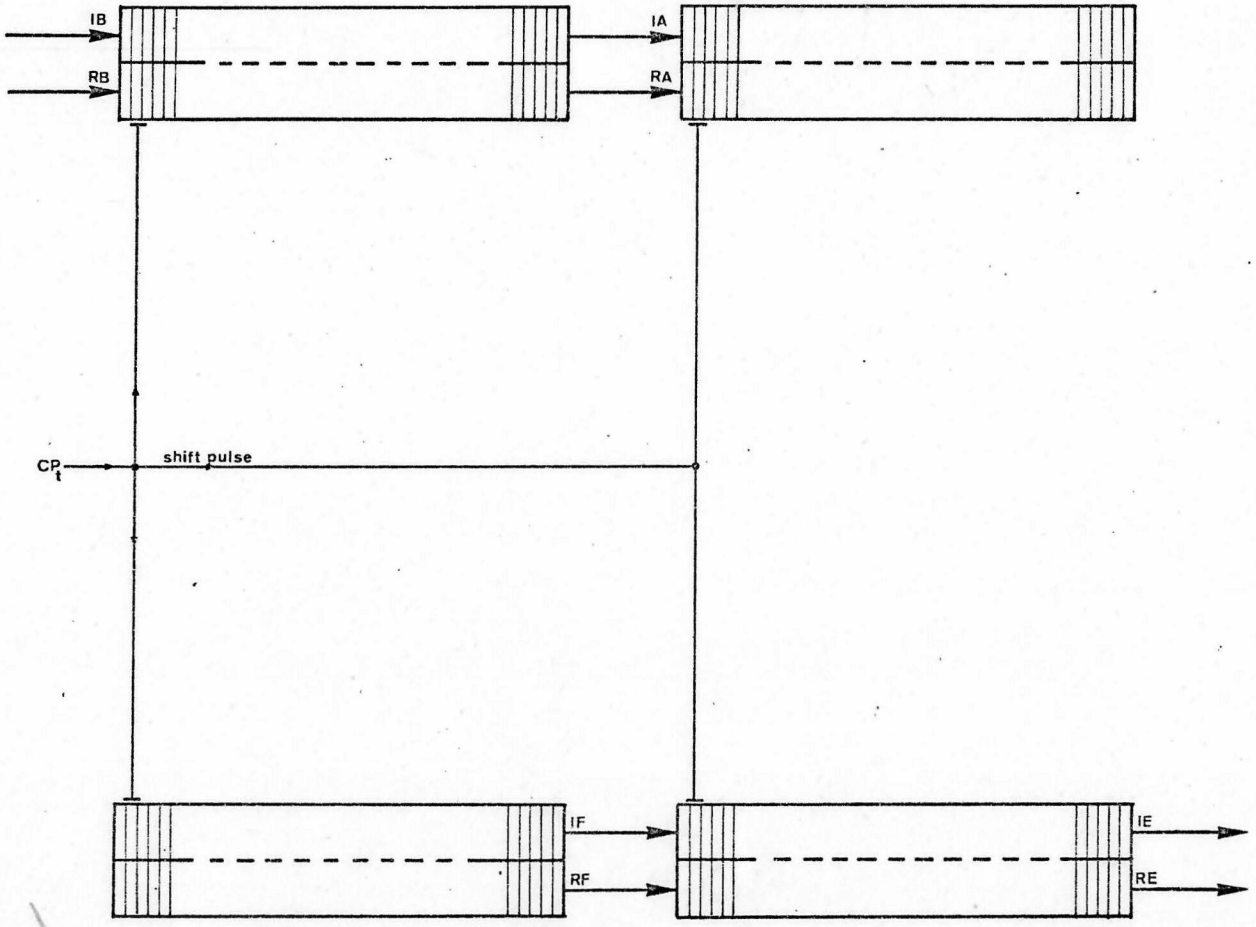


Fig. 3 "Transferphase" des Fouriertransformators. Der nächste Eingangsvektor wird vom A/D-Wandler ins Eingangsregister gelesen. Der fertige Ausgangsvektor wird zur Weiterverarbeitung in den Steuercomputer geführt.

staffelt ablaufen.

Die in Fig. 3 dargestellte "Transferphase" steht am Anfang jeder Zeitreihenkonversion. Das Ausgangsregister wird von alten Resultaten entleert, ins Eingangsregister laufen neue Sequenzen, auf welche operiert werden soll. Für die Geschwindigkeit dieses Vorgangs gibt CP_t der Transferclock, das Taktmass. Dynamische MOS-Schieberegister mit 10 MHz clock-rate sind heute ohne weiteres im Handel erhältlich.

Ausgangspunkt der "Process-" d.h. eigentlichen Rechenphase des schnellen Fouriertransformators (Fig. 1) ist die Anwesenheit einer komplexen Originalsequenz im Eingangsregister.
Durch sukzessives Verschieben und paarweises Abgreifen werden zunächst die Summe/Differenz-Terme berechnet, was der Matrixoperation ***S*** gemäss Gleichung (3) entspricht. Das mit einem Faktor 1/2 angedeutete "scaling" wird allenfalls auch in diesem Kästchen durchgeführt. Im Falle der inversen Fouriertransformation entfällt das scaling, dafür ändert im ROM das Vorzeichen der harmonischen Gewichtsfaktoren. Diese RW und FW laufen in ein Kästchen namens ***M*** und bilden die eine Hälfte der Operanden, deren andere von den beiden Differenztermen RS und IS konstituiert wird. In ***M*** wird komplex multipliziert (siehe die entsprechenden Operatoren in (4)) und die Resultate bei RF und IF im Ausgangsregister abgespeichert. Ins rechte halbe Ausgangsregister bei RE und IE laufen direkt die Summenwerte aus ***S*** .

Alle Speicherelemente, Datenpfade und Operationen sind komplex aufzufassen. Die diskrete Fouriertransformation führt einen komplexen Eingangsvektor in einen komplexen Ausgangsvektor über. Reelle Eingangsvektoren sind ein Spezialfall, der sich ausgangsseitig in der Symmetrie des Realteiles bzw. Antisymmetrie des Imaginärteiles wiederspiegelt. Das ist gleichbedeutend mit N/2

überflüssigen komplexen Ausgangswerten. Zwei reelle Zeitreihen werden mit Vorteil als einen komplexen Eingangsvektor aufgefasst und transformiert. Die Maschine ist dann voll ausgenützt und zeigt die obige Symmetrie nicht mehr. In der Folge allerdings müssen ausgangsseitig die Anteile der beiden unabhängigen Eingänge über eine Linearkombination entmischt werden, wie dies ebenfalls im Anhang A dargestellt ist.

Processphase und die in Fig. 2 aufgezeichnete "Feedbackphase" wechseln im Verlauf einer Fouriertransformation m-mal ab, wobei $2^m = N$ ist. Zur Erleichterung der Darstellung möchte ich das Paar Processphase + Feedbackphase als "Makrozyklus" bezeichnen. "Mikrozyklen" mögen die N/2 (bzw. N) Unterschritte heissen, aus denen die Process- bzw. Feedbackphase zusammengesetzt sind. Die Feedbackphase entspricht dem Permutationsoperator P (Gleichung (3)), der in jedem Makrozyklus eine andere Struktur hat. Die Gesamtheit all dieser m-1 Rückführungen unter Permutation entspricht genau jener bei FFT altbekannten Entmischung der Resultatvektoren, mit dem Unterschied allerdings, dass der rekursive Einbau der Entmischung in die ohnehin nötigen Feedbackzyklen ohne viel zusätzlichen Zeitaufwand erfolgen kann. Aus diesem Grund habe ich auch den bei Corinthios aufgegebenen "Post-Permutation Algorithm" nicht in Erwägung gezogen. Abschliessend einiges über Mikrozyklen und Steuerlogik:

Zu Beginn der Processphase (Fig. 1) wandern die Werte A und B vom Eingangsregister durch die Kästchen S und M ins Ausgangsregister. Vom ROM her stehen auch die Gewichte W an und werden verarbeitet. Dies alles läuft bei Parallelbetrieb ohne Uhrimpuls. Der Clock-Pulse CP_p inkrementiert den Binärzähler BZ 2, durch welchen die nächste (oder selbe, falls AND-Gate zu) ROM-Position angewählt wird. CP_p schiebt aber auch die Register um eine Position weiter. In der Folge laufen durch S und M neue Bitmuster,

die nach Ablauf einer gewissen Zeit T_p zu neuen Ausgangswerten E bzw. F führen. T_p wird durch den längsten Datenpfad bestimmt und kann bei Kenntnis der Organisation und Bestükkung der Kästchen ohne weiteres berechnet werden. CP_p ist demzufolge gleich $1/T_p$ plus eine gewisse Sicherheitsmarge. Nach N/2 Mikrozyklen ist diese Phase beendet. Während des ersten Makrozyklus ist das Schieberegister SR 2 mit lauter Einsen gefüllt. Der Binärzähler projiziert demnach vollkommen auf den Adresseingang des ROM-Speichers. Im zweiten Durchlauf ist die niedrigste Bit-Position gesperrt, da durch "overflow" der Einerblock im Schieberegister um eine Stelle nach rechts verschoben wurde. Die Adressierung erfolgt deshalb nur mit geraden Zahlen, wobei jeder Wert während zwei Mikrozyklen ansteht. Das nächste Mal sind die zwei untersten Bit-Positionen gesperrt, die Sprünge im ROM nach je vier Mikrozyklen umfassen jetzt 4 Zellen; etc. Derart werden die Gewichtssequenzen korrekt erzeugt. Erst ganz am Schluss nach m Makrozyklen wird die Logik über "reset" zurückgestellt.

Mit einer ganz ähnlichen Logik entsteht in Fig. 2 die richtige Permutation. Zu Beginn ist im Schieberegister SR 4 eine Marke 1, welche das unterste Tor öffnet. Die unterste Zelle des Zählers BZ 4 hinwiederum wechselt nach jedem Uhrimpuls CP_f ihren Inhalt, so dass die Daten-Tore von A bzw. B aus abwechselnd getastet werden. Demgemäss werden E bzw. F ins Eingangsregister B zurückgeschleust. Beim nächsten Makrozyklus steht die Marke 1 in SR 4 an der zweiten Stelle von links. Das zweitunterste UND-Gatter ist frei und nur dieses, so dass immer zwei E-Werte von zwei F-Werten abgelöst werden. Beim nächsten Durchlauf sind es 4er-Blöcke, usw. CP_f dürfte hierbei von der Logikgeschwindigkeit abhangen und ist bei bekannter Bestückung leicht zu errechnen. Die serielle Schaltung des Eingangsregisters bewirkt übrigens, dass diese Betriebsphase N und nicht bloss N/2 Mikrozyklen benötigt.

1.3. Die Nachbildung des Prozessors in Minicomputer-Software

Systemsimulation auf einem Allzweck-Computer ist dem Kriegsspiel verwandt, womit ohne Risiko komplizierte kausale Zusammenhänge durchgeprobt, Entscheide bedeutender Tragweite gefällt werden können. Wie im Sandkasten bildmässig jedes denkbare Manöver ablaufen kann, können im Computer im Prinzip logische Zusammenhänge beliebiger Komplexität programmiert und erprobt werden. Die Simulation ist in der Regel mit viel bescheidenerem finanziellen und zeitlichen Aufwand verbunden, als ein direkter Aufbau des fraglichen Systems es wäre.

Die Computersimulation ist in unserem Falle besonders angezeigt, da die Angaben beim Zeitreihen-Analysator von Corinthios so allgemein gehalten sind, dass man nicht geneigt ist, Tausende von Franken Hardwarekosten und viele Arbeitsstunden bloss in den Beweis zu investieren, dass Corinthios' Vorschlag tatsächlich funktioniert. Wichtige Fragen, wie die Optimierung der Wort- und Sequenzlänge, die Frage der Zahldarstellung und Organisation des Multiplizierwerks, das Problem des "windowing" und "smoothing", die Grösse der Ueberlappung im Zeitbereich können ebenfalls mit zum Teil erheblich weniger Umständlichkeit vor dem Bau im Allzweck-Computer gelöst werden.

Steht man vor der Aufgabe, ein digitales System zu simulieren, so muss man sich entscheiden, auf welcher Maschine und in welcher Programmiersprache man die Simulation bewerkstelligen will. Da es ohnehin notwendig wurde, den Computer gut kennenzulernen, mit welchem der EEG-Analysator dereinst verknüpft werden soll, entschieden wir uns für den PDP-12 Minicomputer von Herrn Dr. Dumermuth im EEG-Labor des Kinderspitals.

Der von der Firma Digital Equipment Corporation hergestellte Computer ist eine historisch gewachsene und äusserst vielseitige, ausgereifte Maschine, die sich gut für unser Problem eignet. Sie ist für den Anfänger besonders günstig, da sie über einen sehr komfortablen Editor verfügt, mit dem, in Verbindung mit einem CRT-Bildschirm und einer Tektronix-Speicherdisplay-Konsole, das Programmieren erleichtert wird. Fertige Programme können auf LINC-Tapes abgespeichert werden, die durch ihr handliches Format und ihre übersichtliche Organisation die Arbeit ebenfalls entbürden.

Im Laufe des Sommers wurden im EEG-Labor Plattenspeicher installiert, welche die Wartezeiten durch Umspulen, Ein- und Auslesen ganz wesentlich abkürzten. Die Plattenspeicher standen mir gegen Ende der Programmierarbeiten auch zur Verfügung; ausserdem kam mir die vielseitige und ausgereifte Programmbibliothek des Labors zustatten, was allerdings einige Zeit zum Kennenlernen und Einarbeiten erforderte.

Mehr als bei Grossrechenanlagen mit ihren benützerorientierten Befehlssprachen, ist man bei den beschränkteren Minicomputersystemen zu haushälterischem Programmieren verpflichtet. Die Assemblersprache steht deshalb im Vordergrund, die gegenüber FORTRAN, für welches der PDP-12 auch eingerichtet ist, enorme Vorteile bietet. Der 8K FORTRAN-Compiler ist mit seiner zweistufigen Compilation via symbolic language zum binary code sehr umständlich und beansprucht unverhältnismässig viel Speicherplatz. Assembler bietet aber den für eine Simulation wichtigen Vorteil, dass man die genaue Lage des Programmteils und der Datenvektoren im Kernspeicher ohne weiteres selber bestimmen kann. Man muss sich auch vergegenwärtigen, dass ein

geübter Assemblerprogrammierer, der über ein Repertoire von Subroutinen verfügt, fast gleich schnell programmiert, wie in FORTRAN. Der Eingriff der Assemblebefehle in die Organisation des Computers ist ausserdem viel übersichtlicher, präziser und vielfältiger als bei FORTRAN. Kurz, man hat den Minicomputer besser im Griff und kann seine ganze enorme Potenz voll ausschöpfen.

Weil ich die PDP-12 Maschinenkonfiguration sowieso kennenlernen wollte, habe ich deshalb überwiegend in Assembler programmiert, in welcher Sprache auch die zahlreichen Bibliotheksprogramme des EEG-Labors zur Verfügung standen. Es ist zu hoffen, dass das eine oder andere, nach geeigneter Modifizierung, zum Ausbau der institutseigenen HP-Software herangezogen werden könnte.

Der umfangreiche Assembler-Befehlsschatz des PDP-12-Systems ist eine weniger systematische als vielseitige "Verheiratung" der alten LINC-Computerbefehle mit den Befehlen des PDP-8-Systems. Dementsprechend arbeitet der Computer in zwei Betriebsarten: dem L-Mode (eher für Display, Band, A/D-Konversion, sense switches) und dem P-Mode (eher für Logik und Arithmetik, inkl. EAE, Teletype und Tektronix-Console, Disk). Die Grundeinheit des Kernspeichers mit 4096 (4k) 12-bit Wörtern zerfällt im L-Mode organisatorisch in vier 1024-Wort-Segmente und im P-Mode in 32 pages von je 128 Wörter. Im Kinderspital verfügt man über 8k Speicherumfang, also über zwei Grundeinheiten.

Mein Simulationsprogramm des Hardware-FFT-Prozessors von Corinthios läuft überwiegend im P-Mode. Deshalb halte ich mich bei der folgenden Besprechung an die Speicherordnung in pages. In Fig. 4a)+b) ist die Speicherzuordnung des FFT-Simulationsprogramms abgebildet.

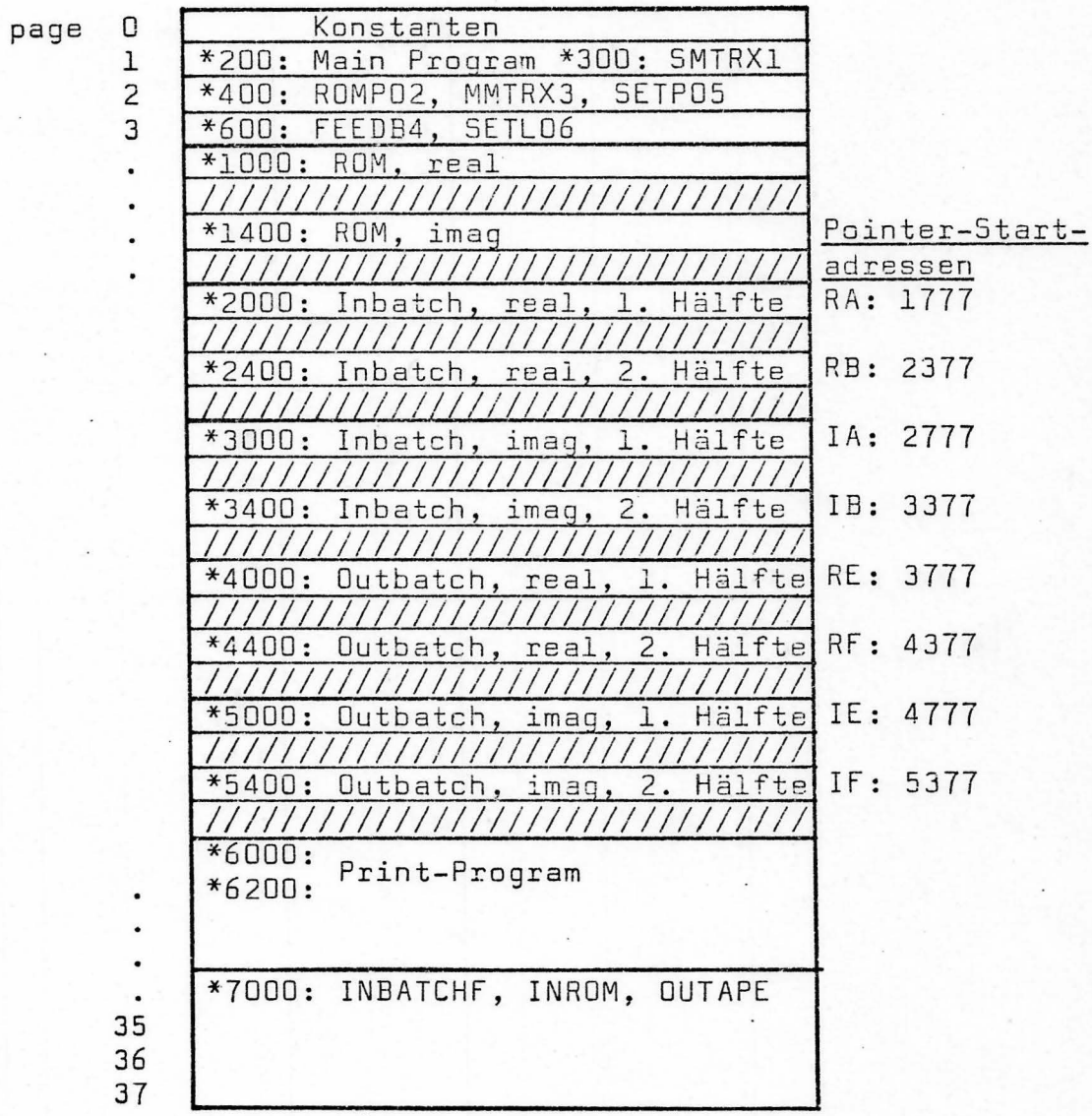


Fig.4a) Definitive Speicherordnung des Simulationsprogramms. Die schräggestrichenen Blöcke bleiben für N=256 frei. Für N=512 ist der Speicher bis *6200 völlig ausgenützt.

	Ø	1	2	3	4	5	6	7
Ø	/////	/////					RRPTR	IRPTR
1	RAØ	IAØ	RBØ	IBØ	REØ	IEØ	RFØ	IFØ
2	SR2Ø	BZ2Ø					RS	IS
3	SR4Ø	BZ4Ø						
4	/////	/////						
5								
6								
7								
1Ø	NØ	HNØ	MØ	MSC				
11								
12								
13								
14	/////	/////						
15								
16								
17								

Fig.4b) Aufteilung der Konstanten und Adressregister auf page 0. Die Zellen 0, 1, 40, 41, 140, 141 werden vom System beansprucht. 10 - 17 sind automatisch inkrementierende Pointer-Register. RRPTR=Realteil ROM-Pointer, IRPTR= Imaginärteil ROM-Pointer. Zu SR 2/4, BZ 2/4, RS, IS, vergleiche man Fig. 1 und 2.

Die Beschränkung auf die erste Grundeinheit gestattet Programme bis zu N=512 Punkte Segmentlänge. Neben den Sternen erscheinen die Speicheradressen als Oktalzahlen. Eine Seite umfasst $128_{10}=200_8$ Zellen. Page Ø ist für Konstanten und "pointers" reserviert, die aus der ganzen Einheit direkt angewählt werden können. Page 1 (*200 - *377) enthält das Hauptprogramm, für welches das Flussdiagramm Fig. 6 aufgestellt werden kann. Anschliessend folgen die verschiedenen Subroutinen. Pages 4 - 7 enthalten den Inhalt des ROM-Speichers, ab Adresse *1000 den Realteil, eine halbe Cosinusperiode, ab *1400 den Imaginärteil, eine halbe Sinusperiode. Durch Herauslesen des Kernspeichers auf Band und graphische Darstellung auf dem Tektronix-Speicherdisplay mit Hilfe eines Programmes LOOKPLOT[1] kann der ROM-Inhalt anschaulich gemacht werden (Fig. 5).

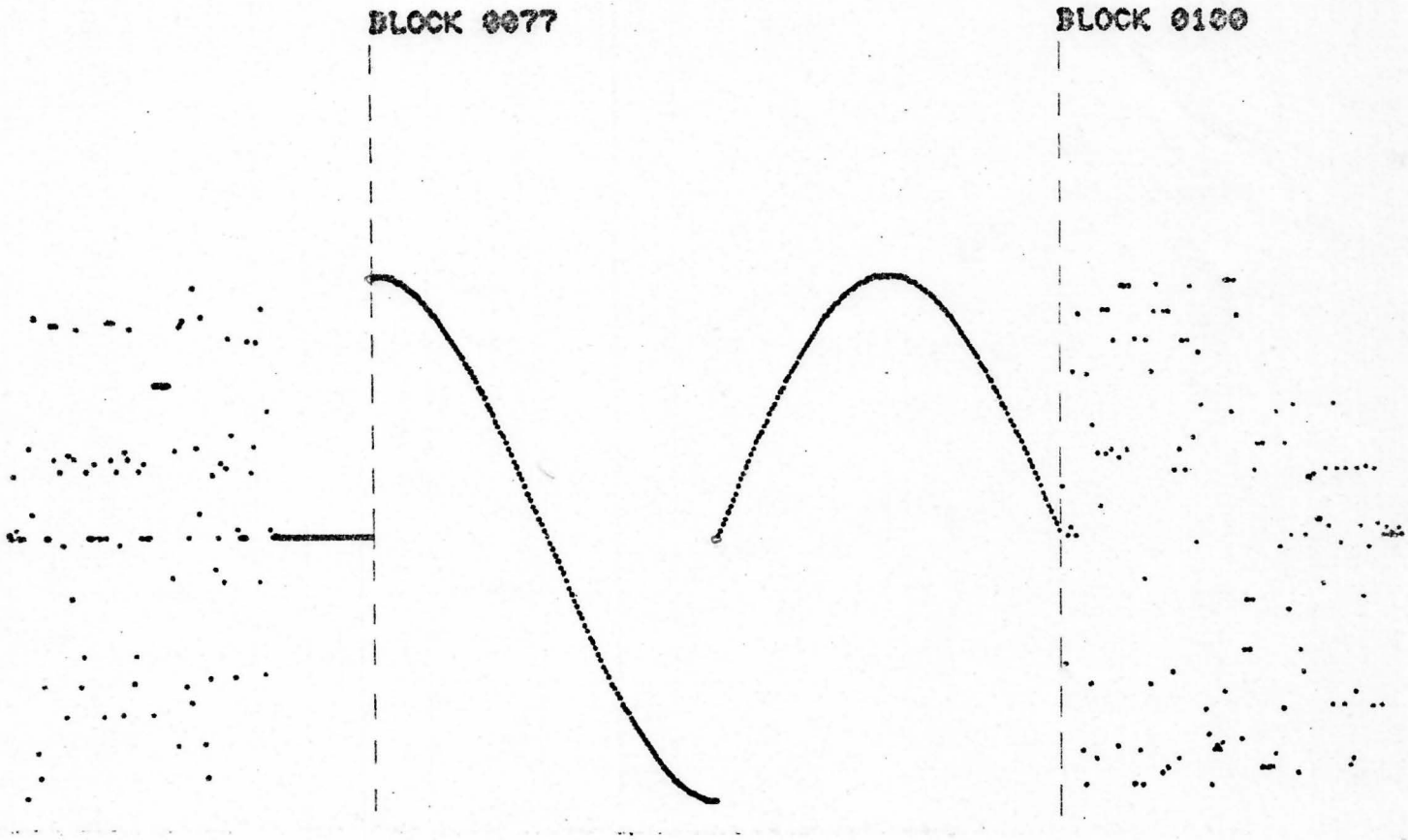


Fig. 5 Graphische Darstellung des auf LINC-Tape gespeicherten ROM-Inhaltes für N=256. Der Realteil befindet sich in der ersten, der Imaginärteil in der zweiten Hälfte von Block 0077. Ein LINCblock umfasst 256 12-bit Wörter.

Den Eingangsvektoren wurden die Plätze *2000 bis *3777 zugewiesen, den Ausgangsvektoren die Plätze *4000 bis *5777. Man kann leicht nachrechnen, dass alle Plätze nur für N=512 beansprucht werden. Für kleinere N bleibt unbenützter Zwischenraum. Die in der Fig. 4 a) rechts angegebenen Zahlen sind Startadressen für zusammenhängende Vektorstücke, die je von einem automatisch inkrementierenden Indexregister (pointer) aufgerufen werden. Die pointers tragen die Namen RA, RB, IA, IB, etc., wie sie auch im Schema Fig. 1 erscheinen. Sie übernehmen im Programm die Simulation der Registerverschiebung und sind, wie aus Fig. 4 b) ersichtlich ist, in page 0 unter den Konstanten zu finden. Im P-Mode sind die Zellen *14 - *17 automatische Indexregister, deren Inhalt als Träger der Zieladresse bei jeder indirekten Adressierung um Eins erhöht wird. So können die Vektorelemente reihenweise abgerufen werden. Schraffierte Zellen in Fig. 4 b) sind dem System vorbehaltene Plätze.

Das Flussdiagramm Fig. 6 zeigt den Ablauf der schnellen Fouriertransformation, wie er zur Simulation des verdrahteten Prozessors von Corinthios dienen kann. In den Kästchen stehen die Namen, unter welchen die Subroutinen aufgerufen werden. Am Schluss wird noch die Subroutine PROCT7 aufgerufen, welche auf den Teletype bzw. auf die Tektronix-Console eine Oktaltabelle mit dem Inhalt des Ausgangsregisters ausdruckt. Dieser Subroutine habe ich die Speicherplätze *6000 und folgende zugewiesen.

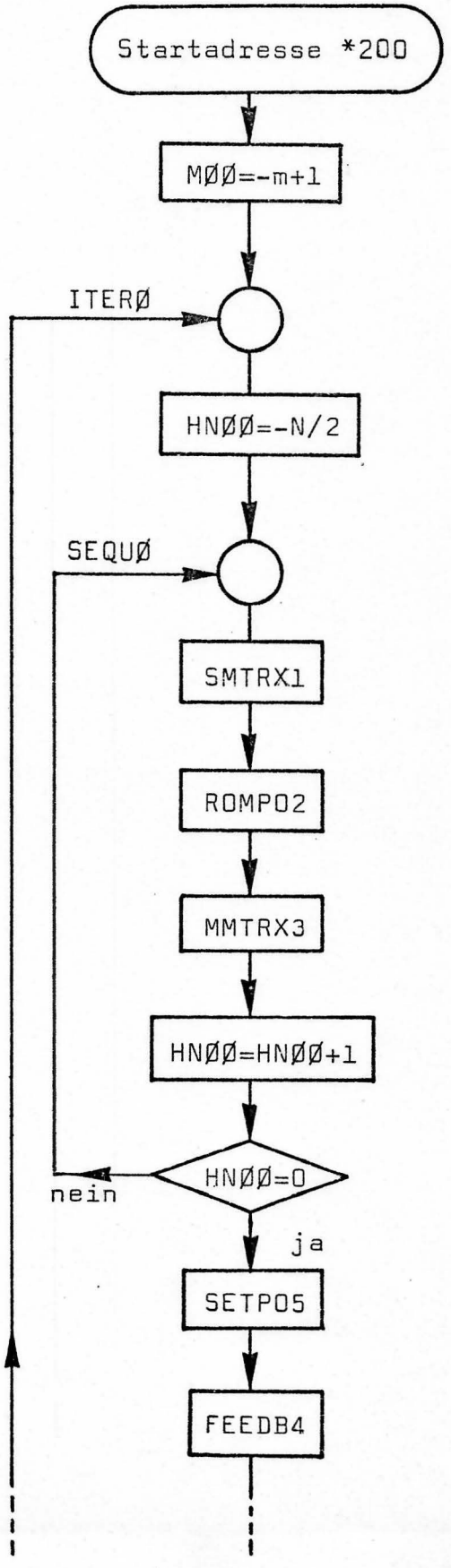
Auf den Seiten 20 bis 34 findet der Leser das vollständig gelistete Programm HARDWARE-FFT-SIMULATION. Am Programmkopf stehen unter der Startadresse *100 die grundsätzlichen Konstanten NØ (negative Segmentlänge N), HNØ (halbe negative Segmentlänge N), MØ (negativer Exponent m, wo $N=2^m$) und MSC (Maske zur Beschneidung der Wortlänge). Diese Konstanten können

---

[1] Dieses Programm wurde mir freundlicherweise von Herrn Germann zur Verfügung gestellt. Herr Germann macht als Arzt bei Dr. Dumermuth eine Dissertation über spektrale Peak-Statistik.

im Rahmen ihres innern Zusammenhangs frei gewählt werden und dirigieren alsdann den Ablauf des ganzen Programms.

Fig. 6:



sämtliche pointers werden korrekt initialisiert

$N=2^m$

Makrozyklus-Loop

Mikrozyklus-Loop

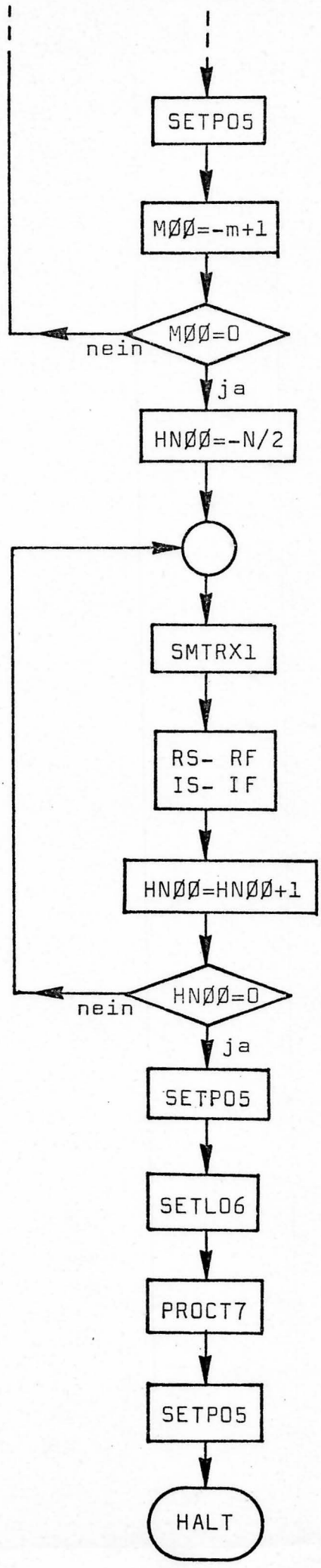
S-Matrix: Add. und Subtr.

simuliert ROM-Adressier-logik, aktuelle ROM-Adresse

komplexe Multiplikation

Rückstellung sämtlicher pointer

permutierender Feedback



Rückstellung sämtlicher pointer

letzter modifizierter Makrozyklus

S-Matrix: Add. + Subtr.

fülle linke Hälfte des Ausgangs unter Umgehung der Multiplikation

Rückstellung sämtlicher pointer

Rückstellung der Logik (Schieberegister und Binärzähler)

Oktaltabelle des Ausgangsregisters auf Tektronix-Display

Rückstellung sämtlicher pointer

```
0000                    *20
0001                    /=======================================
0002                    /===== HARDWARE FFT SIMULATION =========
0003                    /=======================================
0004
0005                    /PROGRAM NAME: F1256
0006
0007                            PMODE
0010
0011                    /INITIAL CONSTANTS
0012                    /-----------------
0013                            *100
0014    0100  7400      N0,     -400            /BATCH LENGTH N=256
0015    0101  7600      HN0,    -200
0016    0102  7770      M0,     -10
0017    0103  0022      MSC,    0
0020
0021                    /MAIN PROGRAM
0022                    /------------
0023                            *200
0024    0200  7300              CLA CLL
0025    0201  1102              TAD M0
0026    0202  7001              IAC
0027    0203  3246              DCA M00         /ITER0 LOOP INDEX
0030
0031                    /MASK FOR ROMPO LOGIC
0032    0204  1100              TAD N0
0033    0205  7041              CIA
0034    0206  7010              RAR
0035    0207  3611              DCA I MASC0
0036    0210  7410              SKP
0037    0211  0426      MASC0,  MASC2
0040
0041    0212  4653              JMS I SR5       /SET POINTERS
0042
0043                    /ITERATION LOOP
0044    0213  1101      ITER0,  TAD HN0
0045    0214  3247              DCA HN00
0046
0047                    /SEQUENCE LOOP
0050    0215  4300      SEQU0,  JMS SMTRX1      /ADDITION AND SUBTR.
0051    0216  4650              JMS I SR2       /SETS CURRENT ROM-POINT.
0052    0217  4651              JMS I SR3       /COMPLEX MULTIPLICATION
0053    0220  2247              ISZ HN00
0054    0221  5215              JMP SEQU0
0055
0056    0222  4653              JMS I SR5       /POINTERS RESET
0057    0223  4652              JMS I SR4       /FEEDBACK OUTBATCH
0060    0224  4653              JMS I SR5       /POINTERS RESET
0061    0225  2246              ISZ M00
0062    0226  5213              JMP ITER0
0063
0064    0227  1101              TAD HN0
0065    0230  3247              DCA HN00
0066
0067                    /FINAL RUN, DIRECT STORE IN OUTBATCH
0070
0071    0231  4300      SEQU00, JMS SMTRX1      /ADD. AND SUBTR.
0072    0232  1026              TAD 26
0073    0233  3416              DCA I AR0
0074    0234  1027              TAD 27
0075
-
```

```
0076    0235  3417          DCA I IF0
0077    0236  2247          ISZ HN00
0100    0237  5231          JMP SEQU00
0101
0102    0240  4653          JMS I SR5       /POINTERS RESET
0103    0241  4654          JMS I SR6       /RESET LOGIC
0104    0242  7402          HLT             /END FFT========
0105
0106                /PRINT OUTBATCH
0107    0243  4655          JMS I SR7       /BINARY OR OCTAL????
0110    0244  4653          JMS I SR5       /RESET LOGIC
0111
0112    0245  7402          HLT             /END MAIN PROGRAM///
0113                                        ////////////////////
0114                                        ////////////////////
0115    0246  0000  M00,     0
0116    0247  0000  HN00,    0
0117    0250  0400  SR2,     ROMPO2
0120    0251  0427  SR3,     MMTRX3
0121    0252  0600  SR4,     FEEDB4
0122    0253  0466  SR5,     SETPO5
0123    0254  0655  SR6,     SETLO6
0124    0255  6000  SR7,     PROCT7
0125
0126                        *20
0127    0020  7777  SR20,    7777            /INITIAL VALUES OF SR
0130    0021  0000  BZ20,    0000            /AND BZ FOR SUBR. 2
0131
0132                /AUTOINDEX REGISTERS FOR BATCH POINTER
0133                        *10
0134    0010  0000  RA0,     0
0135    0011  0000  IA0,     0
0136    0012  0000  RB0,     0
0137    0013  0000  IB0,     0
0140    0014  0000  RE0,     0
0141    0015  0000  IE0,     0
0142    0016  0000  RF0,     0
0143    0017  0000  IF0,     0
0144
0145                        *6
0146    0006  0000  RRPTR,   0               /REAL ROM POINTER
0147    0007  0000  IRPTR,   0               /IMAG.ROM POINTER
0150
0151                        *30
0152    0030  0001  SR40,    0001            /INITIAL VALUES OF SR
0153    0031  0000  BZ40,    0000            /AND BZ FOR SUBR. 4
0154                /=========================================
0155
0156                        *300
0157    0300  0000  SMTRX1,  0               //SUBROUTINE 1
0160    0301  7300          CLA CLL
0161    0302  1410          TAD I RA0
0162    0303  7415          ASR
0163    0304  0000          0
0164    0305  3342          DCA RA1
0165    0306  1411          TAD I IA0
0166    0307  7415          ASR
0167    0310  0000          0
0170    0311  3343          DCA IA1
0171    0312  1412          TAD I RB0
0172    0313  7415          ASR
0173    0314  0000          0
0174    0315  3344          DCA RB1
```

```
0175
          0316  1413            TAD I IB0
0176      0317  7415            ASR
0177      0320  0000            0
0200      0321  3345            DCA IB1
0201      0322  7100            CLL
0202
0203      0323  1342            TAD RA1           //COMPLEX ADDITION
0204      0324  1344            TAD RB1           / A + B
0205      0325  3414            DCA I RE0
0206      0326  1343            TAD IA1
0207      0327  1345            TAD IB1
0210      0330  3415            DCA I IE0
0211
0212      0331  1344            TAD RB1           //COMPL. SUBTRACTION
0213                                              / A - B
0214      0332  7040            CMA               /1COMPL. AC
0215      0333  1342            TAD RA1
0216      0334  3026            DCA 26            /C(26) = RE[A-B]
0217      0335  1345            TAD IB1
0220      0336  7040            CMA
0221      0337  1343            TAD IA1
0222      0340  3027            DCA 27            /C(27) = IM[A-B]
0223
0224      0341  5700  END1,     JMP I SMIRX1      /END SUBR. 1
0225                                              ////////////////
0226      0342  0000  RA1,      0
0227      0343  0000  IA1,      0
0230      0344  0000  RB1,      0
0231      0345  0000  IB1,      0
0232                            ASR=7415
0233
0234                            *400
0235      0400  0000  ROMP02,   0                 //SUBROUTINE 2
0236      0401  1021            TAD BZ20
0237      0402  0020            AND SR20          /AND-GATES, ROM-ADRESS
0240                                              /Y IN AC
0241      0403  7421            MQL               /0->MQ, AC->MQ, 0->AC
0242
0243      0404  1224            TAD RSTRT2        //ROM-POINTER RE IN 22
0244      0405  7501            MQA               /AC=1200!Y
0245      0406  3006            DCA RRPTR         /AC->REAL ROM POINTER
0246
0247      0407  1225            TAD ISTRI2        //ROM-POINTER IM IN 23
0250      0410  7501            MQA               /AC=1200!Y
0251      0411  3007            DCA IRPTR         /AC->IMAG.ROM POINTER
0252
0253      0412  2021            ISZ BZ20          //INCR. BZ
0254      0413  1021            TAD BZ20
0255      0414  0226            AND MASC2         /C(MASC2)=0004
0256      0415  7650            SNA CLA           /SKIP IF AC=/=0
0257      0416  5223            JMP CNT1          /I.E. IF BZ=4
0260
0261      0417  1020            TAD SR20          //SHIFT SR 1 STEP
0262      0420  7104            CLL RAL           /IF BZ=4
0263      0421  3020            DCA SR20
0264      0422  3021            DCA BZ20          /CLEAR BZ
0265
0266      0423  5600  CNT1,     JMP I ROMP02      /END SUBR. 2 ///
0267
0270      0424  1000  RSTRT2,   1000              /REAL ROM START ADR.
0271      0425  1400  ISTRI2,   1400              /IMAG.ROM START ADR.
0272      0426  0000  MASC2,    0
0273                            MQL=7421
-
```

```
0274                                MQA=7501
0275                                                    ////////////////
0276    0427  0000  MMTRX3, 0                          //SUBROUTINE 3
0277    0430  7300          CLA CLL
0300    0481  6141          LINC
0301                        LMODE
0302
0303    0432  1006  RPROD,  LDA RRPTR                  /LOAD REAL.ROW-PTR
0304    0433  1240          MUL                         /MULTIPLY FRACTIONAL
0305    0434  4026          4000+26                     /WITH C(26) -> AC
0306    0435  4465          STC P3                      /STORE RESULT IN P3, CLA
0307
0310    0436  1007          LDA IRPTR
0311    0437  1240          MUL
0312    0440  4027          4000+27
0313    0441  0017          COM                         /NEGATE RESULT
0314    0442  2465          ADD P3                      /ADD PREVIOUS RESULT
0315    0443  4463          STC ZWSPRF                  /INTERSTORE RF
0316
0317    0444  1006  IPROD,  LDA RRPTR
0320    0445  1240          MUL
0321    0446  4027          4000+27
0322    0447  4465          STC P3
0323
0324    0450  1007          LDA IRPTR
0325    0451  1240          MUL
0326    0452  4026          4000+26
0327    0453  2465          ADD P3
0330
0331    0454  4464          STC ZWSPIF                  /INTERSTORE IF
0332    0455  0002          PDP
0333                        PMODE
0334
0335    0456  1263          TAD ZWSPRFR                 //LOAD IND.IN OUTBATCH
0336    0457  3416          DCA I RF0
0337    0460  1264          TAD ZWSPIF
0340    0461  3417          DCA I IF0
0341    0462  5627          JMP I MMTRX3
0342
0343    0463  0000  ZWSPRF, 0
0344    0464  0000  ZWSPIF, 0
0345    0465  0000  P3,     0                           ////////////////
0346
0347
0350    0466  0000  SETP05, 0                          //SUBROUTINE 5
0351    0467  7300          CLA CLL                     //RESETS AUTOINDEX REG.
0352    0470  1311          TAD RA5
0353    0471  3010          DCA RA0
0354    0472  1313          TAD IA5
0355    0473  3011          DCA IA0
0356    0474  1312          TAD RB5
0357    0475  3012          DCA RB0
0360    0476  1314          TAD IB5
0361    0477  3013          DCA IB0
0362    0500  1315          TAD RE5
0363    0501  3014          DCA RE0
0364    0502  1317          TAD IE5
0365    0503  3015          DCA IE0
0366    0504  1316          TAD RF5
0367    0505  3016          DCA RF0
0370    0506  1320          TAD IF5
0371    0507  3017          DCA IF0
0372
-
```

```
0373    0510  5666          JMP I SETP05
0374
0375    0511  1777  RA5,    1777
0376    0512  2377  RB5,    2377
0377    0513  2777  IA5,    2777
0400    0514  3377  IB5,    3377
0401    0515  3777  RE5,    3777
0402    0516  4377  RF5,    4377
0403    0517  4777  IE5,    4777
0404    0520  5377  IF5,    5377
0405
0406                *600
0407    0600  0000  FEEDB4, 0                 //SUBROUTINE 4
0410    0601  7300          CLA CLL
0411    0602  1101          TAD HN0
0412    0603  3254          DCA INCR34
0413
0414    0604  7650  LOOP34, SNA CLA           /IF GATEA=0 -> CNT24
0415    0605  5213          JMP CNT24
0416
0417    0606  1416          TAD I RF0         //F -> A
0420    0607  3410          DCA I RA0
0421    0610  1417          TAD I IF0
0422    0611  3411          DCA I IA0
0423    0612  5217          JMP CNT34
0424
0425    0613  1414  CNT24,  TAD I RE0         //E -> A
0426    0614  3410          DCA I RA0
0427    0615  1415          TAD I IE0
0430    0616  3411          DCA I IA0
0431
0432    0617  2031  CNT34,  ISZ BZ40
0433    0620  1031          TAD BZ40
0434    0621  0030          AND SR40
0435    0622  2254          ISZ INCR34        /SKIP IF INCR34=0
0436    0623  5204          JMP LOOP34
0437
0440    0624  1101          TAD HN0
0441    0625  3254          DCA INCR34
0442
0443    0626  7650  LOOP44, SNA CLA           /IF GATEA=0 -> CNT224
0444    0627  5235          JMP CNT224
0445
0446    0630  1416          TAD I RF0
0447    0631  3412          DCA I RB0
0450    0632  1417          TAD I IF0
0451    0633  3413          DCA I IB0
0452    0634  5241          JMP CNT334
0453
0454    0635  1414  CNT224, TAD I RE0
0455    0636  3412          DCA I RB0
0456    0637  1415          TAD I IE0
0457    0640  3413          DCA I IB0
0460
0461    0641  2031  CNT334, ISZ BZ40
0462    0642  1031          TAD BZ40
0463    0643  0030          AND SR40
0464    0644  2254          ISZ INCR34
0465    0645  5226          JMP LOOP44
0466
0467    0646  7200  END,    CLA
0470    0647  1030          TAD SR40
0471    0650  7104          CLL RAL
-
```

```
0472    0651  3030          DCA SR40
0473    0652  3031          DCA BZ40
0474
0475    0653  5600          JMP I FEEDB4      /END SUBR. 4
0476
0477    0654  0000  INCR34,  0
0500                                          ////////////////
0501    0655  0000  SETL06,  0                //SUBROUTINE 6
0502    0656  1261          TAD SR26          //RESETS SR AND BZ
0503    0657  3020          DCA SR20
0504    0660  7410          SKP
0505    0661  7777  SR26,    7777
0506    0662  1265          TAD SR46
0507    0663  3030          DCA SR40
0510    0664  7410          SKP
0511    0665  0001  SR46,    0001
0512    0666  3021          DCA BZ20
0513    0667  3031          DCA BZ40
0514
0515    0670  5655          JMP I SETL06      /END SUBR. 6
0516                                          ////////////////
0517
0520                /ROM TABLE FOR N=256
0521                /------------------
0522                        *1000             /REAL PART
0523    1000  3777          3777              /0
0524    1001  3777          3777
0525    1002  3776          3776
0526    1003  3772          3772
0527    1004  3766          3766
0530    1005  3761          3761
0531    1006  3752          3752
0532    1007  3742          3742
0533
0534    1010  3731          3731              /10
0535    1011  3716          3716
0536    1012  3703          3703
0537    1013  3666          3666
0540    1014  3650          3650
0541    1015  3631          3631
0542    1016  3610          3610
0543    1017  3567          3567
0544
0545    1020  3544          3544              /20
0546    1021  3520          3520
0547    1022  3473          3473
0550    1023  3445          3445
0551    1024  3416          3416
0552    1025  3366          3366
0553    1026  3335          3335
0554    1027  3302          3302
0555
0556    1030  3247          3247              /30
0557    1031  3212          3212
0560    1032  3155          3155
0561    1033  3117          3117
0562    1034  3057          3057
0563    1035  3017          3017
0564    1036  2755          2755
0565    1037  2713          2713
0566
0567    1040  2650          2650              /40
0570    1041  2604          2604
```

```
0571        1042   2537          2537
0572        1043   2472          2472
0573        1044   2423          2423
0574        1045   2354          2354
0575        1046   2304          2304
0576        1047   2233          2233
0577
0600        1050   2162          2162          /50
0601        1051   2110          2110
0602        1052   2035          2035
0603        1053   1761          1761
0604        1054   1705          1705
0605        1055   1631          1631
0606        1056   1554          1554
0607        1057   1476          1476
0610
0611        1060   1420          1420          /60
0612        1061   1341          1341
0613        1062   1262          1262
0614        1063   1202          1202
0615        1064   1123          1123
0616        1065   1042          1042
0617        1066   0762          0762
0620        1067   0701          0701
0621
0622        1070   0620          0620          /70
0623        1071   0536          0536
0624        1072   0455          0455
0625        1073   0373          0373
0626        1074   0311          0311
0627        1075   0227          0227
0630        1076   0144          0144
0631        1077   0062          0062
0632
0633        1100   0000          0000          /100
0634        1101   7715          7715
0635        1102   7633          7633
0636        1103   7550          7550
0637        1104   7466          7466
0640        1105   7404          7404
0641        1106   7322          7322
0642        1107   7241          7241
0643        1110   7157          7157
0644
0645        1111   7076          7076          /110
0646        1112   7015          7015
0647        1113   6735          6735
0650        1114   6654          6654
0651        1115   6575          6575
0652        1116   6515          6515
0653        1117   6436          6436
0654        1120   6357          6357
0655
0656        1121   6301          6301          /120
0657        1122   6223          6223
0660        1123   6146          6146
0661        1124   6072          6072
0662        1125   6016          6016
0663        1126   5742          5742
0664        1127   5667          5667
0665        1130   5615          5615
0666
0667        1131   5544          5544          /130
-
```

```
0670        1132  5473          5473
0671        1133  5423          5423
0672        1134  5354          5354
0673        1135  5305          5305
0674        1136  5240          5240
0675        1137  5173          5173
0676        1140  5127          5127
0677
0700        1141  5064          5064          /140
0701        1142  5022          5022
0702        1143  4760          4760
0703        1144  4720          4720
0704        1145  4660          4660
0705        1146  4622          4622
0706        1147  4565          4565
0707        1150  4530          4530
0710
0711        1151  4475          4475          /150
0712        1152  4442          4442
0713        1153  4411          4411
0714        1154  4361          4361
0715        1155  4332          4332
0716        1156  4304          4304
0717        1157  4257          4257
0720        1160  4233          4233
0721
0722        1161  4210          4210          /160
0723        1162  4167          4167
0724        1163  4146          4146
0725        1164  4127          4127
0726        1165  4111          4111
0727        1166  4074          4074
0730        1167  4061          4061
0731        1170  4046          4046
0732
0733        1171  4035          4035          /170
0734        1172  4025          4025
0735        1173  4016          4016
0736        1174  4011          4011
0737        1175  4005          4005
0740        1176  4001          4001
0741        1177  4000          4000
0742        1200  4000          4000
0743
0744
0745                            *1400         //IMAG PART
0746        1400  0000          0000          /0
0747        1401  0062          0062
0750        1402  0144          0144
0751        1403  0227          0227
0752        1404  0311          0311
0753        1405  0373          0373
0754        1406  0455          0455
0755        1407  0536          0536
0756
0757        1410  0602          0602          /10
0760        1411  0701          0701
0761        1412  0762          0762
0762        1413  1042          1042
0763        1414  1123          1123
0764        1415  1202          1202
0765        1416  1262          1262
0766        1417  1341          1341
```

```
0767
0770        1420   1420          1423              /20
0771        1421   1476          1476
0772        1422   1554          1554
0773        1423   1631          1631
0774        1424   1705          1705
0775        1425   1761          1761
0776        1426   2035          2035
0777        1427   2110          2110
1000
1001        1430   2162          2162              /30
1002        1431   2233          2233
1003        1432   2304          2304
1004        1433   2354          2354
1005        1434   2423          2423
1006        1435   2472          2472
1007        1436   2537          2537
1010        1437   2604          2604
1011
1012        1440   2650          2650              /40
1013        1441   2713          2713
1014        1442   2755          2755
1015        1443   3017          3017
1016        1444   3057          3057
1017        1445   3117          3117
1020        1446   3155          3155
1021        1447   3212          3212
1022
1023        1450   3247          3247              /50
1024        1451   3302          3302
1025        1452   3335          3335
1026        1453   3366          3366
1027        1454   3416          3416
1030        1455   3445          3445
1031        1456   3473          3473
1032        1457   3520          3520
1033
1034        1460   3544          3544              /60
1035        1461   3567          3567
1036        1462   3610          3610
1037        1463   3631          3631
1040        1464   3650          3650
1041        1465   3666          3666
1042        1466   3703          3703
1043        1467   3716          3716
1044
1045        1470   3731          3731              /70
1046        1471   3742          3742
1047        1472   3752          3752
1050        1473   3761          3761
1051        1474   3766          3766
1052        1475   3772          3772
1053        1476   3776          3776
1054        1477   3777          3777
1055
1056        1500   3777          3777              /100
1057        1501   3777          3777
1060        1502   3776          3776
1061        1503   3772          3772
1062        1504   3766          3766
1063        1505   3761          3761
1064        1506   3752          3752
1065        1507   3742          3742
-
```

```
1066
1067        1510  3731          3731          /110
1070        1511  3716          3716
1071        1512  3703          3703
1072        1513  3666          3666
1073        1514  3650          3650
1074        1515  3631          3631
1075        1516  3610          3610
1076        1517  3567          3567
1077
1100        1520  3544          3544          /120
1101        1521  3520          3520
1102        1522  3473          3473
1103        1523  3445          3445
1104        1524  3416          3416
1105        1525  3366          3366
1106        1526  3335          3335
1107        1527  3302          3302
1110
1111        1530  3247          3247          /130
1112        1531  3212          3212
1113        1532  3155          3155
1114        1533  3117          3117
1115        1534  3057          3057
1116        1535  3017          3017
1117        1536  2755          2755
1120        1537  2713          2713
1121
1122        1540  2650          2650          /140
1123        1541  2604          2604
1124        1542  2537          2537
1125        1543  2472          2472
1126        1544  2423          2423
1127        1545  2354          2354
1130        1546  2304          2304
1131        1547  2233          2233
1132
1133        1550  2162          2162          /150
1134        1551  2110          2110
1135        1552  2035          2035
1136        1553  1761          1761
1137        1554  1705          1705
1140        1555  1631          1631
1141        1556  1554          1554
1142        1557  1476          1476
1143
1144        1560  1420          1420          /160
1145        1561  1341          1341
1146        1562  1262          1262
1147        1563  1202          1202
1150        1564  1123          1123
1151        1565  1042          1042
1152        1566  0762          0762
1153        1567  0701          0701
1154
1155        1570  0620          0620          /170
1156        1571  0536          0536
1157        1572  0455          0455
1160        1573  0373          0373
1161        1574  0311          0311
1162        1575  0227          0227
1163        1576  0144          0144
1164        1577  0062          0062          //END ROM256
-
```

```
1165
1166
1167                            *6000
1170      6000  0000  PROCT7,  0                 ///PROCT7///
1171      6001  7344          CLA CLL CMA RAL
1172      6002  3333          DCA COUNT2        /COUN12=-2
1173      6003  3331          DCA RØI1          /CLEAR RØI1
1174      6004  6046          TLS               /CLEAR PRINT FLAG
1175
1176      6005  7000  SKPLP,  NOP               //SKIPLOOP,TWO RUNS
1177      6006  3332          DCA EØF1          /CLEAR EØF1
1200      6007  1331          TAD RØI1          /1.RUN=Ø, 2.RUN=1
1201      6010  7450          SNA
1202      6011  4615          JMS I SRE         /FIRST RUN
1203      6012  7440          SZA
1204      6013  4617          JMS I SRI         /SECND.RUN
1205      6014  7410          SKP
1206      6015  6244  SRE,    REAL
1207      6016  7410          SKP
1210      6017  6272  SRI,    IMAG
1211      6020  7200          CLA               /STORE R-I-INDEX
1212      6021  4734          JMS I SRCRLF
1213      6022  4771          JMS I SP4
1214
1215      6023  1327          TAD HEADX
1216      6024  3330          DCA HEADIN        /SET HEADINDEX =COL-NR.
1217      6025  3231          DCA COLIN         /CLEAR COLINDEX
1220
1221      6026  4771  HEADLP, JMS I SP4         //LOOP PRINTS COL.HEAD
1222      6027  4770          JMS I SP3
1223      6030  4335          JMS COLNR
1224      6031  0000  COLIN,  Ø
1225      6032  2231          ISZ COLIN         /COLINDEX Ø...7
1226      6033  2330          ISZ HEADIN        /COUNTINDEX -7...Ø
1227      6034  5226          JMP HEADLP
1230
1231      6035  4734          JMS I SRCRLF
1232      6036  4734          JMS I SRCRLF
1233
1234      6037  3251          DCA LNR           /CLEARS LINE NR.
1235
1236      6040  1100          TAD NØ            //RESET NØD3
1237      6041  7415          ASR
1240      6042  0002          2
1241      6043  3325          DCA NØD3
1242
1243      6044  1100  ADRGAP, TAD NØ
1244      6045  7415          ASR
1245      6046  0003          3
1246      6047  3326          DCA NØD16
1247
1250      6050  4343  LNRLP,  JMS LINENR        //LP FOR LINE NR.
1251      6051  0000  LNR,    Ø                 /LINE NR.[TO SR]
1252      6052  2251          ISZ LNR
1253      6053  1327          TAD HEADX
1254      6054  3330          DCA HEADIN        /COUNTINDEX -7...Ø
1255
1256      6055  4771  LPRLP,  JMS I SP4         //LINEPRINT-LOOP
1257      6056  1332          TAD EØF1
1260      6057  7650          SNA CLA
1261      6060  5262          JMP E
1262      6061  5266          JMP F
1263      6062  1331  E,      TAD RØI1          /IF EØF1=Ø
```

```
1264    6063  7650          SNA CLA
1265    6064  5272          JMP RE
1266    6065  5300          JMP IE
1267    6066  1331  F,      TAD R0I1        /IF E0F1=1
1270    6067  7650          SNA CLA
1271    6070  5275          JMP RF
1272    6071  5303          JMP IF
1273    6072  7000  RE,      NOP             /IF R0I1=0
1274    6073  1414          TAD I RE0
1275    6074  5305          JMP PR
1276    6075  7000  RF,      NOP             /IF R0I1=0
1277    6076  1416          TAD I RF0
1300    6077  5305          JMP PR
1301    6100  7000  IE,      NOP             /IF R0I1=1
1302    6101  1415          TAD I IE0
1303    6102  5305          JMP PR
1304    6103  7000  IF,      NOP             /IF R0I1=1
1305    6104  1417          TAD I IF0
1306    6105  4352  PR,      JMS PRAC
1307    6106  2330          ISZ HEADIN
1310    6107  5255          JMP LPRLP
1311    6110  4773          JMS I CRL
1312    6111  2325          ISZ N0D8
1313    6112  7410          SKP
1314    6113  5320          JMP ENDBA
1315    6114  2326          ISZ N0D16
1316    6115  5250          JMP LNRLP
1317    6116  2332          ISZ E0F1
1320    6117  5244          JMP ADRGAP
1321    6120  7402  ENDBA,   HLT             /PAGE HALT
1322    6121  2331          ISZ R0I1
1323    6122  2333          ISZ COUNT2
1324    6123  5205          JMP SKPLP
1325
1326    6124  7402          HLT             /END SR PROCT7
1327                                        ////////////////
1330
1331    6125  0000  N0D8,    0
1332    6126  0000  N0D16,   0
1333    6127  7770  HEADX,   -10
1334    6130  0000  HEADIN,  0
1335    6131  0000  R0I1,    0
1336    6132  0000  E0F1,    0
1337    6133  0000  COUNT2,  0
1340    6134  6234  SRCRLF,  CRLF
1341                        ASR=7415
1342
1343    6135  0000  COLNR,   0               //SR TYPES COL.NUMBER
1344    6136  1735          TAD I COLNR     /HEADIN->AC
1345    6137  1367          TAD K260        /ADD ASCII-MASC
1346    6140  4772          JMS I TYP
1347    6141  2335          ISZ COLNR
1350    6142  5735          JMP I COLNR
1351
1352    6143  0000  LINENR,  0               //SR LINE NR.
1353    6144  1743          TAD I LINENR
1354    6145  7104          CLL RAL
1355    6146  7006          RTL
1356    6147  4352          JMS PRAC
1357    6150  2343          ISZ LINENR
1360    6151  5743          JMP I LINENR
1361
1362
-
```

```
1363    6152  0000  PRAC,   0               //SUBR. PRINT AC
1364    6153  7421          MQL             /AC->MQ,0->AC
1365    6154  1365          TAD X           /LOAD LOOP INDEX
1366    6155  3366          DCA SHFTIN
1367
1370    6156  7413  SHFTLP, SHL             /MQ0...MQ2-BIT ->AC
1371    6157  0002          2
1372    6160  1367          TAD K260
1373    6161  4772          JMS I TYP
1374    6162  2366          ISZ SHFTIN
1375    6163  5356          JMP SHFTLP
1376
1377    6164  5752          JMP I PRAC
1400    6165  7774  X,      -4
1401    6166  0000  SHFTIN, 0
1402                        SHL=7413
1403                        MQL=7421
1404    6167  0260  K260,   0260
1405    6170  6200  SP3,    SPACE3
1406    6171  6213  SP4,    SPACE4
1407    6172  6224  TYP,    TYPE
1410    6173  6234  CRL,    CRLF
1411    6174  6244  REA,    REAL
1412
1413
1414                        *6200
1415    6200  0000  SPACE3, 0               //SEPARATES COLUMNS
1416    6201  1210          TAD X3
1417    6202  3212          DCA I4
1420
1421    6203  1211  LOOP4,  TAD K240
1422    6204  4224          JMS TYPE
1423    6205  2212          ISZ I4
1424    6206  5203          JMP LOOP4
1425
1426    6207  5600          JMP I SPACE3
1427    6210  7775  X3,     -3              /LENGTH OF SPACE
1430    6211  0240  K240,   240             /ASCII FOR 1 SPACE
1431    6212  0000  I4,     0
1432
1433    6213  0000  SPACE4, 0
1434    6214  1223          TAD X4
1435    6215  3212          DCA I4
1436    6216  1211  LOOP40, TAD K240
1437    6217  4224          JMS TYPE
1440    6220  2212          ISZ I4
1441    6221  5216          JMP LOOP40
1442
1443    6222  5613          JMP I SPACE4
1444    6223  7774  X4,     -4
1445
1446
1447    6224  0000  TYPE,   0               //TYPE SUBROUTINE
1450    6225  6041          TSF             /SKIP ON TTY FLAG = 1
1451    6226  5225          JMP .-1
1452    6227  6046          TLS             /PRINT THE CARACTER
1453    6230  6041          TSF
1454    6231  5230          JMP .-1
1455    6232  7300          CLA CLL
1456    6233  5624          JMP I TYPE
1457
1460    6234  0000  CRLF,   0               //CARR.RET./LINE FEED
1461    6235  1242          TAD K215
-
```

```
1462        6236  4224            JMS TYPE
1463        6237  1243            TAD K212
1464        6240  4224            JMS TYPE
1465        6241  5634            JMP I CRLF
1466        6242  0215  K215,     215                 /ASCII FOR CARR.RET.
1467        6243  0212  K212,     212                 /ASCII FOR LINE FEED
1470
1471
1472        6244  0000  REAL,     0
1473        6245  4647            JMS I SSP
1474        6246  7410            SKP
1475        6247  6200  SSP,      SPACE3
1476        6250  1263            TAD CHREAL
1477        6251  3010            DCA 10
1500        6252  1270            TAD M4
1501        6253  3271            DCA MM4
1502        6254  1410  LOOP3,    TAD I 10
1503        6255  4657            JMS I STY
1504        6256  7410            SKP
1505        6257  6224  STY,      TYPE
1506        6260  2271            ISZ MM4
1507        6261  5254            JMP LOOP3
1510        6262  5644            JMP I REAL
1511        6263  6263  CHREAL,   .
1512        6264  0322            322
1513        6265  0305            305
1514        6266  0301            301
1515        6267  0314            314
1516        6270  7774  M4,       -4
1517        6271  0000  MM4,      0
1520
1521        6272  0000  IMAG,     0
1522        6273  4647            JMS I SSP
1523        6274  1305            TAD CHIMAG
1524        6275  3010            DCA 10
1525        6276  1270            TAD M4
1526        6277  3271            DCA MM4
1527        6300  1410  LOOP33,   TAD I 10
1530        6301  4657            JMS I STY
1531        6302  2271            ISZ MM4
1532        6303  5300            JMP LOOP33
1533        6304  5672            JMP I IMAG
1534        6305  6305  CHIMAG,   .
1535        6306  0311            311
1536        6307  0315            315
1537        6310  0301            301
1540        6311  0307            307

NO ERRORS

ADRGAP 6044
ASR    7415
BZ20   0021
BZ40   0031
CHIMAG 6305
CHREAL 6263
CNT1   0423
CNT224 0635
CNT24  0613
CNT334 0641
CNT34  0617
COLIN  6031
COLNR  6135
-
```

```
COUNT2  6133
CRL     6173
CRLF    6234
E       6062
END     0646
ENDBA   6120
END1    0341
E0F1    6132
F       6066
FEEDB4  0600
HEADIN  6130
HEADLP  6026
HEADX   6127
HN0     0101
HN00    0247
IA0     0011
IA1     0343
IA5     0513
IB0     0013
IB1     0345
IB5     0514
IE      6100
IE0     0015
IE5     0517
IF      6103
IF0     0017
IF5     0520
IMAG    6272
INCR34  0654
IPROD   0444
IRPTR   0007
ISTRT2  0425
ITER0   0213
I4      6212
K212    6243
K215    6242
K240    6211
K260    6167
LINENR  6143
LNR     6051
LNRLP   6050
LOOP3   6254
LOOP33  6300
LOOP34  0604
LOOP4   6203
LOOP40  6216
LOOP44  0626
LPRLP   6055
MASC0   0211
MASC2   0426
MMTRX3  0427
MM4     6271
MQA     7501
MQL     7421
MSC     0103
M0      0102
M00     0246
M4      6270
N0      0100
N0D16   6126
N0D8    6125
PR      6105
PRAC    6152
PROCT7  6000
P3      0465
RA0     0010
RA1     0342
RA5     0511
RB0     0012
RB1     0344
RB5     0512
RE      6072
REA     6174
REAL    6244
RE0     0014
RE5     0515
RF      6075
RF0     0016
RF5     0516
ROMP02  0400
RPROD   0432
RRPTR   0006
RSTRT2  0424
R0I1    6131
SEQU0   0215
SEQU00  0231
SETL06  0655
SETP05  0466
SHFTIN  6166
SHFTLP  6156
SHL     7413
SKPLP   6005
SMTRX1  0300
SPACE3  6200
SPACE4  6213
SP3     6170
SP4     6171
SRCRLF  6134
SRE     6015
SRI     6017
SR2     0250
SR20    0020
SR26    0661
SR3     0251
SR4     0252
SR40    0030
SR46    0665
SR5     0253
SR6     0254
SR7     0255
SSP     6247
STY     6257
TYP     6172
TYPE    6224
X       6165
X3      6210
X4      6223
ZWSPIF  0464
ZWSPRF  0463
```

Die Kolonne ganz links besteht aus der fortlaufenden Zeilennummer der Liste. Die nächste Kolonne gibt die jeweilige Adresse an. Die dritte Zahlenkolonne enthält den Binärcode der einzelnen Befehle. Die bis 6-teiligen Namen werden von Kommata abgeschlossen und die Assembler-Mnemonics bestehen wie üblich aus drei Buchstaben. Nach Schrägstrichen / folgen Kommentare. Wo nötig, ist der Befehlsablauf kommentiert, was die Uebersicht auch für den Programmierenden wesentlich erhöht.

Einige Bemerkungen zur Zahldarstellung:
Im LINC-Mode existiert ein Befehl MUL, der 9,6 µsec braucht für eine vollständige Einerkomplement-Multiplikation von Bruchzahlen. Die vergleichbare EAE[2]-Multiplikation MUY im P-Mode verarbeitet nur Integerzahlen ohne Vorzeichen, dies in 9 µsec. Mit Hilfe des EAE-Befehlsschatzes würde man mehrere Befehle brauchen, um MUL nachzuvollziehen. Um ein schnelles Programm zu haben (für Simulation kein wichtiges Argument) und - vor allem - ein möglichst einfaches, entschied ich mich nach einigen Versuchen, MUL zu verwenden. Die hiefür notwendige Bruch-(engl. fractional-) Darstellung im Einerkomplement ist in Fig. 7 erklärt. Zur Zahldarstellung durch Komplemente vergleiche man (24).

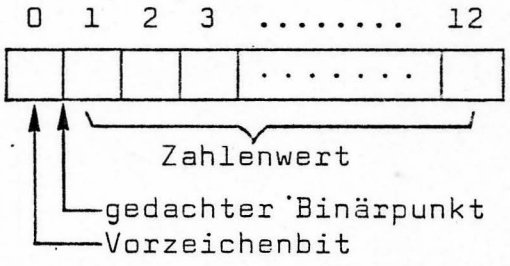


Fig. 7:

oktal		dezimal	
3777	=	+0,999	
	:		
0000	=	+0	} identisch
7777	=	-0	
	:		
4000	=	-0,999	

In einer rekursiven Festkomma-Maschine ist Bruchdarstellung angezeigt, weil nur damit mühelos eine feste Wortlänge im ganzen Rechenablauf eingehalten werden kann. Die Multiplikation zweier b-bit Zahlen ergibt ein 2b-bit Produkt. Zur Einschränkung auf die Wortlänge b lässt man die letzten b Ziffern einfach fallen.

---

[2] EAE= extended arithmetic element, Befehlsrepertoire für umfassende Multiplikations-/Divisions-Subroutinen

Eine solche Approximation lässt sich mit Integer-Zahlen nicht durchführen. Die Entscheidung zugunsten der Bruchdarstellung ist deshalb auch für den Hardware-Prozessor verbindlich. Die Wahl des Einerkomplements ist hier allerdings nur eine Programmierfrage; bei der verdrahteten Maschine sind ausschliesslich Hardware-Rücksichten für die Entscheidung massgebend. In den Kapiteln 2 und 3 gehe ich auf diese Fragen genauer ein.-

Weitere Einzelheiten sind aus dem Programm selber und aus dortigen Kommentaren ersichtlich. Die Bedeutung der Befehle lassen sich in den PDP-12 Handbüchern nachschlagen, insbesondere gibt das Werklein (6) einen kurzen Ein- und Ueberblick. Auf den Seiten 25bis 29 ist die ROM-Tabelle der harmonischen Gewichtsfaktoren ausgedruckt, zu welcher der Graph in Fig. 5 gehört. Diese ROM-Tabelle wurde von Hand als Quellenprogramm eingetippt. Anschliessend folgt ab Seite 30 die überraschend umfangreiche output-Routine PROCT7 für Tabellendarstellung. Der Schluss wird gebildet von einem alphabethischen listing der Namen und EAE-Befehle mit ihren zugehörigen Adressen bzw. Code-Nummern.

Zur Optimierung des Programms habe ich, vor allem aus ästhetischen Gründen, einige Zeit investiert. Die abgebildete beste Version braucht für N=256 0,36 sec., für N=512 0,81 sec Rechenzeit. Der Vergleich zum festverdrahteten FFT-Prozessor ist interessant: 0,0015 sec für N=256, wie ich in (14) abgeschätzt habe, also gut 200-mal schneller als sein Simulationsprogramm. Punkto Rechenzeit kann dieses aber trotzdem als sehr gutes FFT-Programm angesehen werden. Denn die allgemein anerkannte PDP-12 Fast-Fourier-Routine braucht umgerechnet für 256 komplexe Punkte (≙ 512 reelle Pkte) 1,33 sec. Selbst wenn darin mehr als eine rohe FFT-Prozedur enthalten ist, ist mein Simulationsprogramm wesentlich schneller. Der Matrix-Algorithmus von Pease ist mithin auch softwaremässig beachtenswert, ein Resultat, das vielleicht Gegenstand einer kleineren Publikation sein könnte.

1.4. Einige Simulationsergebnisse

Zur praktischen Erprobung des Simulationsprogrammes stellte sich zunächst das Problem, den ROM-Inhalt und den Eingangsvektor, beides auf LINC-Tape gespeichert, an den vorgesehenen Plätzen (s. Fig. 4) in den Kernspeicher zu lesen. Ausserdem war der Resultatvektor zurück auf's Band zu befördern, wo er mit LOOKPLOT inspiziert werden konnte. Drei kleine Assembler-Routinen im L-Mode mussten hiefür entwickelt werden und sind unter den Namen INROM, INBATCHF und OUTAPE anschliessend dargestellt:

```
0000                    *20
0001                    /********** INROM **************
0002
0003                    /PROGRAM TRANSFERS BLOCKS NR 110 AND
0004                    /111 FROM LINC-TAPE UNIT 1
0005                    /TO INBATCH REGISTERS 1000-1777
0006
0007                        PMODE
0010                        *7030
0011    7030  6141          LINC
0012                        LMODE
0013    1031  0640          LDF 0           /SET DATA FIELD 0
0014    1032  0710          RDC U           /READ AND CHECK 1 TBLK
0015    1033  6110          6110            /MBLK 6 AND TBLK 110
0016    1034  0710          RDC U
0017    1035  7111          7111            /MBLK 7 AND TBLK 111
0020                /MBLK 6 = 1000-1377 IF DF=0
0021                /MBLK 7 = 1400-1777 IF DF=0
0022    1036  0002          PDP
0023                        PMODE
0024    7037  7402          HLT
```

```
0000                    *20
0001                    /********** INBATCHF **************
0002
0003                    /PROGRAM TRANSFERS BLOCKS NR 100 - 103
0004                    /FROM LINC-TAPE UNIT 1
0005                    /TO ALL INBATCH REGISTERS 2000 - 3777
0006
0007                    /PROGRAM F CONTAINS PROGRAMS 1 AND 2 *** !
0010
0011                            PMODE
0012                            *7000
0013        7000  6141          LINC
0014                            LMODE
0015        1001  0641          LDF 1               /SET DATA FIELD 1
0016        1002  0710          RDC U               /READ AND CHECK 1 TBLK
0017        1003  4100          4100                /MBLK 4 AND TBLK 100
0020        1004  0710          RDC U
0021        1005  5101          5101
0022        1006  0710          RDC U
0023        1007  6102          6102
0024        1010  0710          RDC U
0025        1011  7103          7103
0026                    /MBLK 4 = 2000-2377
0027                    /MBLK 5 = 2400-2777
0030                    /MBLK 6 = 3000-3377
0031                    /MBLK 7 = 3400-3777
0032        1012  0002          PDP
0033                            PMODE -
0034        7013  7402          HLT
```

```
0000                    *20
0001                    /********* OUTAPE ************
0002
0003                    /PROGRAM TRANSFERS FULL OUTBATCH
0004                    /TO LINC-TAPE UNIT 1
0005                    /RE -> TBLK 120, RF -> TBLK 121
0006                    /IE -> TBLK 122, IF -> TBLK 123
0007
0010                            PMODE
0011                            *7100
0012        7100  6141          LINC
0013                            LMODE
0014        1101  0642          LDF 2               /SET DATA FIELD 2
0015                    /MBLK 4 = 4000-4377
0016                    /MBLK 5 = 4400-4777
0017                    /MBLK 6 = 5000-5377
0020                    /MBLK 7 = 5400-5777
0021        1102  0714          WRC U               /WRITE AND CHECK 1 TBLK
0022        1103  4120          4120                /MBLK 4 AND TBLK 120
0023        1104  0714          WRC U
0024        1105  5121          5121
0025        1106  0714          WRC U
0026        1107  6122          6122
0027        1110  0714          WRC U
0030        1111  7123          7123
0031        1112  0002          PDP
0032                            PMODE
0033        7113  7402          HLT    -
```

Die Programme zur Erzeugung des ROM-Inhalts für 256 bzw. 512 Punkte und zur Synthese der Testvektoren habe ich in FOCAL-12 abgefasst (6). FOCAL-12 ist eine hochinteraktive, benützerorientierte Befehlssprache, die, speziell für den PDP-12 entwickelt, ~~eine~~ eine erstaunliche Leistungsfähigkeit besitzt. Sie kann mit Assemblerroutinen verknüpft werden und beeinflusst deshalb auch periphere Einheiten (LINC-tape, scope, relay registers, A/D-channels). Sie verfügt über einen Satz wichtiger Funktionsprozeduren, was sie zur Erzeugung künstlicher Datensegmente und -fenster hervorragend geeignet macht.

Für N=512 erzeugt das Programm ROMFO512 den ROM-Inhalt (Fig. 8) und speichert ihn in den beiden Bandblöcken 110 und 111, die je 256 Punkte umfassen. Im Gegensatz zur Fig. 5, die ein im Quellenprogramm von Hand eingetipptes ROM für N=256 darstellt, ist dieses ROM im Quellenprogramm nicht sichtbar. 512 Punkte von Hand zu programmieren ist mühsamer, als die Alternative über FOCAL und Band.

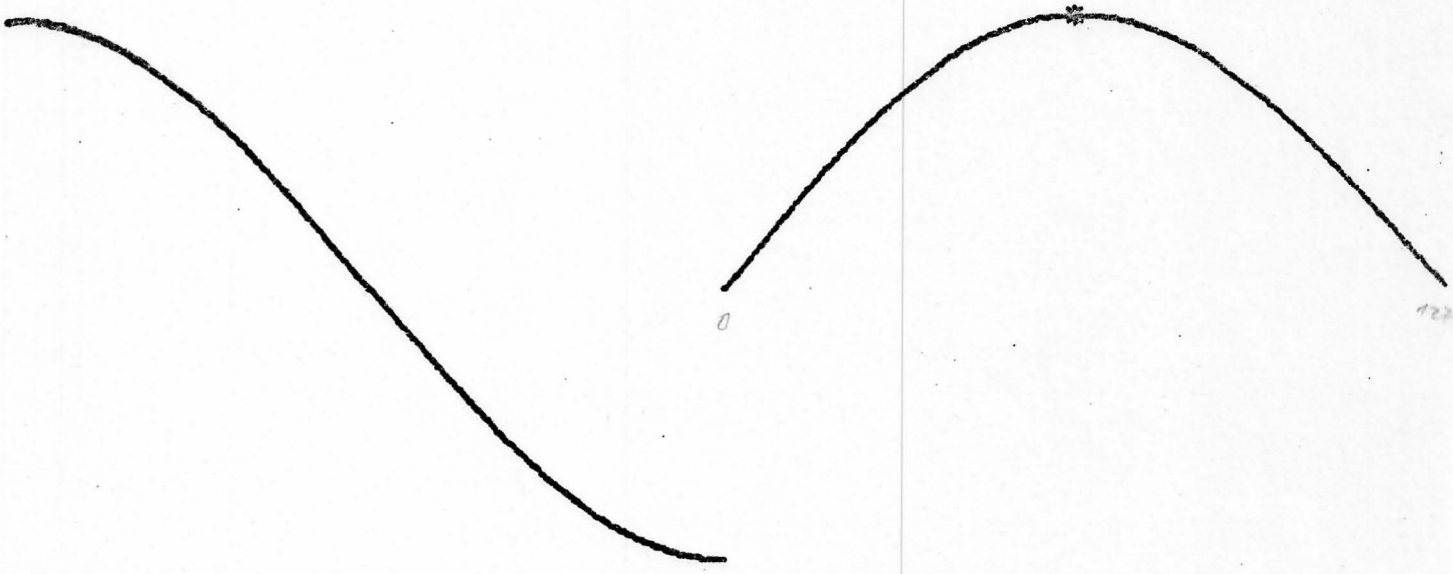
Ab Seite 41 ff folgen für N=512 einige Simulationsbeispiele, und zwar in der Anordnung:

1. Seite: FOCAL-12-Programm und Kommentar
2. Seite: Eingangsvektor Realteil (Block 100 + 101)
   " Imaginärteil (Block 102 + 103)
   Ausgangsvektor Realteil (Block 120 + 121)
   " Imaginärteil (Block 122 + 123)

Gegebenenfalls sind die Graphen mit Zahlenblöcken versehen, welche die Folge der Extremwerte (Spektrallinien) enthält; oder es schliesst eine vollständige alle Punkte enthaltende Tabelle an, die mit der Subroutine PROCT7 erzeugt wurde. Sterne * in den Graphen bezeichnen einzelne Punkte, deren Koordinaten (Anzahl Punkte von links / Ordinate in Fractional-Darstellung) dabeistehen. Alle Zahlen sind oktal.

```
*L L.ROMF0512.10
*W
C FOCAL-12

01.10 C ********** ROMF0512 ************
01.11 C
01.12 C DIESES PROGRAMM ERZEUGT DEN ROM INHALT FUER N=512
01.13 C UND SCHREIBT IHN IN BLOCK NR 110 UND 111 VON UNIT 1
01.20 L 0,F0,I,@110,1
01.30 S PI=3.1415926
01.40 F I=0,255;S F0(I)=2047*FCOS(2*PI*I/512)
01.50 L 0,F1,I,@111,1
01.60 F I=0,255;S F1(I)=2047*FSIN(2*PI*I/512)
01.70 L C,F0
01.80 L C,F1
*
```

BLOCK 0110 **BLOCK 0111**



200

3777

Fig. 8 ROM-Inhalt für N=512. Die beiden Band-Blöcke wurden mit Hilfe von LOOKPLOT auf den Tektronix-Bildschirm gezeichnet und mittels einer hard-copy unit auf Thermopapier gebannt. Die Strichschärfe ist leider nicht zu verbessern. Von den nebenstehenden Oktalzahlen bedeutet die obere die x-Position des Sterns * in Anzahl Punkten vom linken Blockrand (Beginn Sinusbogen), die untere die y-Position in Fractional-Darstellung (3777≙ +0,999...).

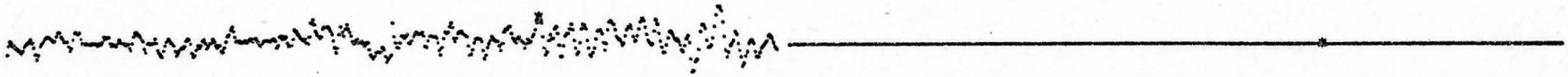
Das Beispiel Fig. 9 ist ein EEG-Stück mit geringer Aussteuerung (Amplitudenmaximum * = 0540 = 17 % des vollen Dynamikbereichs 0 - 3777). Der EEG-Vektor sitzt im Realteil des Eingangs, währenddem der imaginäre Eingang mit Nullen gefüllt ist (obere Zeile). Ausgangsseitig sieht man schön die in diesem Fall zu erwartenden Symmetrien:

- Der Realteil ist eine symmetrische Funktion (wegen der im Falle von DFT zulässigen periodischen Fortsetzung denke man sich Block 121 vor Block 120). Der erste Punkt von Block 120 ist dann die Spektrallinie mit Index 0 = DC-Pegel. Der DC-Pegel ist ungleich Null und positiv. Die Amplitude der Spektrallinien sind der geringen Aussteuerung wegen klein.

- Der Imaginärteil ist (bei derselben gedachten periodischen Fortsetzung mit Index Null zu Beginn von Block 122) eine antisymmetrische Funktion. Der Partner des unter dem Stern \ verborgenen Punkts zeigt deutlich nach unten (ganz rechts). Die Spektrallinie mit Index Null ist gleich Null.

Auf den an Fig. 9 anschliessenden beiden Seiten steht der zu Fig. 9 gehörige Teletype-Output, der mit der Subroutine PROCT7 erzeugt wurde. Alle Symmetrien werden - bis auf Rundungs- bzw. Quantisierungsfehler - bestätigt. Insbesondere ist der DC-Pegel RE (A(0)) = 0027 und der korrespondierende Wert IM (A(0)) = 0000. Die Sterne RE (A(65)) = 0011 und IM (A(63)) = 0030 haben in der Tabelle dieselben Werte. Die Quantisierungsfehler, die durch die iterative Festkomma-Multiplikation mit 12 bit-Operanden entstehen, haben einen Maximalbetrag von 0006. So hoch ist der Fehler allerdings nur bei einzelnen wenigen Punkten. Im übrigen werden höchstens die beiden letzten Stellen affiziert. Die Spektralamplituden sind relativ klein, entsprechend der geringen Aussteuerung im Eingang. Im Interesse eines grossen Geräuschabstandes sollten eingangsseitig höhere Amplituden verwendet werden.

BLOCK 0100 BLOCK 0101 BLOCK 0102 BLOCK 0103



144

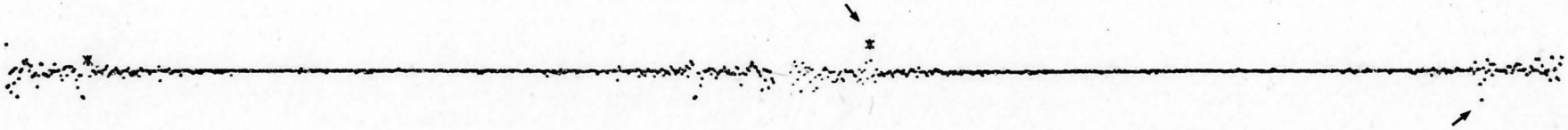
0540

BLOCK 0120 BLOCK 0121

144

0000

BLOCK 0122 BLOCK 0123



065

0011

063

0030

REAL

	0	1	2	3	4	5	6	7
0000	0027	7761	7762	7775	7754	0000	7764	0002
0010	7750	7770	7776	0001	0010	0006	0011	0006
0020	7777	0004	7772	7773	7775	7761	7773	0006
0030	0005	7774	7774	7773	7772	0000	7774	7777
0040	0005	7776	0000	0000	7773	7767	0001	0003
0050	7773	0003	0003	0003	7776	0001	7767	7760
0060	7773	7773	7750	7747	0007	0011	0005	7772
0070	0005	7777	7776	7774	0003	7776	7776	7774
0100	0000	0001	0004	7772	7775	7774	7777	7774
0110	0002	7777	7775	7774	7776	7771	7777	7777
0120	7777	0002	7775	7773	7776	7774	0000	7772
0130	0002	7777	7777	7777	0000	7777	7774	7777
0140	7777	7776	7776	7777	0003	7766	0000	7777
0150	7777	7776	7777	0001	7775	7775	7776	0000
0160	7776	7776	7777	7773	0000	0000	0000	7777
0170	7777	7776	7776	7775	7775	7777	7776	7775
0200	0000	7775	7777	0000	7777	7777	7777	7777
0210	7776	0000	7775	7777	7777	7775	7777	7775
0220	7777	7776	7777	7776	0000	7773	7777	7777
0230	7776	7776	7777	7776	0000	7777	7776	7777
0240	7776	7776	0000	7776	7777	7776	7776	7776
0250	0000	7777	7776	7777	7777	7776	7777	7776
0260	0000	7777	7776	7777	7776	7776	7777	7776
0270	0000	0000	0000	0000	7777	7777	7776	7776
0300	7777	7777	7776	7775	0001	7777	7776	7776
0310	7777	7777	7777	7777	7776	7776	7777	7776
0320	0000	7777	7777	7776	7777	7775	7777	7777
0330	7777	7776	7777	7776	7777	7777	7776	7776
0340	0000	7776	0000	7776	7777	7776	0000	7777
0350	0000	7775	7777	7776	0000	7777	7777	7775
0360	7776	7775	0000	7775	0000	7777	7777	7776
0370	7777	7776	0000	7776	7777	7777	7777	7776
0400	0000	7776	7777	7776	7777	7777	0001	7777
0410	7777	7777	7777	7776	7777	7777	0000	7777
0420	0000	7777	7777	7776	0000	7776	0000	7777
0430	0000	7777	7777	7776	7777	7777	7777	7776
0440	0000	7777	7777	7777	0000	7776	0000	7776
0450	7776	7776	0000	7776	7777	7776	0000	7777
0460	0000	7776	7777	7776	0000	7776	0000	7777
0470	0000	7776	7777	7777	0000	7775	7777	7775
0500	7777	7776	7777	7777	0000	7777	7776	7777
0510	7777	7776	7776	7777	7777	7776	0000	7776
0520	0000	7777	0000	7776	7777	7777	7777	7777
0530	7777	7776	7776	7776	7777	7776	0001	7776
0540	7776	7777	7777	7776	0000	7777	7777	7776
0550	7776	7775	0000	7772	0000	7776	7777	7775
0560	7777	7775	7776	7774	0001	7777	7775	0000
0570	7776	7777	7777	7776	0000	0000	7777	7774
0600	7777	7776	7776	7777	7776	7776	7776	7776
0610	7777	7777	0000	0000	0000	7774	7776	7776
0620	7776	7777	7776	7777	7775	0000	0000	7776
0630	7777	7777	0000	7767	0003	0000	7777	7776
0640	7777	7777	7775	7777	0000	7777	7777	7777
0650	0001	7772	7777	7774	0000	7773	7776	0001
0660	7777	7776	7777	7772	7777	7775	7776	7777
0670	0001	7773	7777	7775	7776	7774	0003	0001
0700	0030	7774	7777	7776	0002	7774	7777	7777
0710	0004	7772	0002	0010	0005	7747	7750	7775
0720	7775	7762	7770	7777	7776	0000	0000	0002
0730	7774	0001	0000	7771	7774	7776	7777	7777
0740	0003	7777	7775	7777	7774	7775	7775	7774
0750	0003	0002	7774	7763	7775	7774	7774	0002
0760	7777	0004	0037	0004	0003	0000	7776	7773

I.AG

	0	1	2	3	4	5	6	7
0000	0000	7754	7764	0011	0005	7770	7777	7756
0010	0010	7774	0004	0000	0004	7766	0002	7756
0020	7772	7761	7774	7774	7770	0002	0004	0002
0030	7772	0000	7776	7766	0000	7766	7770	7774
0040	0000	7771	7767	0002	7777	7777	0000	0001
0050	0002	7774	7773	7766	0002	7770	7776	7764
0060	0001	0010	7771	0030	0012	7773	7774	7766
0070	0002	7774	7773	7776	7775	7773	7775	7775
0100	0000	7777	7774	7772	7776	7777	0001	0000
0110	7777	7775	0001	7775	7773	7774	7777	0002
0120	0002	7777	7777	7772	0003	7776	7776	7777
0130	0000	7775	7773	7777	7777	7774	7777	0002
0140	7777	0001	7776	0000	7775	7775	7776	7773
0150	7775	7777	7776	7774	7777	7775	7776	7774
0160	7777	7774	7777	7776	7776	7776	7777	7775
0170	7776	7774	7776	7775	7777	0000	7775	7773
0200	7777	7776	7777	7775	7776	7776	7776	7776
0210	7777	7776	7776	7776	7776	7776	7776	7777
0220	0000	7776	7776	7776	7776	7776	7776	7776
0230	7776	7775	7776	7776	7777	7777	7776	7774
0240	7776	7775	7776	7775	7777	7777	7776	7776
0250	7777	7775	7777	7776	7776	7777	7776	7776
0260	7776	7776	7776	7776	7776	7776	7777	7776
0270	7776	7776	7777	7777	7776	7777	7776	7777
0300	7777	7776	7776	7777	7776	7776	7777	7776
0310	7777	7776	7777	7776	7776	7776	7775	7776
0320	7776	7776	7776	7776	7776	7776	7777	7777
0330	7776	7777	7777	7776	7776	7777	7776	7776
0340	7776	7776	7776	7776	7776	7776	7776	7776
0350	7776	0000	7776	0000	7776	7776	7775	7777
0360	7776	7777	7777	7775	7777	7777	7776	7776
0370	7777	7776	7776	7776	7777	7776	7776	7776
0400	7777	7777	7777	7776	0000	7777	7776	7777
0410	7777	7777	7777	7777	7777	7777	7777	7777
0420	7777	0000	7777	7777	7777	7777	7777	7777
0430	7777	7777	7777	7777	7777	7777	7777	7777
0440	7777	7776	7776	7777	7776	0000	7777	7776
0450	7777	7777	0000	0001	7777	7777	7777	7777
0460	7776	7777	0000	7777	7777	0000	7777	7777
0470	7777	7777	7776	7777	7776	7776	7776	7776
0500	7777	7776	0001	7777	7777	7776	7776	7777
0510	7776	0000	7776	0000	0000	7777	0000	7777
0520	7777	7776	7776	7777	0000	7777	7777	0000
0530	7777	7776	0000	7776	7776	7777	7776	7777
0540	0000	0000	7777	7777	7776	7776	7777	0000
0550	0000	7776	7777	7777	0000	0000	7777	7777
0560	7776	7775	7776	7777	7777	0001	0000	0000
0570	7775	7775	7777	7776	0000	7777	7776	7776
0600	7776	0001	0000	7776	7777	7777	7777	7777
0610	7776	7777	7777	7777	7777	7777	7777	0000
0620	7775	7777	7777	7777	7777	7777	7777	7777
0630	7777	0000	7777	7777	0000	7776	7777	7775
0640	7775	7774	7777	0000	7776	7776	0001	7777
0650	7776	7776	0000	7777	7773	0002	7777	7777
0660	7775	7775	7777	7777	0001	7777	7774	7777
0670	7777	7775	7776	7776	7777	0002	0001	7776
0700	7776	7777	7777	0000	7777	7777	0002	7777
0710	7774	0007	0000	0001	7763	7745	0002	7767
0720	7775	0011	7777	0005	7773	0005	0000	0000
0730	7777	7776	7776	7777	7777	7774	0003	0003
0740	7775	7777	0003	0005	7777	0005	7777	7777
0750	0001	7773	7773	7775	0003	7777	0000	0012
0760	7777	0012	7774	0004	7774	7776	7773	7777
0770	7766	0013	7777	0001	7770	7767	0005	0012

```
*U
C FOCAL-12

01.10 C ********** DREIECK **********
01.11 C
01.12 C DIESE PROGRAMM ERZEUGT EINEN KOHAERENTEN
01.13 C DREIECK IN UNIT 1 NR 102-103 [IMAGINAER!]
01.14 C REAL INBATCH NR 100-101 SIND GLEICH NULL
01.20 L 0,F0,I,$100.1
01.30 F I=0,15;S A(I)=I-8
01.31 F I=16,31;S A(I)=24-I
01.40 F I=0,15;F K=0,31;S F0(32*I+K+512)=A(K)*200
01.50 F I=0,511;S F0(I)=0

02.10 L C,F0
*
```

Obiges FOCAL-Programm generiert im Imaginärteil des Eingangs eine sog. "kohärente" oder besser harmonische Dreieckschwingung, deren Grundfrequenz ein ganzzahliges Vielfaches n_o der reziproken Fensterläufe 1/T ist. In diesem Fall ist n_o=16 (= 20 oktal). Tatsächlich erscheint beim Punkt Im(A(20)) die Grundwelle 6564, was dem Amplitudenbetrag 1213 entspricht. Wie es beim Dreieck sein muss, erscheinen nur die ungeradzahligen Harmonischen, nämlich A(60), A(120), A(160),... (immer oktale Indices!) Die Summe des Amplitudenbetrags ist

$$1213 + 114 + 35 + 21 + 13 + 10 + 7 + 10$$
$$+ 1205 + 110 + 33 + 16 + 12 + 10 + 7 + 6 = 3074$$

das bis auf den Quantisierungsfehler der Maximalamplitude 3100 im Zeitbereich entspricht. Die Spektrallinien sind korrekterweise negativ; die beiden ersten stehen im Verhältnis 8,56 : 1 bzw. 8,95 : 1, theoretisch ist es 9 : 1. Der Fehler ist also 4,8 % bzw. 0,6 %.

Man beachte das Faktum, dass das Dreieck in die imaginäre Hälfte geschoben wurde. Hiefür gilt im Frequenzbereich: Realteil ungerade, Imaginärteil gerade. Bei allen Beispielen ausser Dreieck ist das umgekehrt.

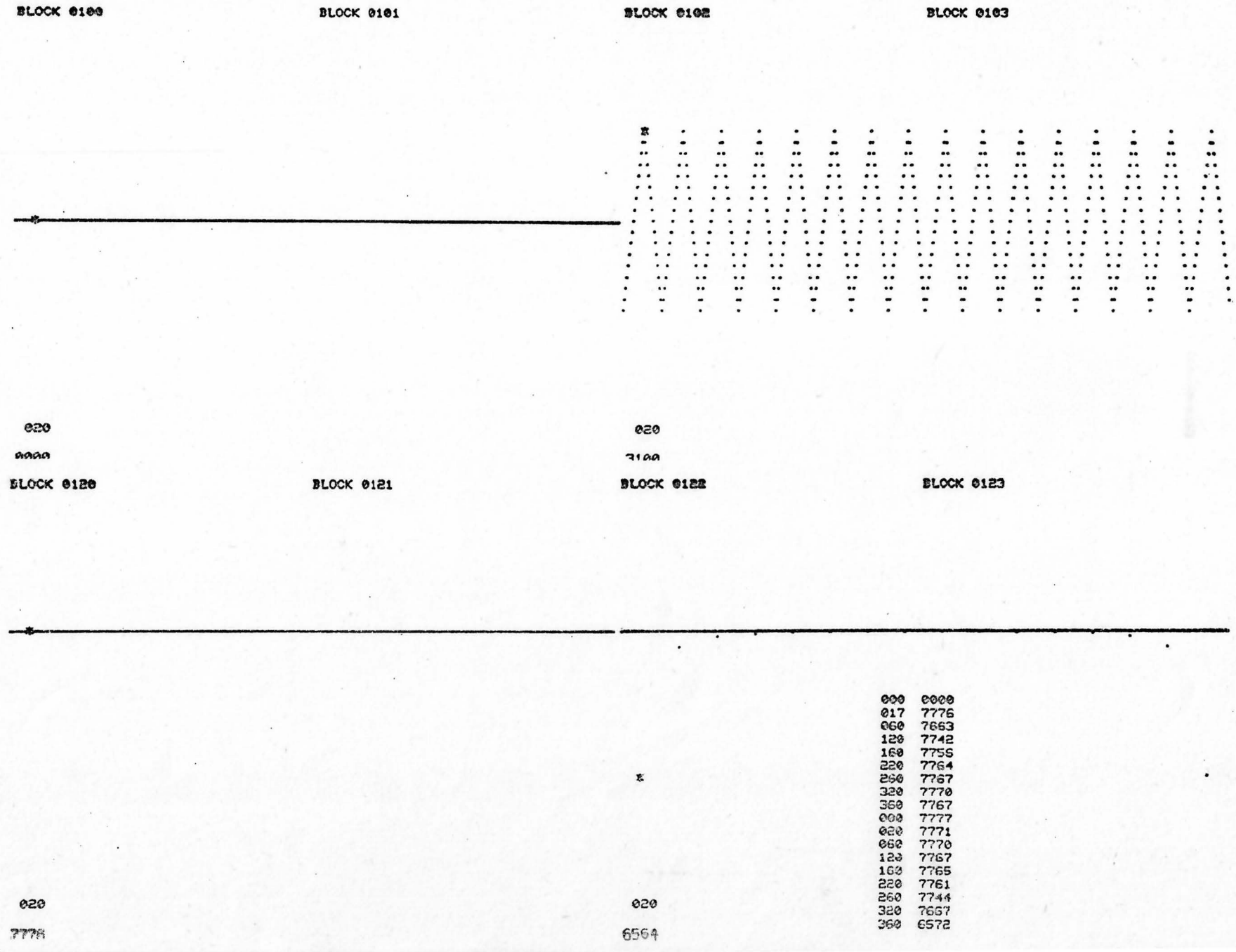

BLOCK 0100
BLOCK 0101
BLOCK 0102
BLOCK 0103
020
0000
020
3100
BLOCK 0120
BLOCK 0121
BLOCK 0122
BLOCK 0123
020
7776
020
6564
000 0000
017 7776
060 7663
120 7742
160 7755
220 7764
260 7767
320 7770
360 7767
000 7777
020 7771
060 7770
120 7767
160 7765
220 7761
260 7744
320 7667
360 6572

Fig. 10

```
*L L.INKODREI.10
*J
C FOCAL-12

01.10 C ********** INKODREI ************
01.11 C
01.12 C DIESE PROGRAMM ERZEUGT EINEN INKOHAERENTEN
01.13 C DREIECK IN UNIT 1 NR 102-103 [IMAGINAER!]
01.14 C REAL INBATCH NR 100-101 SIND GLEICH NULL
01.20 L 0.F0,I,$100.1
01.30 F I=0,17;S A(I)=I-9
01.31 F I=18,35;S A(I)=27-I
01.40 F I=0,15;F K=0,35;S F0(36*I+K+512)=A(K)*200
01.50 F I=0,511;S F0(I)=0

02.10 L C.F0
*
```

Hier (Fig. 11) ist der Dreieck "inkohärent" bzw. anharmonisch, es steht keine ganze Periodenzahl im Zeitfenster: n_o=14,22. Es ist also zu erwarten, dass die Spektrallinien A(16) und A(17) bzw. ihre Spiegelbilder betragmässig besonders hervorstechen. Dies wird bestätigt! Obwohl der Dreieck noch höhere Amplitude hat als in Fig. 10, ist der Betrag von Im (A(16))= 1010, also eindeutig kleiner als im obigen harmonischen Fall (1213). Dies entsteht durch die Verteilung der Signalleistung auf die Seitenlinien und ist, wie im nächsten Kapitel gezeigt ist, letztlich eine Folge des endlichen Datenfensters (leakage, picked-fence effect). Die Seitenlinien fallen schneller ab, wenn entweder im Zeitbereich mit einer geeigneten Fensterfunktion multipliziert wird, oder aber im Frequenzbereich geglättet wird (z.B. Hanning). Auch eine spektrale Mittelung sukzessiver Zeitepochen kommt in Betracht.

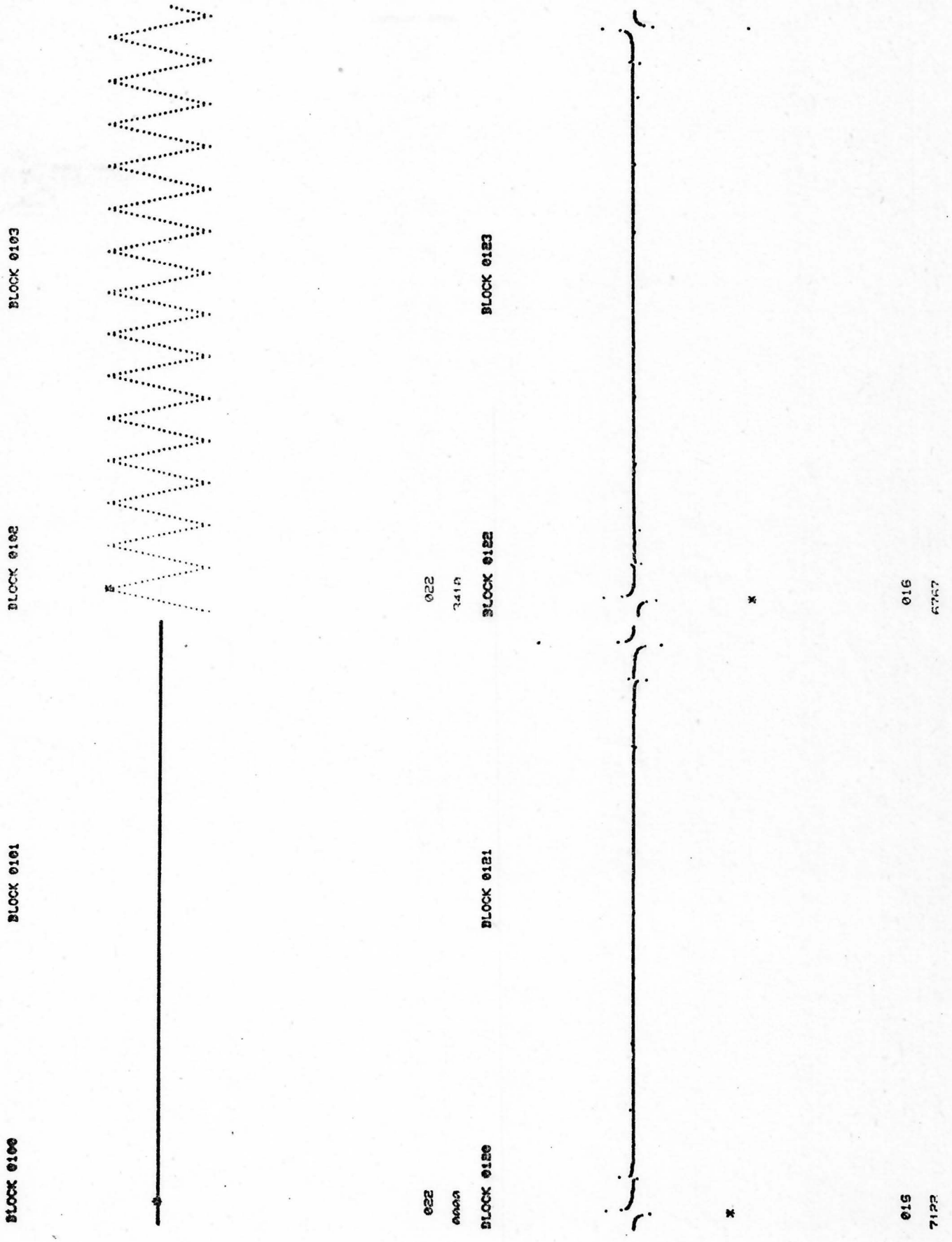

BLOCK 0100
BLOCK 0101
BLOCK 0102
BLOCK 0103
022
0000
BLOCK 0120
BLOCK 0121
BLOCK 0122
BLOCK 0123
016
7122

Fig. 11

```
U
C FOCAL-12

01.10 C ********** INCOSIN **************
01.20 C
01.30 C DIESES PROGRAMM ERZEUGT EINEN INKOHAERENTEN
01.31 C SINUS UND SCHREIBT IHN IN BLOCK NR 100 UND
01.32 C 101 VON UNIT 1. NR 102 UND 103 SIND GLEICH NULL
01.40 L O.F0,I,$100,1
01.50 S PI=3.1415926
01.60 F I=0.511;S F0(I)=2047*FSIN(PI*I/9)
01.70 F I=512,1023;S F0(I)=0
01.80 L C.F0
*
```

Ein anharmonischer Sinus wird untersucht. $n_o = 28{,}4$, also ist die 28. (dezimal) bzw. 34. (oktal) Spektrallinie besonders angeregt. Das Signal erzeugt erwartungsgemäss Reaktionen im Real- und Imaginärteil. Die Energie ist infolge "truncation" verschmiert.

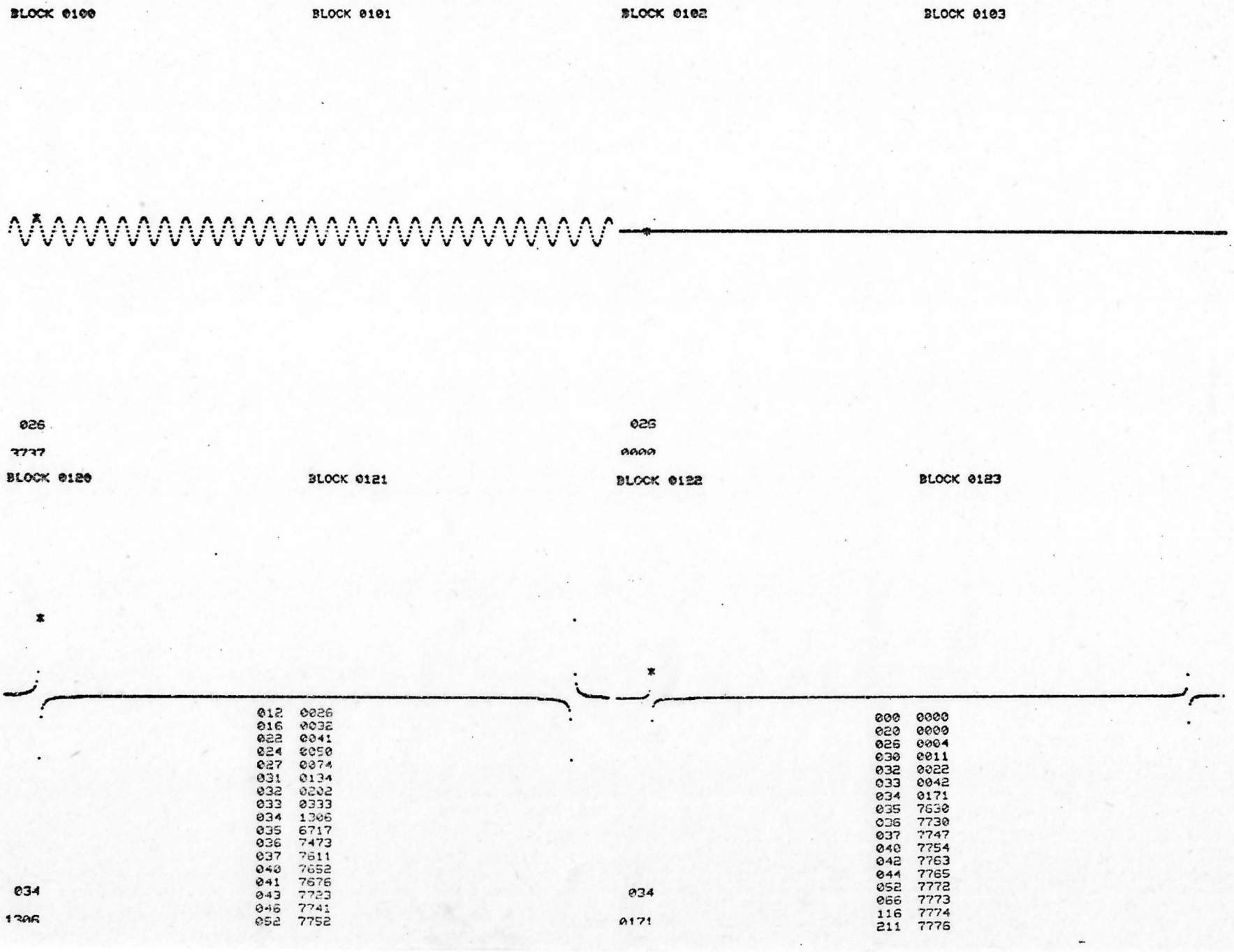

BLOCK 0100
BLOCK 0101
BLOCK 0102
BLOCK 0103
026
3737
BLOCK 0120
BLOCK 0121
026
0000
BLOCK 0122
BLOCK 0123
012 0026
016 0032
022 0041
024 0050
027 0074
031 0134
032 0202
033 0333
034 1306
035 6717
036 7473
037 7611
040 7652
041 7676
043 7723
046 7741
052 7752
034
1306
034
0171
000 0000
020 0000
026 0004
030 0011
032 0022
033 0042
034 0171
035 7630
036 7730
037 7747
040 7754
042 7763
044 7765
052 7772
066 7773
116 7774
211 7776

Fig. 12

```
*L L.KOSAEGE.10
*U
C FOCAL-12

01.10 C ********** KOSAEGE ************
01.11 C
01.12 C PROGRAMM ERZEUGT EINEN KOHAERENTEN SAEGE-
01.13 C ZAHN AUF UNIT 1 NR 100-101
01.14 C NR 102-103 SIND GLEICH NULL
01.20 L O,F0,I,$100,1
01.30 F I=0,15;F K=0,31;S F0(32*I+K)=(K-15)*100
01.32 F I=512,1023;S F0(I)=0
01.40 L C,F0
*
```

Der harmonische Sägezahn (Fig. 13) unterscheidet sich grundsätzlich vom Dreieck durch seine Sprungstellen. Mit $n_o = 16_{10}$ wird IM (A(20)) zur Grundlinie. Alle harmonischen "Obertöne" sind vorhanden mit absteigender Amplitude gemäss der Folge 1, 1/2, 1/3, 1/4, ... Es sind Sinusschwingungen, was stimmt. Im Realteil ist ein kleiner DC-Pegel zu bemerken: RE(A(0)) = 0062. Diese Unsymmetrie wird durch Auszählen verifiziert. Ferner gibt es im Realteil eine auffällige, nicht abklingende Impulsfolge, eine Art Delta-Kamm. Dessen Urbild im Zeitbereich ist ebenfalls ein Delta-Kamm und zwar mit derselben Periode wie der Sägezahn. Es ist offenbar für DFT charakteristisch, Sprungstellen in dieser Weise zu transformieren. Es sieht aus wie eine Kompensation für ein Ueberschwingen in der Art des Gibbs-Phänomens!

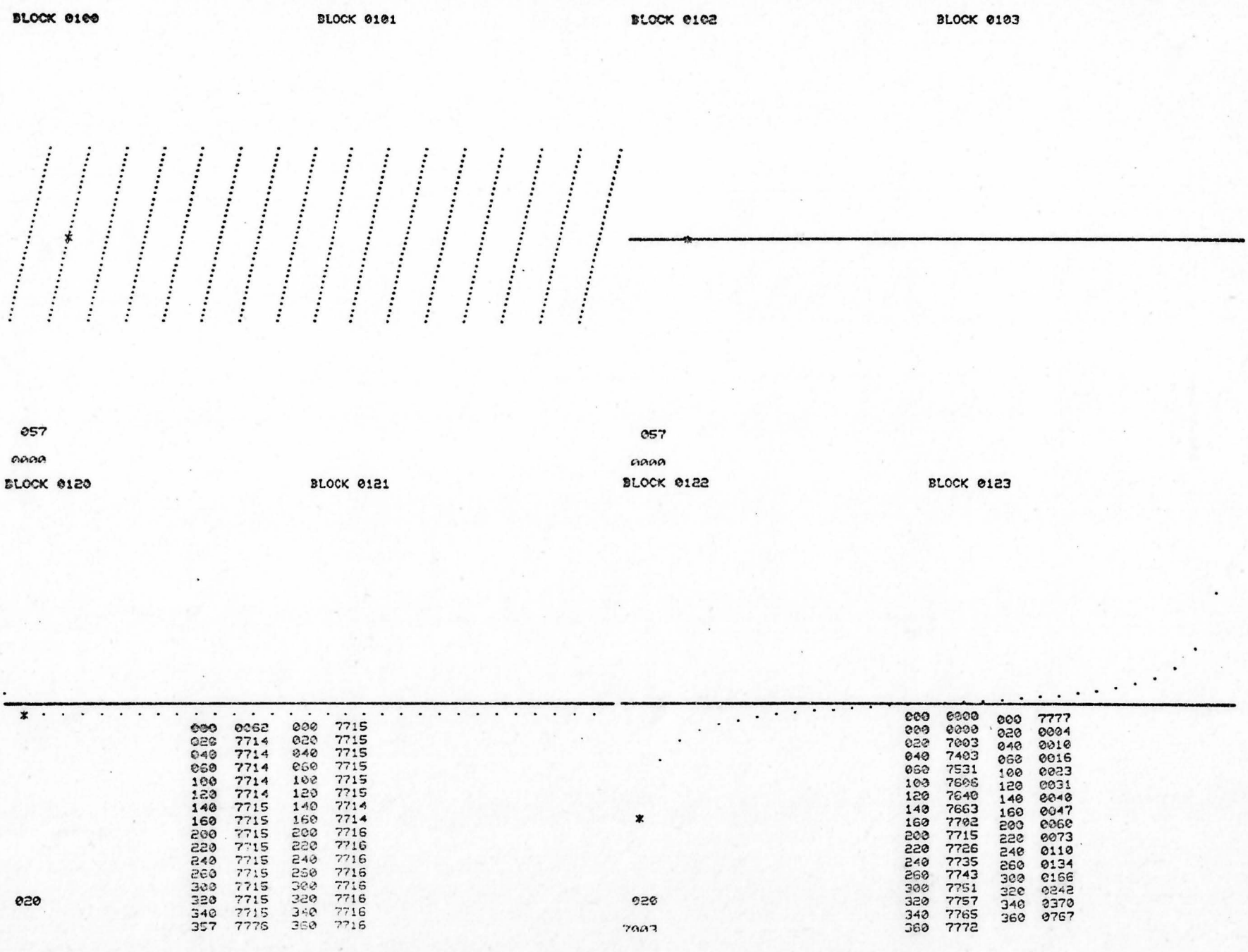

BLOCK 0100
BLOCK 0101
BLOCK 0102
BLOCK 0103
057
0000
BLOCK 0120
BLOCK 0121
057
0000
BLOCK 0122
BLOCK 0123
020
7003
000 0062 000 7715
020 7714 020 7715
040 7714 040 7715
060 7714 060 7715
100 7714 100 7715
120 7714 120 7715
140 7715 140 7714
160 7715 160 7714
200 7715 200 7716
220 7715 220 7716
240 7715 240 7716
260 7715 260 7716
300 7715 300 7716
320 7715 320 7716
340 7715 340 7716
357 7776 360 7716
000 0000 000 7777
000 0000 020 0004
020 7003 040 0010
040 7403 060 0016
060 7531 100 0023
100 7606 120 0031
120 7640 140 0040
140 7663 160 0047
160 7702 200 0060
200 7715 220 0073
220 7726 240 0110
240 7735 260 0134
260 7743 300 0166
300 7751 320 0242
320 7757 340 0370
340 7765 360 0767
360 7772

Fig. 13

```
*W
C FOCAL-12

01.10 C ********** KORECHT **********
01.11 C
01.12 C DIESES PROGRAMM ERZEUGT EINEN KOHAERENTEN
01.13 C RECHTECK AUF UNIT 1 NR 100-101
01.14 C NR 102-103 SIND GLEICH NULL
01.20 L 0.F0,I,$100,1
01.30 F I=0,15;S A(I)=1000
01.31 F I=16,31;S A(I)=-1000
01.40 F I=0,15;F K=0,31;S F0(32*I+K)=A(K)
01.41 F I=512,1023;S F0(I)=0
01.50 L C,F0
*
```

Ganz ähnlich präsentiert sich die Zerlegung des harmonischen Rechtecks, mit dem Unterschied allerdings, dass nur ungeradzahlige Harmonische auftreten mit Amplituden proportional zu 1, 1/3, 1/5, 1/7, ... Die Vorzeichen der Sprünge nach vollen Perioden sind nun positiv, dementsprechend ist das Vorzeichen des Delta-Kamms positiv.

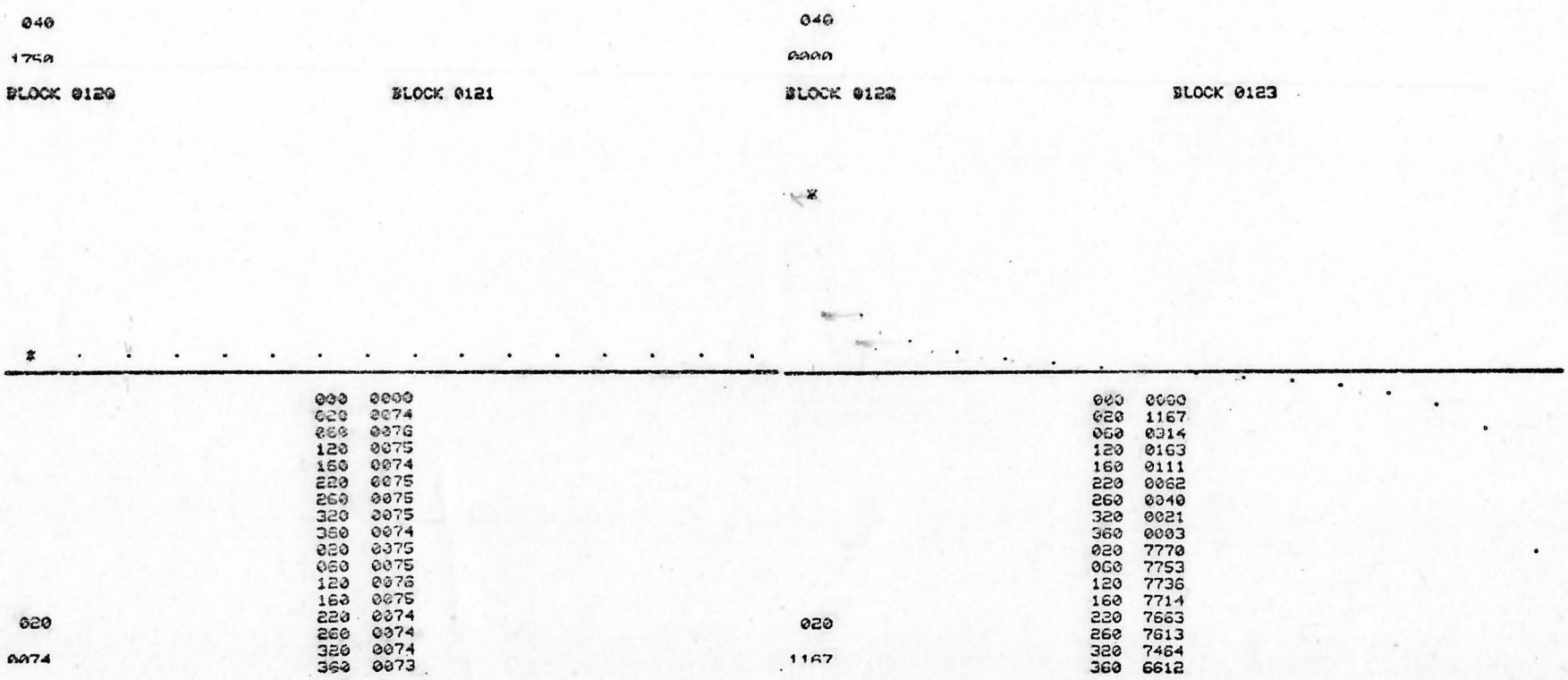


Fig. 14

000	0000	000	0000
020	0074	020	1167
040	0076	040	0314
120	0075	120	0163
160	0074	160	0111
220	0075	220	0062
260	0075	260	0040
320	0075	320	0021
360	0074	360	0003
020	0375	020	7770
060	0075	060	7753
120	0076	120	7736
160	0075	160	7714
220	0074	220	7663
260	0074	260	7613
320	0074	320	7464
360	0073	360	6612

```
*U
C FOCAL-12

01.10 C ********** INKORE **************
01.11 C
01.12 C DIESES PROGRAMM ERZEUGT EINEN INKOHAERENTEN
01.13 C RECHTECK AUF UNIT 1 NR 100-101
01.14 C NR 102-103 SIND GLEICH NULL
01.20 L 0.F0,I,$100.1
01.30 F I=0,14,S A(I)=1000
01.31 F I=15,29,S A(I)=-1000
01.40 F I=0,17;F K=0,29,S F0(30*I+K)=A(K)
01.41 F I=512,1023,S F0(I)=0
01.50 L C,F0
*
```

Die beiden Punkte, die in Block 101 der Fig. 15 ganz rechts noch zusätzlich eingeschoben wurden, machen den Rechteck anharmonisch und sind allein Ursache für die enorme Veränderung gegenüber Fig. 14. Abermals sind die Linien "verschmiert" infolge des 512 Punkte umfassenden Rechteckfensters T. In den anschliessenden Seiten folgt die Tabelle der Punkte in Block 120 - 123. Wie zu erwarten ist, treten Sinus- und Cosinusglieder auf.

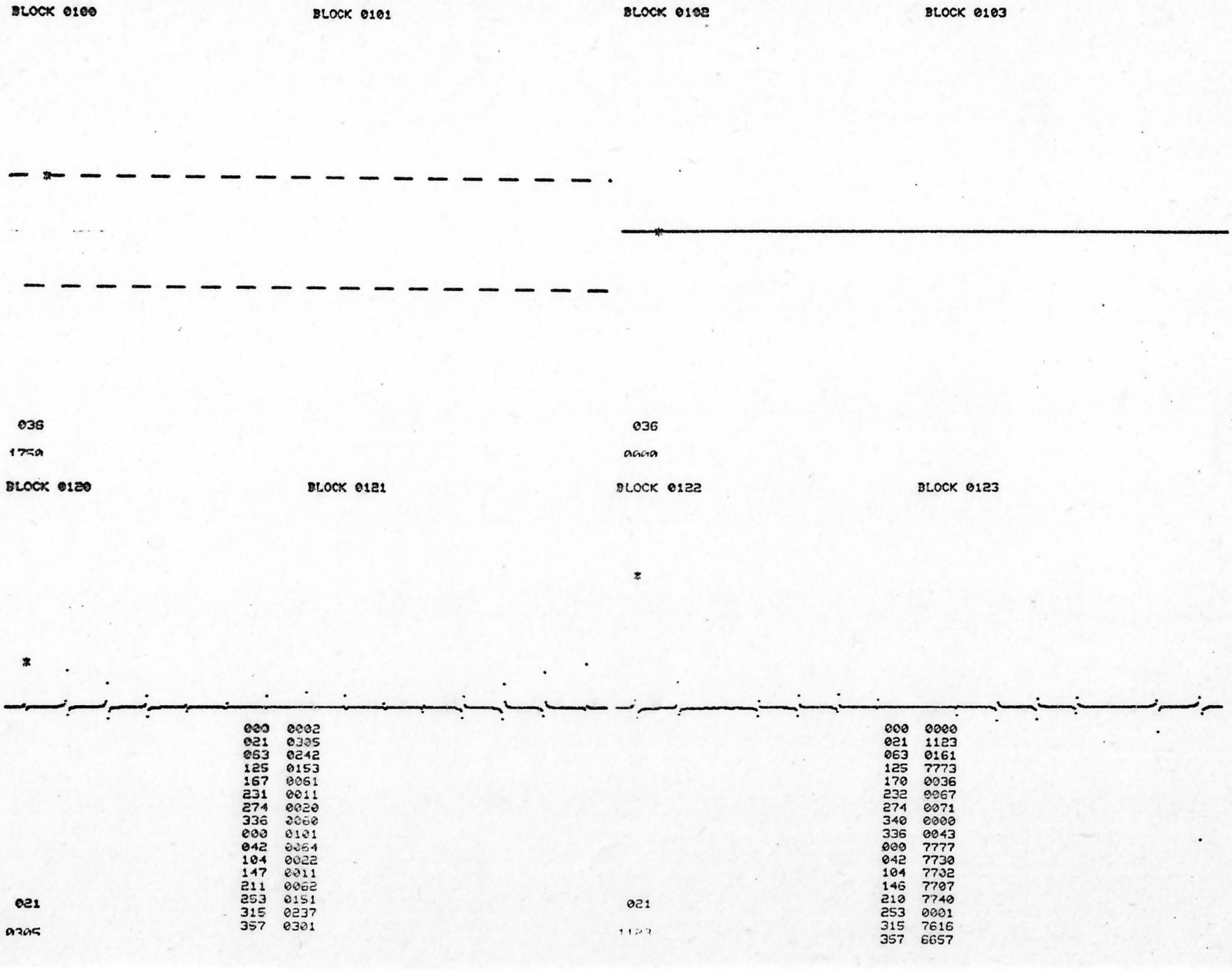

BLOCK 0100
BLOCK 0101
BLOCK 0102
BLOCK 0103
036
1750
036
0000
BLOCK 0120
BLOCK 0121
BLOCK 0122
BLOCK 0123
000 0002
021 0305
063 0242
125 0153
167 0061
231 0011
274 0020
336 0060
000 0101
042 0064
104 0022
147 0011
211 0062
253 0151
315 0237
357 0301
021
0305
000 0000
021 1123
063 0161
125 7773
170 0036
232 0067
274 0071
340 0000
336 0043
000 7777
042 7730
104 7702
146 7707
210 7740
253 0001
315 7616
357 6657
021
1123

Fig. 15

REAL

	0	1	2	3	4	5	6	7
0000	0002	0000	0000	0000	0000	0001	0001	0002
0010	0001	0002	0002	0002	0003	0003	0004	0007
0020	0014	0305	7763	7772	7774	7775	7777	7777
0030	0000	7777	0000	0000	0000	0001	0000	0000
0040	0000	0001	0001	0000	0002	0002	0002	0002
0050	0002	0002	0003	0003	0005	0006	0005	0010
0060	0011	0015	0032	0242	7731	7756	7765	7767
0070	7773	7773	7775	7775	7775	7775	7775	7776
0100	7776	7776	7777	7777	7777	0001	0000	0000
0110	7777	0001	0001	0001	0002	0002	0003	0004
0120	0006	0006	0010	0016	0031	0153	7712	7752
0130	7764	7766	7770	7772	7773	7773	7773	7773
0140	7774	7774	7775	7775	7775	7777	7776	7776
0150	7776	7776	7776	7776	7777	7777	0000	0000
0160	0000	0000	0003	0002	0004	0007	0015	0061
0170	7720	7760	7766	7770	7771	7772	7772	7773
0200	7773	7773	7773	7775	7774	7774	7775	7775
0210	7775	7775	7775	7775	7775	7775	7775	7776
0220	7776	7776	7777	7776	7776	7776	7776	7777
0230	0001	0011	7753	7772	7773	7774	7774	7775
0240	7774	7775	7774	7775	7776	7775	7775	7775
0250	7775	7775	7775	7776	7775	7775	7774	7776
0260	7775	7775	7775	7774	7775	7775	7775	7775
0270	7775	7774	7773	7770	0020	0001	7777	7777
0300	7776	7776	7776	7776	7777	7775	0000	7777
0310	7776	7776	7776	7777	7776	7776	7775	7776
0320	7776	7776	7776	7775	7776	7775	7775	7775
0330	7776	7775	7775	7775	7774	7770	0060	0001
0340	0000	0000	7777	7776	7776	7776	7776	7776
0350	7776	7776	7776	7776	0000	7777	7777	7776
0360	7776	7777	7776	7776	7777	0000	7777	7776
0370	7776	7777	7777	7776	7777	7777	7777	7777
0400	0101	7777	7777	7777	7777	7776	7776	7777
0410	7776	7777	7777	7777	7776	7776	7777	7776
0420	7777	0000	7776	7777	7777	0000	0000	0000
0430	7777	0000	7777	7777	7777	0000	0001	0001
0440	0001	0004	0064	7767	7773	7775	7775	7775
0450	7775	7775	7776	7776	7776	7775	7776	7775
0460	7776	7776	7775	7775	7776	7777	7776	7776
0470	7776	7776	7776	7776	7776	7776	7776	7777
0500	7777	7777	0000	0002	0022	7766	7773	7775
0510	7774	7774	7774	7776	7775	7775	7774	7775
0520	7775	7775	7775	7775	7776	7774	7775	7775
0530	7775	7775	7775	7775	7774	7774	7774	7774
0540	7775	7775	7774	7774	7774	7770	7753	0011
0550	0001	7777	7777	7777	7776	7776	7775	7775
0560	7775	7775	7776	7775	7775	7776	7774	7774
0570	7775	7775	7775	7775	7774	7775	7775	7774
0600	7774	7774	7774	7772	7771	7767	7764	7756
0610	7716	0062	0016	0006	0004	0002	0000	7777
0620	7777	7777	0000	7777	7777	7777	7777	7776
0630	7776	7776	7776	7775	7776	7775	7775	7774
0640	7775	7774	7775	7772	7773	7772	7770	7766
0650	7762	7752	7712	0151	0030	0014	0007	0005
0660	0004	0002	0002	0001	0000	0000	0000	0000
0670	0000	0001	0000	7777	7777	7776	7776	7776
0700	7777	7777	7777	7777	7776	7776	7775	7774
0710	7773	7771	7765	7760	7731	0237	0030	0015
0720	0011	0005	0005	0004	0003	0002	0002	0002
0730	0001	0002	0000	0000	0001	0001	0001	0000
0740	0001	0001	0000	7777	7777	7777	7777	7777
0750	7777	7777	7777	7777	7775	7774	7764	0301
0760	0013	0004	0003	0001	0002	0001	0000	0001
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0000	0000	0000

IMAG

	2	1	2	3	4	5	6	7
0000	0200	7774	7774	7774	7776	7776	7776	7776
0010	7777	0000	0000	0002	0003	0006	0010	0016
0020	0041	1123	7722	7750	7757	7763	7764	7765
0030	7767	7770	7770	7771	7771	7772	7771	7772
0040	7773	7773	7773	7773	7774	7774	7775	7775
0050	7774	7774	7776	7775	7776	7776	0000	0000
0060	0002	0006	0016	0161	7736	7757	7764	7765
0070	7766	7767	7770	7771	7771	7773	7771	7771
0100	7772	7772	7773	7773	7773	7773	7773	7773
0110	7773	7773	7774	7774	7773	7773	7773	7773
0120	7773	7774	7774	7773	7773	7773	7774	7774
0130	7774	7774	7774	7773	7774	7774	7773	7773
0140	7773	7773	7773	7774	7774	7774	7774	7772
0150	7773	7774	7773	7772	7771	7773	7773	7771
0160	7771	7771	7772	7770	7766	7766	7761	7726
0170	0036	0007	0003	0000	7776	7777	7777	7776
0200	7776	7776	7776	7776	7776	7775	7776	7775
0210	7775	7775	7774	7775	7774	7774	7773	7774
0220	7774	7773	7773	7772	7772	7772	7770	7765
0230	7761	7727	0067	0015	0006	0004	0002	0001
0240	0000	0000	0000	0000	0000	7777	7777	7776
0250	7776	7777	7777	7776	7776	7777	7776	7776
0260	7776	7775	7776	7776	7774	7775	7774	7773
0270	7773	7773	7767	7753	0071	0011	0003	0002
0300	0001	0001	0001	0001	0000	0000	0000	7777
0310	7777	7777	7777	7777	0000	7777	0000	0000
0320	7777	7777	7777	7777	7776	7776	7776	7775
0330	7776	7776	7777	7775	7775	7772	0043	0001
0340	0000	0000	0000	0000	7777	0000	7777	7777
0350	7776	7776	7776	7776	7776	7776	7777	7776
0360	7777	7777	7777	0000	7776	7776	7776	0000
0370	7777	7777	7776	7776	7776	7776	7776	7776
0400	7777	7777	7777	7777	7777	7777	7777	7777
0410	0000	7777	7777	7777	7776	7777	7777	7777
0420	7776	7776	7777	7777	7776	7776	7777	7776
0430	7776	7775	7777	7776	7776	7775	7776	7775
0440	7774	7772	7730	0004	0001	0001	0000	7776
0450	7777	7777	7777	7776	7777	7777	7777	7777
0460	7777	7777	7777	7776	7777	7776	7777	7776
0470	7775	7776	7775	7776	7774	7775	7774	7774
0500	7773	7773	7770	7762	7702	0024	0010	0004
0510	0004	0002	0001	0001	0000	0000	0000	0000
0520	0000	7777	7777	0000	0000	0000	7777	7777
0530	7777	7777	7777	7776	7775	7775	7776	7774
0540	7774	7774	7774	7771	7767	7761	7707	0045
0550	0016	0007	0006	0005	0004	0004	0002	0002
0560	0002	0002	0001	0001	0001	0001	0000	0001
0570	7777	0000	0000	7777	7777	7776	7776	7777
0600	7777	7777	7777	7777	7777	7776	7773	7766
0610	7740	0050	0015	0010	0005	0005	0004	0003
0620	0003	0002	0002	0003	0003	0003	0003	0002
0630	0002	0002	0002	0002	0001	0001	0001	0002
0640	0001	0001	0001	0001	0001	0002	0002	0001
0650	0001	0000	0002	0001	0001	0000	0001	0001
0660	0001	0002	0001	0001	0001	0002	0001	0002
0670	0002	0002	0002	0002	0002	0002	0002	0003
0700	0002	0002	0002	0002	0003	0003	0003	0004
0710	0006	0006	0010	0016	0035	7616	7757	7771
0720	7774	7776	7776	7776	7777	7777	7777	0000
0730	7777	7777	0000	0000	0000	0001	0002	0002
0740	0001	0001	0001	0001	0002	0003	0004	0004
0750	0005	0005	0007	0011	0013	0023	0050	6657
0760	7736	7760	7766	7771	7773	7775	7775	7775
0770	7776	7776	7777	7777	7777	7777	7777	7777

```
U
C FOCAL-12

01.10 C ********** IN3777 **************
01.11 C
01.12 C DIESES PROGRAMM ERZEUGT IN NR.100-101
01.13 C DIE KONSTANTE EINS E=3777 OKTAL FRACTION
01.14 C NR.102-103 FUELLT ES MIT NULLEN
01.20 L 0.F0.I.$100.1
01.21 F I=0,511,S F0(I)=2047
01.22 F I=512,1023,S F0(I)=0
01.23 L C.F0
*
*
```

Die Auswirkung des Zeitfensters allein:
RE(A(o))=3776 als einzelne Linie. Sie entspricht genau dem DC-Wert der verschobenen Linie im Eingang. Man kann das eingangsseitige Signal als Fensterfunktion auffassen, die mit dem zu transformierenden Signal multipliziert wird.

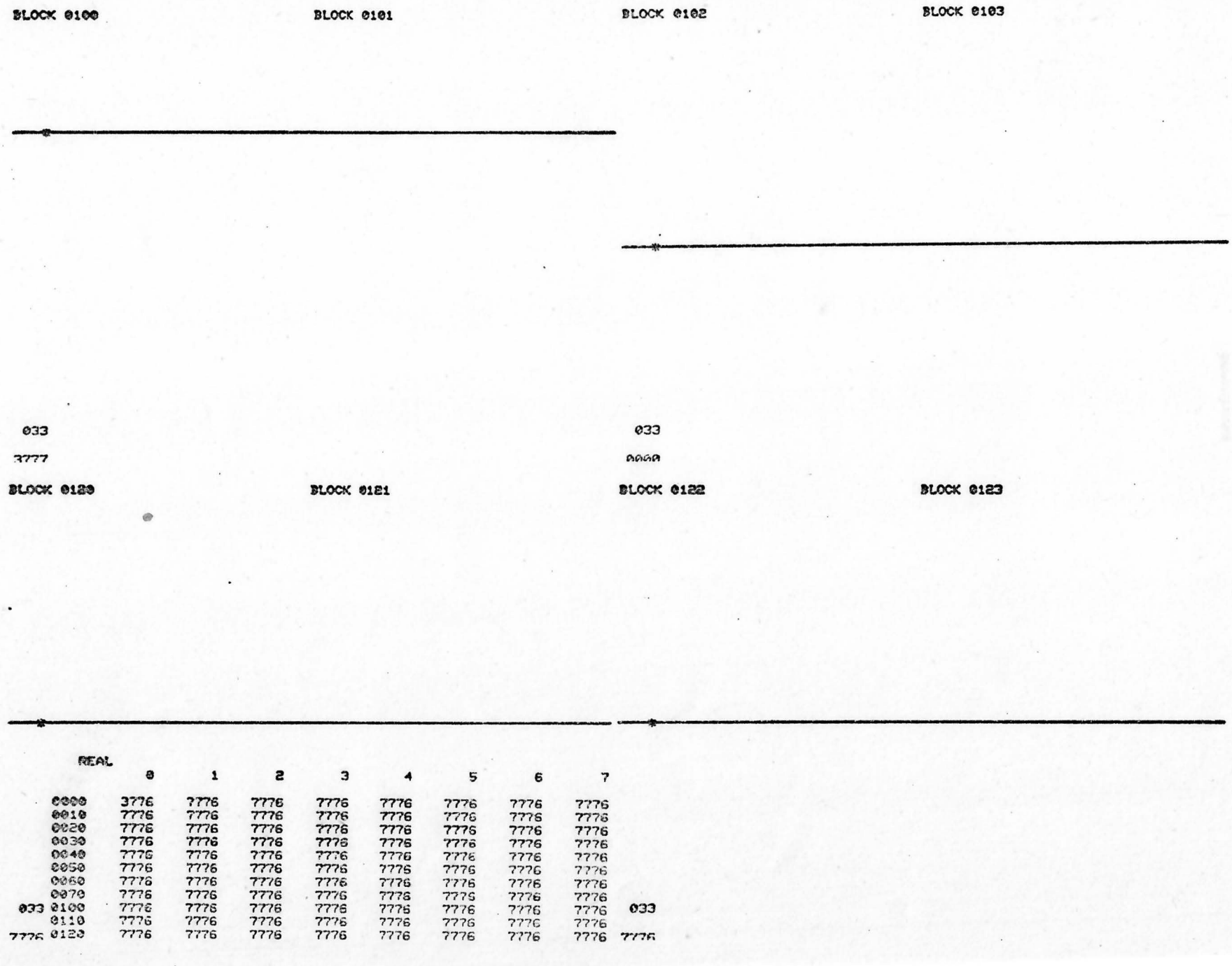
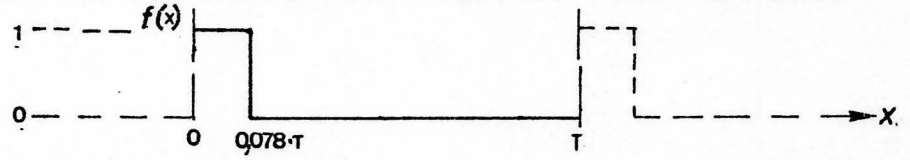

BLOCK 0100
BLOCK 0101
BLOCK 0102
BLOCK 0103
033
3777
BLOCK 0120
BLOCK 0121
033
0000
BLOCK 0122
BLOCK 0123
REAL
0 1 2 3 4 5 6 7
0000 3776 7776 7776 7776 7776 7776 7776 7776
0010 7776 7776 7776 7776 7776 7776 7776 7776
0020 7776 7776 7776 7776 7776 7776 7776 7776
0030 7776 7776 7776 7776 7776 7776 7776 7776
0040 7776 7776 7776 7776 7776 7776 7776 7776
0050 7776 7776 7776 7776 7776 7776 7776 7776
0060 7776 7776 7776 7776 7776 7776 7776 7776
0070 7776 7776 7776 7776 7776 7776 7776 7776
033 0100 7776 7776 7776 7776 7776 7776 7776 7776 033
0110 7776 7776 7776 7776 7776 7776 7776 7776
7776 0120 7776 7776 7776 7776 7776 7776 7776 7776 7776

Fig. 16

```
U
C FOCAL-12

01.10 C ********** IN3777 **************
01.11 C
01.12 C DIESES PROGRAMM ERZEUGT IN NR.100-101
01.13 C EIN KURZES RECHTECKFENSTER DER LAENGE 40
01.14 C NR.102-103 FUELLT ES MIT NULLEN
01.20 L O.F0,I,$100.1
01.21 F I=0.39;S F0(I)=2047
01.22 F I=40.1023;S F0(I)=0
01.23 L C.F0
*
*GO
```

Sobald das Fenster schmäler gemacht wird, wie in Fig. 17, ändert sich das Bild. Die Fourier-Transformierte eines Rechteckstosses ist eine - grob gesagt - sinx/x-Funktion. Genauer gilt für die Fourierkoeffizienten der Funktion



$$a_k = \frac{2}{T}\int_0^{0,0785} 1 \cdot \cos k\frac{2\pi}{T}x\,dx = \frac{1}{\pi k}\sin(0,156 \cdot \pi \cdot k) \qquad k = 0,1,2,3,\ldots$$

$$b_k = \frac{2}{T}\int_0^{0,0785} 1 \cdot \sin k\frac{2\pi}{T}x\,dx = \frac{1}{\pi k}(1 - \cos 0,156\,\pi k)$$

a_k wird erstmals Null zwischen k = 6 und k = 7

b_k wird erstmals Null für das Doppelte, ca. k = 13,

wo $\cos(0,156\,k\,\pi) \approx +1$ ist. b_o ist Null; $a_o = 0,156 \Longrightarrow RE(A(o)) =$

$$= \frac{a_o}{2} = 0,078 \triangleq 0237_8$$

All dies wird vom Graph in Fig. 17 und der anschliessenden Tabelle schön bestätigt.

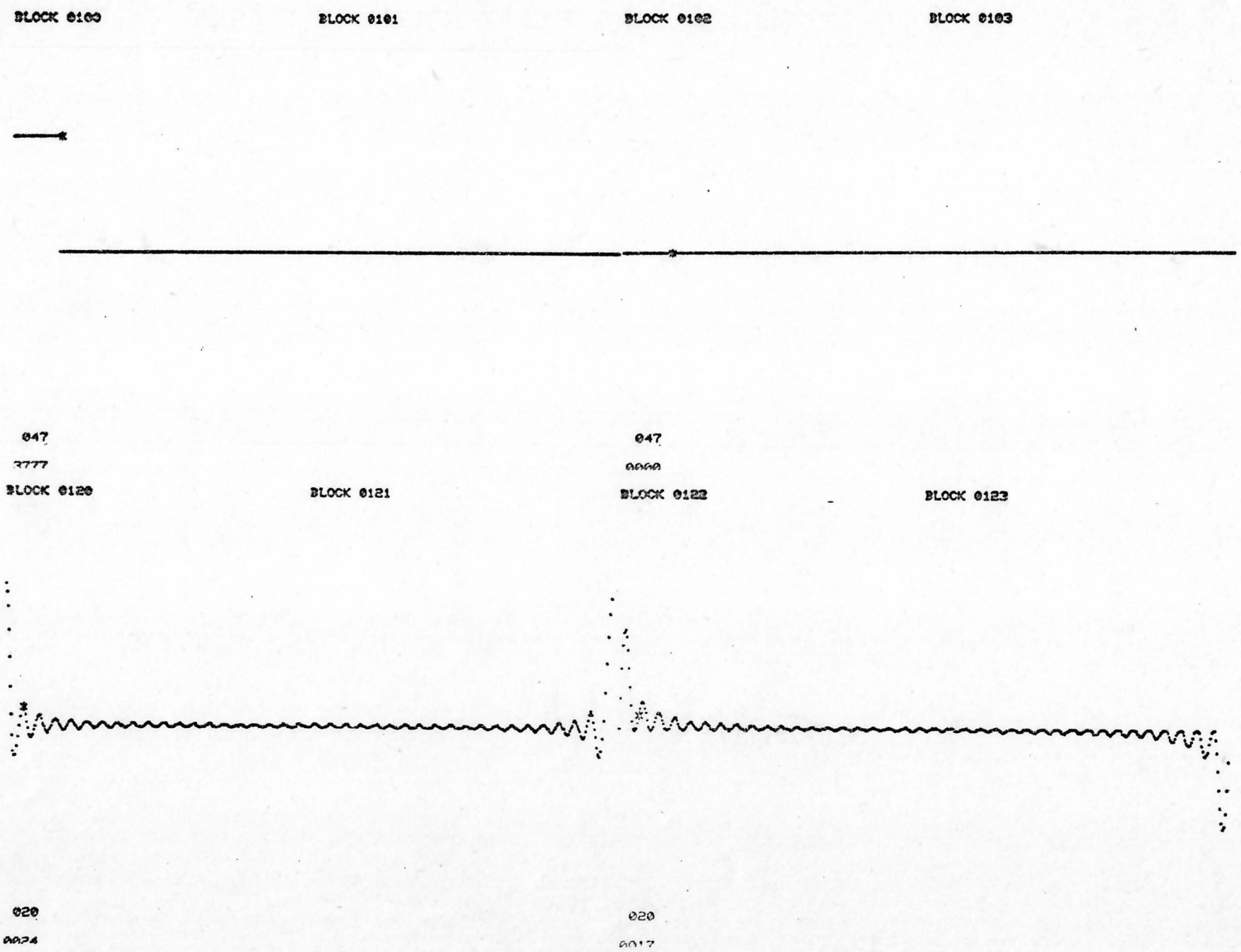

BLOCK 0100
BLOCK 0101
BLOCK 0102
BLOCK 0103
047
3777
BLOCK 0120
BLOCK 0121
047
0000
BLOCK 0122
BLOCK 0123
020
0024
020

Fig. 17

REAL

	0	1	2	3	4	5	6	7
0000	0236	0224	0204	0152	0113	0052	0014	7764
0010	7743	7736	7737	7750	7763	0000	0012	0022
0020	0024	0022	0014	0004	7774	7765	7762	7762
0030	7765	7772	0000	0006	0011	0013	0013	0007
0040	0002	7776	7772	7770	7767	7772	7774	0000
0050	0004	0006	0007	0006	0005	0002	7777	7774
0060	7772	7772	7773	7776	0001	0004	0005	0006
0070	0006	0003	0001	7777	7774	7773	7773	7775
0100	7776	0002	0003	0004	0004	0004	0003	0002
0110	7777	7776	7774	7775	7776	7776	0001	0002
0120	0003	0004	0003	0003	0002	0000	7777	7775
0130	7775	7775	7776	0001	0002	0003	0003	0003
0140	0002	0000	0001	7776	7775	7775	7776	7777
0150	0001	0002	0003	0003	0003	0002	0000	7777
0160	7777	7776	7776	7776	7777	0001	0002	0003
0170	0004	0003	0002	0001	0000	7777	7776	7776
0200	7776	0000	0001	0002	0003	0003	0003	0003
0210	0001	7777	7776	7777	7776	7776	0000	0001
0220	0001	0002	0002	0003	0002	0001	0000	7777
0230	7776	7776	7777	0000	0001	0001	0002	0002
0240	0003	0001	0000	0000	7776	7776	7777	7777
0250	0000	0001	0002	0002	0003	0003	0002	0000
0260	0000	0000	7776	7776	7777	0000	0001	0002
0270	0002	0001	0002	0002	0001	0000	7777	7776
0300	7776	0000	0000	0001	0001	0000	0002	0002
0310	0002	0001	7777	7777	7776	7776	7777	0001
0320	0001	0002	0001	0002	0002	0001	0001	0000
0330	7777	7777	7776	7777	0000	0001	0002	0002
0340	0002	0002	0001	0000	0000	0000	7776	7777
0350	0000	0001	0001	0003	0001	0002	0002	0000
0450	7777	7777	0000	0001	0002	0003	0002	0001
0460	0001	0001	0000	7777	7776	7777	0000	0001
0470	0001	0002	0002	0002	0003	0002	0000	0000
0500	7777	7777	7776	7777	0001	0001	0002	0003
0510	0002	0001	0001	0000	7777	7777	7776	7777
0520	0000	0001	0002	0002	0003	0003	0002	0002
0530	0000	7776	7777	7776	7777	0000	0000	0002
0540	0003	0003	0002	0001	0000	0000	7777	7776
0550	7776	7777	0000	0000	0002	0003	0003	0002
0560	0002	0001	7777	7777	7776	7776	7777	0000
0570	0001	0002	0003	0002	0003	0002	0001	7777
0600	7777	7777	7776	7777	0000	0000	0002	0002
0610	0002	0002	0001	0000	7777	7777	7777	7776
0620	7776	7777	0001	0002	0003	0002	0003	0002
0630	7777	7777	7776	7775	7776	7776	7777	0001
0640	0002	0002	0003	0003	0001	7777	0000	7776
0650	7775	7774	7775	7777	0000	0002	0003	0003
0660	0003	0001	7777	7777	7776	7775	7774	7775
0670	7777	0000	0001	0003	0004	0003	0002	7777
0700	7777	7775	7773	7774	7776	7777	0001	0003
0710	0003	0004	0004	0002	7777	7777	7774	7772
0720	7772	7773	7776	0001	0003	0006	0006	0005
0730	0002	0000	7777	7772	7771	7770	7771	7775
0740	0001	0003	0010	0011	0007	0003	7777	7774
0750	7767	7764	7764	7766	7774	0003	0011	0017
0760	0023	0017	0007	7777	7766	7752	7743	7740
0770	7746	7764	0011	0050	0110	0146	0177	0220

IMAG	0	1	2	3	4	5	6	7
0000	0000	0041	0102	0133	0152	0155	0145	0126
0010	0102	0055	0031	0010	7776	7774	7777	0006
0020	0017	0026	0034	0035	0034	0026	0021	0011
0030	0002	7776	7774	7776	0002	0007	0014	0020
0040	0021	0020	0016	0011	0005	0000	7777	7775
0050	7776	0000	0004	0010	0012	0014	0014	0011
0060	0005	0002	7777	7776	7775	7775	0000	0002
0070	0005	0006	0007	0007	0005	0003	0002	0000
0100	7776	7775	7776	7777	0002	0005	0006	0007
0110	0005	0004	0003	7777	7777	7776	7775	7775
0120	7777	0001	0003	0005	0006	0004	0003	0001
0130	7777	7777	7776	7774	7776	7777	0001	0004
0140	0003	0004	0005	0002	0001	7777	7776	7776
0150	7775	7774	7777	0000	0003	0002	0003	0003
0160	0002	0001	7777	7776	7775	7776	7776	7776
0170	7777	0001	0002	0002	0003	0001	0001	7777
0200	7776	7776	7775	7775	0000	0001	0002	0003
0210	0002	0001	0000	0000	7777	7776	7776	7775
0220	7775	7777	0000	0001	0002	0002	0001	0000
0230	0000	7776	7776	7775	7776	7776	7777	0000
0240	0001	0001	0001	0001	0000	7777	7776	7776
0250	7775	7776	7777	7777	7777	0000	0001	0001
0260	0001	0000	0000	7776	7775	7775	7776	7775
0270	7777	7777	0001	0001	0001	0001	0000	7777
0300	7776	7776	7774	7775	7775	7777	7777	0001
0310	0001	0001	0000	0000	7776	7776	7776	7775
0320	7775	7776	7776	7777	0000	0001	0001	0000
0330	7777	7776	7776	7775	7775	7776	7776	7776
0460	0000	0001	0000	7777	7776	7776	7775	7775
0470	7774	7775	7776	0000	0000	0000	0001	7777
0500	7777	7776	7775	7774	7773	7774	7775	7776
0510	7776	7777	0000	0000	0000	7777	7776	7776
0520	7774	7774	7774	7774	7775	7777	0000	0000
0530	0000	0000	7777	7775	7775	7774	7774	7773
0540	7774	7775	7776	7777	0000	0000	7777	7777
0550	7776	7775	7774	7773	7772	7775	7776	7776
0560	7777	0000	0000	7777	7776	7775	7773	7773
0570	7772	7774	7775	7775	7776	0000	0000	0000
0600	7777	7777	7776	7774	7773	7774	7773	7774
0610	7775	7776	7777	7777	0000	7777	7777	7776
0620	7774	7772	7773	7772	7773	7775	7776	7777
0630	7777	7777	7777	7776	7775	7773	7772	7771
0640	7772	7774	7774	7776	7777	0000	7777	7777
0650	7776	7773	7772	7772	7770	7771	7772	7774
0660	7776	7777	7777	7777	7776	7776	7773	7772
0670	7770	7770	7770	7772	7774	7776	7777	0000
0700	7777	7777	7775	7772	7770	7766	7766	7766
0710	7770	7772	7775	7777	7777	7777	7777	7774
0720	7770	7765	7763	7762	7763	7766	7772	7775
0730	7776	7777	7777	7776	7772	7765	7761	7757
0740	7756	7757	7762	7767	7775	7776	7777	7777
0750	7775	7766	7756	7750	7743	7742	7744	7747
0760	7760	7767	7775	7777	7777	7765	7746	7723
0770	7676	7652	7632	7623	7626	7644	7676	7735

```
W
C FOCAL-12

01.10 C ********** COSWIND **********
01.11 C
01.12 C DIESES PROGRAMM MULTIPLIZIERT DIE EINGANGS-
01.13 C VEKTOREN IN NR 100-101 BZW. 102-103
01.14 C MIT EINEM COSINE-SQUARED WINDOW
01.15 C DAS WINDOW BEFINDET SICH IN NR 120-127
01.16 C DIE ABSCHWAECHUNG WIRKT AUF JE N RANDPUNKTE
01.17 S N=200

02.10 L 0,F0,S,#120,1
02.11 L 0,F1,S,#130,1
02.12 S PI=3.1415926535
02.20 F I=0,N-1;S F1(I)=FSIN(PI*I/(2*N))*FSIN(PI*I/(2*N))
02.30 F I=0,1023;S F0(I)=1
02.40 F I=0,N-1;S F0(I)=F1(I)
02.41 F I=0,N-1;S F0(511-I)=F1(I)
02.42 F I=0,N-1;S F0(512+I)=F1(I)
02.43 F I=0,N-1;S F0(1023-I)=F1(I)
02.50 L 0,F2,I,#100,1
02.60 F I=0,1023;S F2(I)=F2(I)*F0(I)

03.10 L C,F0;L C,F1;L C,F2
*GO
```

Die Randabschwächung (tapering) des Rechteckfensters T wird hier mit einer halben Periode eines quadrierten Cosinus vorgenommen, wobei ein grosser Teil des Fensters miteinbezogen ist. Das Spektrum enthält eine grosse DC-Linie und nur etwa 3 Cosinus-Linien, wovon zwei mit negativem Vorzeichen.

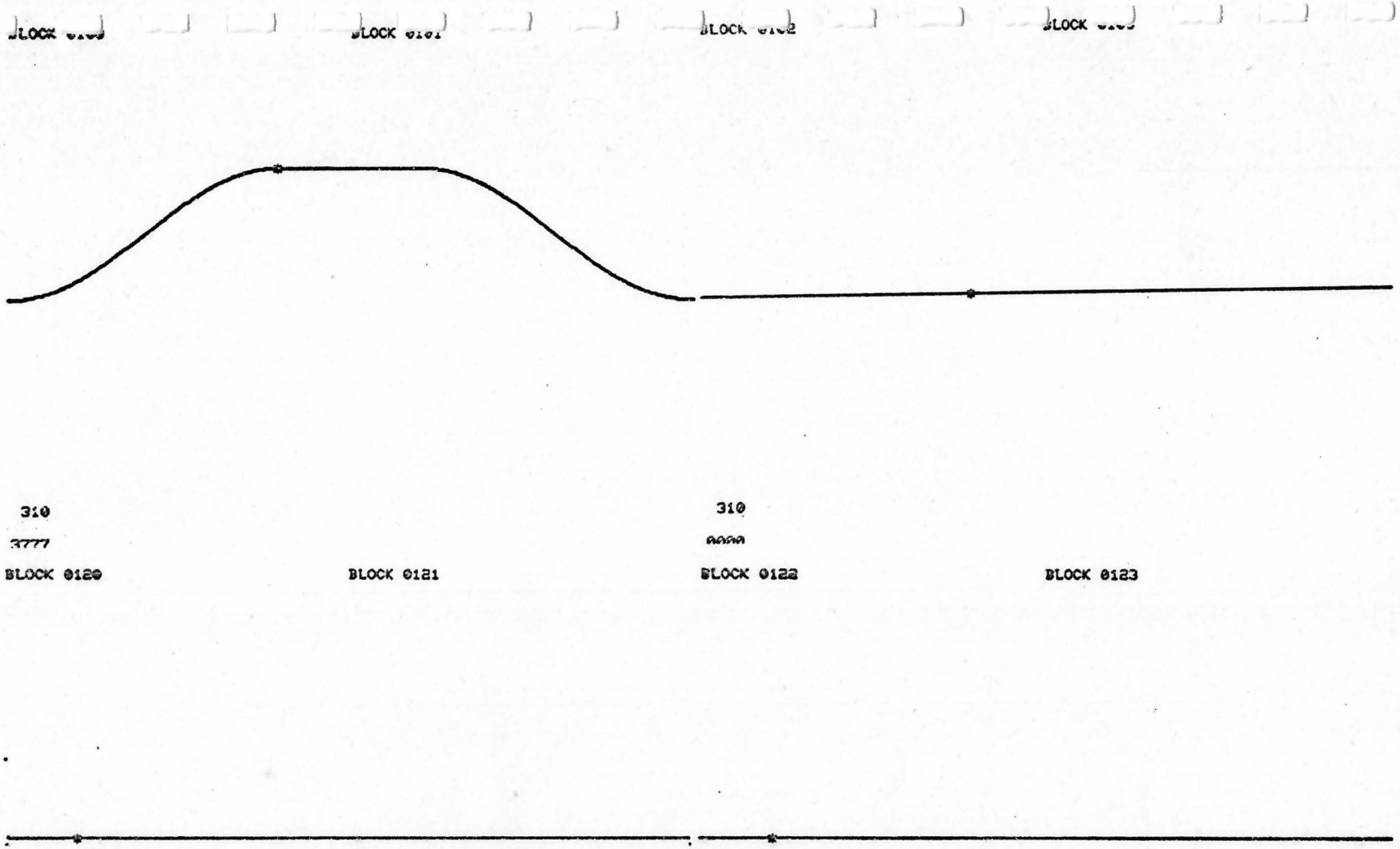


Fig. 18

	REAL 0	1	2	3	4	5	6	7	
0000	2330	6752	7620	0024	7772	7774	0000	7774	
0010	7776	7776	7776	7776	7776	7776	7776	7775	
0020	7776	7776	7776	7776	7776	7776	7776	7776	
065 0750	7777	0000	7777	7777	7777	7777	7777	0000	065
0760	7777	7777	7777	7777	7777	0000	7777	7777	
7776 0770	0000	7777	0000	7777	7777	0015	7626	6762	7776

```
J
C FOCAL-12

01.10 C ********** KORECHT **********
01.11 C
01.12 C DIESES PROGRAMM ERZEUGT EINEN KOHAERENTEN
01.13 C RECHTECK AUF UNIT 1 NR 100-101
01.14 C NR 102-103 SIND GLEICH NULL
01.20 L O,F0,I,$100,1
01.30 F I=0,15;S A(I)=1000
01.31 F I=16,31;S A(I)=-1000
01.40 F I=0,15;F K=0,31;S F0(32*I+K)=A(K)
01.41 F I=512,1023;S F0(I)=0
01.50 L C,F0
*GO
*


*W
C FOCAL-12

01.10 C ********** COSWIND **********
01.11 C
01.12 C DIESES PROGRAMM MULTIPLIZIERT DIE EINGANGS-
01.13 C VEKTOREN IN NR 100-101 BZW. 102-103
01.14 C MIT EINEM COSINE-SQUARED WINDOW
01.15 C DAS WINDOW BEFINDET SICH IN NR 120-127
01.16 C DIE ABSCHWAECHUNG WIRKT AUF JE N RANDPUNKTE
01.17 S N=200

02.10 L O,F0,S,$120,1
02.11 L O,F1,S,$130,1
02.12 S PI=3.1415926535
02.20 F I=0,N-1;S F1(I)=FSIN(PI*I/(2*N))*FSIN(PI*I/(2*N))
02.30 F I=0,1023;S F0(I)=1
02.40 F I=0,N-1;S F0(I)=F1(I)
02.41 F I=0,N-1;S F0(511-I)=F1(I)
02.42 F I=0,N-1;S F0(512+I)=F1(I)
02.43 F I=0,N-1;S F0(1023-I)=F1(I)
02.50 L O,F2,I,$100,1
02.60 F I=0,1023;S F2(I)=F2(I)*F0(I)

03.10 L C,F0;L C,F1;L C,F2
*GO
```

Ein harmonischer Rechteck X(j) wird mit dem in Fig. 18 vorgestellten Cosinus-Window C(j) multipliziert. Frequenzseitig entspricht das gemäss dem Faltungstheorem einer Faltung der entsprechenden Fouriertransformierten $\tilde{X}(n)$ und $\tilde{C}(n)$

$$X(j).C(j) \xleftrightarrow{FT} A(n)=\tilde{X}(n)*\tilde{C}(n)=\sum_{k=0}^{N-1}\tilde{C}(K).\tilde{X}(n-k)$$

Ist $\tilde{X}(n)$ eine scharfe Linie (festes n), wie sie in Fig. 14 ausschliesslich auftreten, so ist in diesem Fall A(n) durch das $\tilde{C}(o)<1$ wesentlich kleiner. Insgesamt bildet sich die transformierte Fensterfunktion $\tilde{C}(n)$ über jeder scharfen Linie ab, was den typischen Verlauf von Fig. 19 erklärt.

BLOCK 0100 BLOCK 0101 BLOCK 0102 BLOCK 0103

Fig. 19

310 COSINE-SQUARED FENSTER WIRKT LINKS VOM STERN

1750

310

0000

BLOCK 0120 BLOCK 0121 BLOCK 0122 BLOCK 0123

020	0044
060	0044
120	0045
160	0044
220	0045
260	0045
320	2044
360	0044
020	0045
060	0045
120	0044
160	0045
220	0044
260	0044
320	0043
360	0043

020	0576
060	0172
120	0105
157	7752
160	0053
220	0035
260	0022
320	0011
360	0002
020	7773
060	7763
120	7752
160	7740
220	7722
260	7671
320	7604
360	7203

017

7756

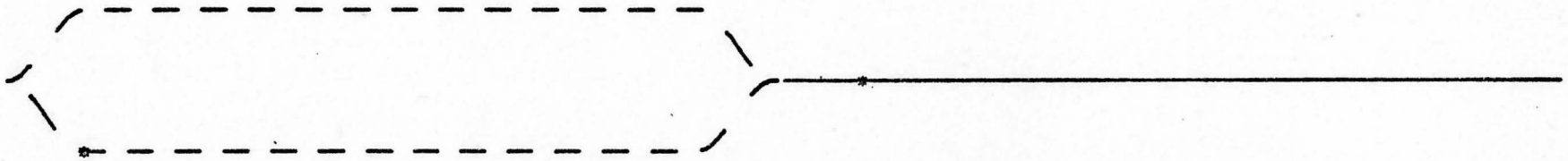
017

7532

```
W
C FOCAL-12

01.10 C ********** COSWIND **********
01.11 C
01.12 C DIESES PROGRAMM MULTIPLIZIERT DIE EINGANGS-
01.13 C VEKTOREN IN NR 100-101 BZW. 102-103
01.14 C MIT EINEM COSINE-SQUARED WINDOW
01.15 C DAS WINDOW BEFINDET SICH IN NR 120-127
01.16 C DIE ABSCHWAECHUNG WIRKT AUF JE N RANDPUNKTE
01.17 S N=50

02.10 L O,F0,S,$120,1
02.11 L O,F1,S,$130,1
02.12 S PI=3.1415926535
02.20 F I=0,N-1;S F1(I)=FSIN(PI*I/(2*N))*FSIN(PI*I/(2*N))
02.30 F I=0,1023;S F0(I)=1
02.40 F I=0,N-1;S F0(I)=F1(I)
02.41 F I=0,N-1;S F0(511-I)=F1(I)
02.42 F I=0,N-1;S F0(512+I)=F1(I)
02.43 F I=0,N-1;S F0(1023-I)=F1(I)
02.50 L O,F2,I,$100,1
02.60 F I=0,1023;S F2(I)=F2(I)*F0(I)

03.10 L C,F0;L C,F1;L C,F2
*GO
```

In Fig. 20 wurde der Wirkungsbereich des tapering verkleinert. Demgemäss ändert sich der Verlauf der Fenstertransformierten. Qualitativ ist aber dieselbe Charakteristik wie in Fig. 19 zu erkennen. Der Rechteck ist nach wie vor harmonisch in T.

BLOCK 0100 BLOCK 0101 BLOCK 0102 BLOCK 0103



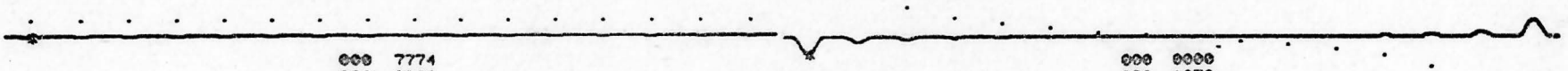
062 COSINE-SQUARED FENSTER WIRKT LINKS VOM STERN

6030

062

0000

BLOCK 0120 BLOCK 0121 BLOCK 0122 BLOCK 0123



000	7774	000	0000
020	0066	020	1070
060	0066	060	0256
120	0066	120	0147
160	0067	160	0102
220	0067	220	0055
260	0066	280	0034
320	0067	320	0017
360	0066	360	0004
020	0067	020	7771
060	0067	060	7755
120	0067	120	7740
160	0066	160	7721
220	0066	220	7674
260	0065	260	7627
320	0066	320	7510
360	0065	360	6711

021

7770

021

7760

Fig. 20

```
*L L.KORECHT.10
#W
C FOCAL-12

01 10 C ********** KORECHT **********
01 11 C
01 12 C DIESES PROGRAMM ERZEUGT EINEN KOHAERENTEN
01 13 C RECHTECK AUF UNIT 1 NR 100-101
01 14 C NR 102-103 SIND GLEICH NULL
01 20 L 0,F0,I,#100,1
01 30 F I=0,15;S A(I)=1000
01 31 F I=16,31,S A(I)=-1000
01 40 F I=0,15,F K=0,31;S F0(32*I+K)=A(K)
01 41 F I=512,1023,S F0(I)=0
01 50 L C,F0
*GO
```

Der FFT-Algorithmus wurde in diesem Beispiel insofern verändert, als nun nicht mehr die volle Wortlänge von 12 bit wirksam ist. Mit einer Maske MSC, die auf Zeile 20 unter den initial constants zu finden ist (vgl. Seite 20), wurden an den Register- und ROM-Ausgängen Engpässe eingeführt. In diesem Beispiel habe ich so eine 8 bit Maschine simuliert. Gegenüber der 12-bit-Version Fig. 14 ist ein verstärktes Quantisierungsgeräusch zu erkennen. Die 8 bit-Simulation ist nicht ganz korrekt, denn die Arithmetik wurde intern nicht maskiert. Eine konsequente Maskierung wäre ziemlich aufwendig.

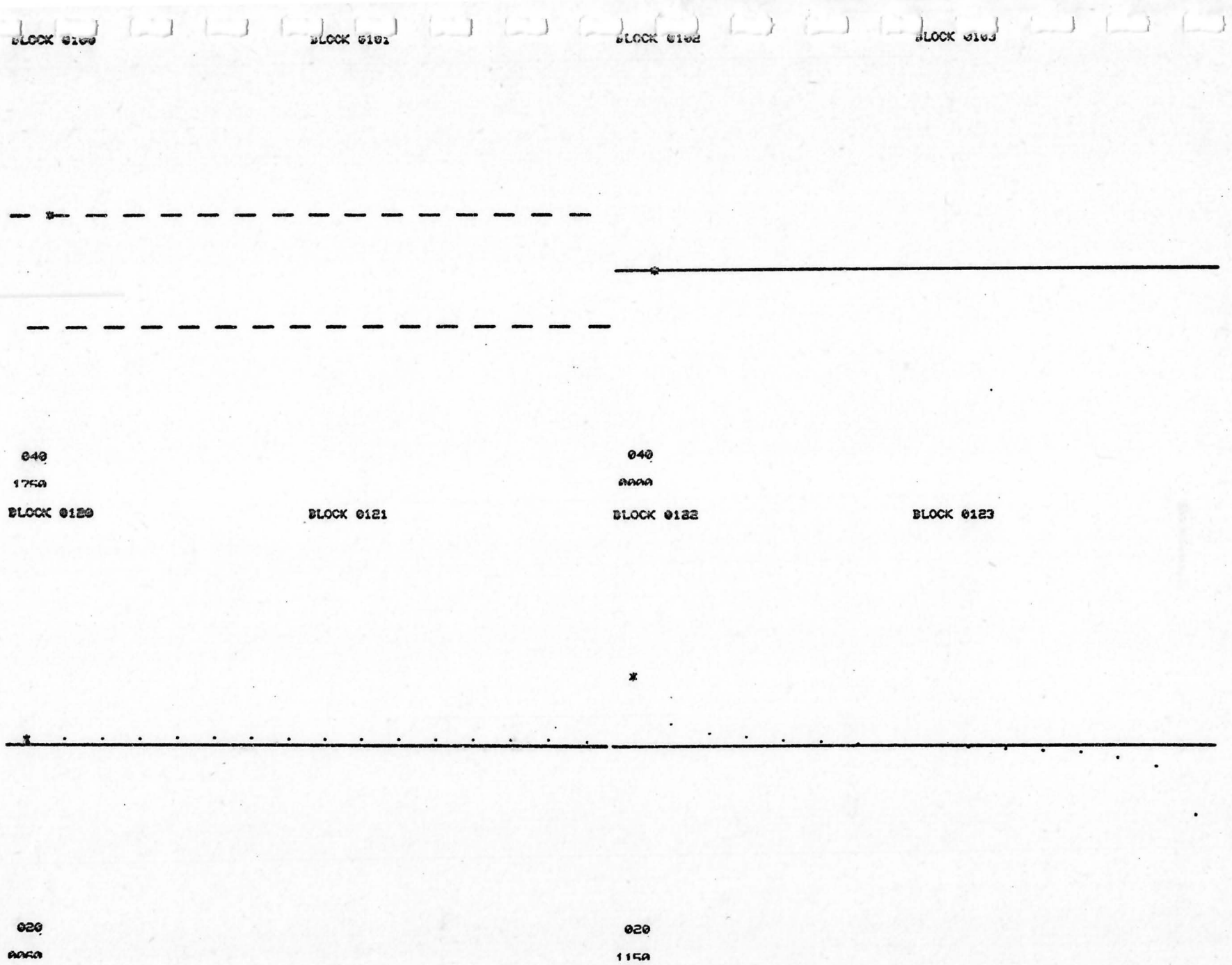

BLOCK 0100
BLOCK 0101
BLOCK 0102
BLOCK 0103
040
1750
BLOCK 0120
BLOCK 0121
040
0000
BLOCK 0122
BLOCK 0123
020
0050
020
1150

Fig. 21

```
*W
C FOCAL-12

01.10 C ********** INKODRE ***********
01.11 C
01.12 C DIESE PROGRAMM ERZEUGT EINEN INKOHAERENTEN
01.13 C DREIECK IN UNIT 1 NR 102-103 [IMAGINAER!]
01.14 C REAL INBATCH NR 100-101 SIND GLEICH NULL
01.20 L O,F0,I,$100,1
01.30 F I=0,17;S A(I)=I-9
01.31 F I=18,35;S A(I)=27-I
01.40 F I=0,15;F K=0,35,S F0(36*I+K+512)=A(K)*200
01.50 F I=0,511;S F0(I)=0

02.10 L C,F0
*GO
```

Ebenfalls eine 8 bit-Simulation mit einem anharmonischen Dreieck. Die Quantisierungsstufen von 0020 werden hier schon sehr auffällig, da das Spektrum verschmiert ist. Die verstärkte Fig. 22 zeigt das noch deutlicher. Der Quantisierungsschritt (Differenz zwischen beiden Sternen) ist de facto bloss 0010, wenigstens in der linken Hälfte, was eben damit zusammenhängt, dass ich in der Arithmetik keine Zwischenmasken eingesetzt habe. Die Einschränkung auf 8 bit führt in der anschliessenden Tabelle zu einer Null in der letzten Stelle jeder Oktalzahl.

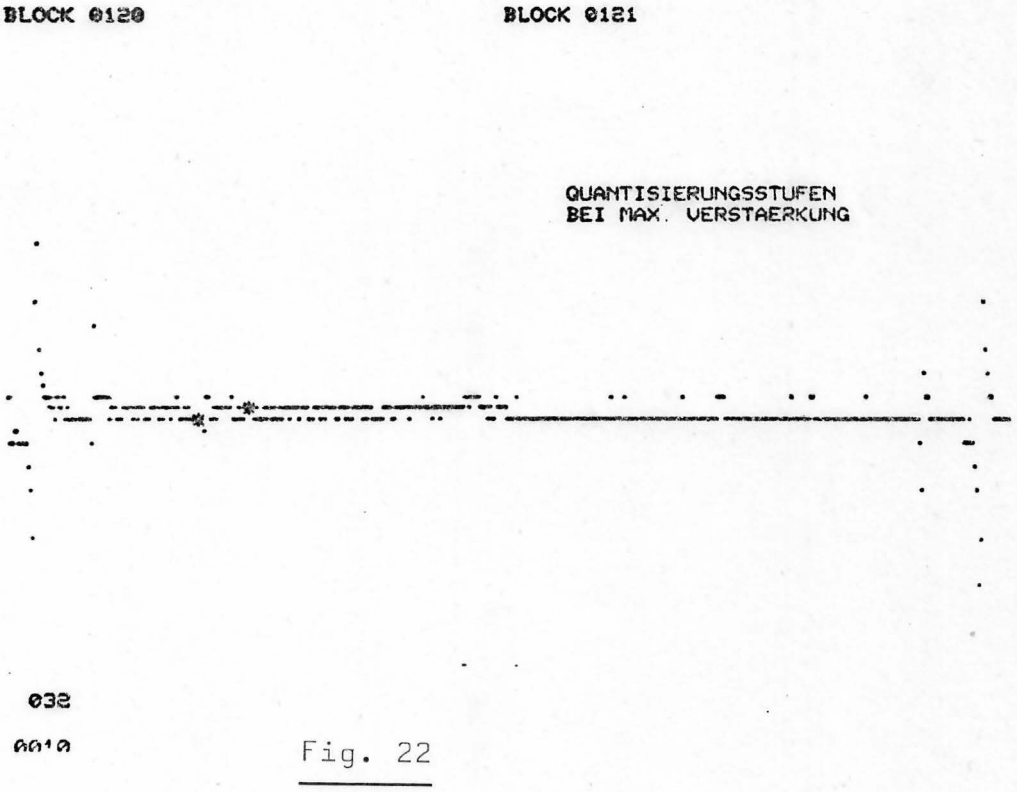

BLOCK 0120
BLOCK 0121
QUANTISIERUNGSSTUFEN
BEI MAX. VERSTAERKUNG
032
0010

Fig. 22

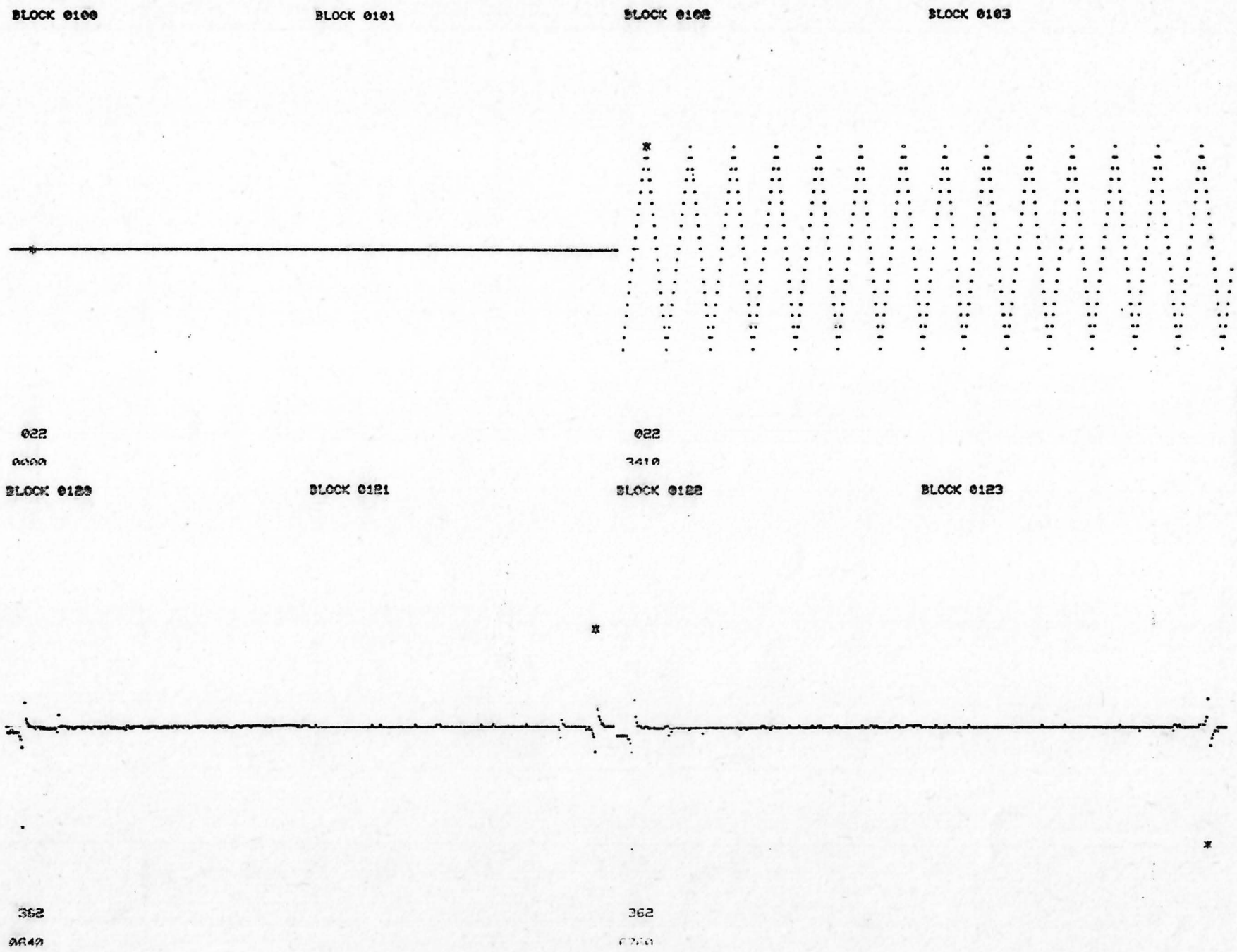

BLOCK 0100
BLOCK 0101
BLOCK 0102
BLOCK 0103
022
0000
022
3410
BLOCK 0120
BLOCK 0121
BLOCK 0122
BLOCK 0123
362
0640
362

Fig. 23

REAL

	0	1	2	3	4	5	6	7
0000	7760	7760	7760	7760	7760	7760	7760	7760
0010	7760	7760	7760	7760	7760	7760	7760	7760
0020	0060	7760	7760	7760	7760	7760	7760	7760
0030	7760	7760	7760	7760	7760	7760	7760	7760
0040	7760	7760	7760	7760	7760	7760	7760	7760
0050	7760	7760	7760	7760	7760	7760	7760	7760
0060	0060	7760	7760	7760	7760	7760	7760	7760
0070	7760	7760	7760	7760	7760	7760	7760	7760
0100	7760	7760	7760	7760	7760	7760	7760	7760
0110	7760	7760	7760	7760	7760	7760	7760	7760
0120	0060	7760	7760	7760	7760	7760	7760	7760
0130	7760	7760	7760	7760	7760	7760	7760	7760
0140	7760	7760	7760	7760	7760	7760	7760	7760
0150	7760	7760	7760	7760	7760	7760	7760	7760
0160	0060	7760	7760	7760	7760	7760	7760	7760
0170	7760	7760	7760	7760	7760	7760	7760	7760
0200	7760	7760	7760	7760	7760	7760	7760	7760
0210	7760	7760	7760	7760	7760	7760	7760	7760
0220	0070	7760	7760	7760	7760	7760	7760	7760
0230	7760	7760	7760	7760	7760	7760	7760	7760
0240	7760	7760	7760	7760	7760	7760	7760	7760
0250	7760	7760	7760	7760	7760	7760	7760	7760
0260	0070	7760	7760	7760	7760	7760	7760	7760
0270	7760	7760	7760	7760	7760	7760	7760	7760
0300	7760	7760	7760	7760	7760	7760	7760	7760
0310	7760	7760	7760	7760	7760	7760	7760	7760
0320	0070	7760	7760	7760	7760	7760	7760	7760
0330	7760	7760	7760	7760	7760	7760	7760	7760
0340	7760	7760	7760	7760	7760	7760	7760	7760
0350	7760	7760	7760	7760	7760	7760	7760	7760
0360	0060	7760	7760	7760	7760	7760	7760	7760
0370	7760	7760	7760	7760	7760	7760	7760	7760
0400	7760	7760	7760	7760	7760	7760	7760	7760
0410	7760	7760	7760	7760	7760	7760	7760	7760
0420	0060	7760	7760	7760	7760	7760	7760	7760
0430	7760	7760	7760	7760	7760	7760	7760	7760
0440	7760	7760	7760	7760	7760	7760	7760	7760
0450	7760	7760	7760	7760	7760	7760	7760	7760
0460	0060	7760	7760	7760	7760	7760	7760	7760
0470	7760	7760	7760	7760	7760	7760	7760	7760
0500	7760	7760	7760	7760	7760	7760	7760	7760
0510	7760	7760	7760	7760	7760	7760	7760	7760
0520	0060	7760	7760	7760	7760	7760	7760	7760
0530	7760	7760	7760	7760	7760	7760	7760	7760
0540	7760	7760	7760	7760	7760	7760	7760	7760
0550	7760	7760	7760	7760	7760	7760	7760	7760
0560	0060	7760	7760	7760	7760	7760	7760	7760
0570	7760	7760	7760	7760	7760	7760	7760	7760
0600	7760	7760	7760	7760	7760	7760	7760	7760
0610	7760	7760	7760	7760	7760	7760	7760	7760
0620	0060	7760	7760	7760	7760	7760	7760	7760
0630	7760	7760	7760	7760	7760	7760	7760	7760
0640	7760	7760	7760	7760	7760	7760	7760	7760
0650	7760	7760	7760	7760	7760	7760	7760	7760
0660	0060	7760	7760	7760	7760	7760	7760	7760
0670	7760	7760	7760	7760	7760	7760	7760	7760
0700	7760	7760	7760	7760	7760	7760	7760	7760
0710	7760	7760	7760	7760	7760	7760	7760	7760
0720	0060	7760	7760	7760	7760	7760	7760	7760
0730	7760	7760	7760	7760	7760	7760	7760	7760
0740	7760	7760	7760	7760	7760	7760	7760	7760
0750	7760	7760	7760	7760	7760	7760	7760	7760
0760	0040	7760	7760	7760	7760	7760	7760	7760
0770	7760	7760	7760	7760	7760	7760	7760	7760

IMAG

	0	1	2	3	4	5	6	7
0000	0000	7760	7760	7760	7760	7760	7760	7760
0010	7760	7760	7760	7760	7760	7760	7760	7760
0020	1150	7760	7760	7760	7760	7760	7760	7760
0030	7760	7760	7760	7760	7760	7760	7760	7760
0040	7760	7760	7760	7760	7760	7760	7760	7760
0050	7760	7760	7760	7760	7760	7760	7760	7760
0060	0300	7760	7760	7760	7760	7760	7760	7760
0070	7760	7760	7760	7760	7760	7760	7760	7760
0100	7760	7760	7760	7760	7760	7760	7760	7760
0110	7760	7760	7760	7760	7760	7760	7760	7760
0120	0150	7760	7760	7760	7760	7760	7760	7760
0130	7760	7760	7760	7760	7760	7760	7760	7760
0140	7760	7760	7760	7760	7760	7760	7760	7760
0150	7760	7760	7760	7760	7760	7760	7760	7760
0160	0110	7760	7760	7760	7760	7760	7760	7760
0170	7760	7760	7760	7760	7760	7760	7760	7760
0200	7760	7760	7760	7760	7760	7760	7760	7760
0210	7760	7760	7760	7760	7760	7760	7760	7760
0220	0050	7760	7760	7760	7760	7760	7760	7760
0230	7760	7760	7760	7760	7760	7760	7760	7760
0240	7760	7760	7760	7760	7760	7760	7760	7760
0250	7760	7760	7760	7760	7760	7760	7760	7760
0260	0040	7760	7760	7760	7760	7760	7760	7760
0270	7760	7760	7760	7760	7760	7760	7760	7760
0300	7760	7760	7760	7760	7760	7760	7760	7760
0310	7760	7760	7760	7760	7760	7760	7760	7760
0320	0010	7760	7760	7760	7760	7760	7760	7760
0330	7760	7760	7760	7760	7760	7760	7760	7760
0340	7760	7760	7760	7760	7760	7760	7760	7760
0350	7760	7760	7760	7760	7760	7760	7760	7760
0360	0000	7760	7760	7760	7760	7760	7760	7760
0370	7760	7760	7760	7760	7760	7760	7760	7760
0410	7760	7760	7760	7760	7760	7760	7760	7760
0420	7760	7760	7760	7760	7760	7760	7760	7760
0430	7760	7760	7760	7760	7760	7760	7760	7760
0440	7760	7760	7760	7760	7760	7760	7760	7760
0450	7760	7760	7760	7760	7760	7760	7760	7760
0460	7740	7760	7760	7760	7760	7760	7760	7760
0470	7760	7760	7760	7760	7760	7760	7760	7760
0500	7760	7760	7760	7760	7760	7760	7760	7760
0510	7760	7760	7760	7760	7760	7760	7760	7760
0520	7720	7760	7760	7760	7760	7760	7760	7760
0530	7760	7760	7760	7760	7760	7760	7760	7760
0540	7760	7760	7760	7760	7760	7760	7760	7760
0550	7760	7760	7760	7760	7760	7760	7760	7760
0560	7700	7760	7760	7760	7760	7760	7760	7760
0570	7760	7760	7760	7760	7760	7760	7760	7760
0600	7760	7760	7760	7760	7760	7760	7760	7760
0610	7760	7760	7760	7760	7760	7760	7760	7760
0620	7660	7760	7760	7760	7760	7760	7760	7760
0630	7760	7760	7760	7760	7760	7760	7760	7760
0640	7760	7760	7760	7760	7760	7760	7760	7760
0650	7760	7760	7760	7760	7760	7760	7760	7760
0660	7600	7760	7760	7760	7760	7760	7760	7760
0670	7760	7760	7760	7760	7760	7760	7760	7760
0700	7760	7760	7760	7760	7760	7760	7760	7760
0710	7760	7760	7760	7760	7760	7760	7760	7760
0720	7460	7760	7760	7760	7760	7760	7760	7760
0730	7760	7760	7760	7760	7760	7760	7760	7760
0740	7760	7760	7760	7760	7760	7760	7760	7760
0750	7760	7760	7760	7760	7760	7760	7760	7760
0760	6600	7760	7760	7760	7760	7760	7760	7760
0770	7760	7760	7760	7760	7760	7760	7760	7760

Den Umgang mit dem Simulationsprogramm zusammenfassend und aufgrund allgemeiner Ueberlegungen möchte ich die nachstehenden Schlussfolgerungen ziehen:

1. Die Maschine von Corinthios funktioniert. Schon ihre Software-Simulation ist ein guter und rascher Algorithmus, wobei allerdings der Speicherbedarf gegenüber konventionellen Prozeduren etwa doppelt so gross ist. Von der Hardware-Ausführung darf eine für unsere Ansprüche bestens geeignete Maschine erwartet werden.

2. Bei einer 8 bit-Version wird die Quantisierung störend. Es ist für einen Prototyp eines ausbaufähigen Systems nicht sinnvoll, das Herzstück des Prozessors minimal auszulegen. Ich schlage deshalb vor, in Registern, ROM und Festkommaarithmetik 12 bit-Wörter zu verwenden.Damit ist auch ein stetiger Uebergang zum Steuercomputer PDP-8/A (siehe Kapitel 3) sowie zu einem PDP-12 Laborcomputer gewährleistet. Man vergleiche hierzu die genaueren Untersuchungen in Abschnitt 2.3.
3. Die Festkomma-Arithmetik - wobei ich aus Gründen der Schnelligkeit und Einfachheit nichts Abweichendes empfehle - erzeugt bei N= 512 im Extremfall in den beiden (oder maximal drei) letzten bits Rechen- bzw. Quantisierungsfehler. Das iterative Rechenschema bzw. zyklische Durchschleusen ist für die Fehlergrösse der zureichende Grund. Damit stehen inklusive Vorzeichen 10 korrekte bits pro Wort im Frequenzbereich zur Verfügung. Der Quantisierungsfehler ist so unbedeutend. Die fehlerbehafteten Stellen können beim Ausgang in den Steuercomputer mit Hilfe einer Maske abgedeckt werden.

4. Die Ein- und Ausgangsregister sollten 512 komplexe Doppelwörter umfassen. Dabei sollten Zapfstellen nach je 64 bytes vorliegen. Bedenkt man, dass, wie Cooley zeigt (3), in einem komplexen Eingang der Länge N ein reeller Vektor der Län-

ge 2N eingelesen und korrekt transformiert werden kann, dann ist einzusehen, dass wir mit einem achtgeteilten 512er-Register Vektorlängen von 128 bis zu 1024 Punkten verarbeiten können. Diese Auswahl ist völlig ausreichend, da wir mit überlappenden Zeitfenstern mitteln wollen. Längere Vektoren verändern die Arithmetik nicht. Sie wirken sich nur in der Registerlänge und der Steuerlogik aus. Zusätzliche Register können nachträglich ohne weiteres angehängt werden. Die Steuerlogik kann schon im vornherein im Blick auf längere Register gebaut werden. Ich glaube aber nicht, dass dies eines Tages erwünscht sein wird.

5. Abschliessend sei noch einmal darauf hingewiesen, dass ein reines Parallelmultiplizierwerk angezeigt wäre. (Vgl. meine Gegenüberstellung im Anhang C). Die Multiplikation ist die wichtigste und aufwendigste Operation. Sie bestimmt den Zeitbedarf der Grundoperation und damit die Schnelligkeit des ganzen FFT-Prozessors. Im Falle eines andern Entscheides kompliziert sich die Steuerlogik ausserdem beträchtlich.

2. DIE FEHLERQUELLEN DER DISKRETEN FOURIERTRANSFORMATION

Dieses Kapitel gibt einen knappen Einblick in die wichtigsten Fehlerquellen der diskreten Fouriertransformation. Allgemeinen Ueberlegungen wird nur insoweit nachgegangen, als sie unsere angestrebte Realisierung beeinflussen. Besonderer Nachdruck liegt auf den Konsequenzen für unser Projekt, die am Ende der einzelnen Abschnitte zusammengefasst sind. Voraussetzungen, Definitionen, Zwischenschritte und Folgerungen sind im Bestreben nach übersichtlicher Darstellung und nicht nach lückenloser Verifikation zusammengestellt. Zur Verifikation benütze man die im Text angegebenen Quellen.

2.1. Beschränktes Zeitfenster (vgl. (3) und (10))

a) Definitionen:

Das Fundament zur Spektralanalyse des Elektroenzephalogramms ist die Definition eines Zufallsprozesses $\{X(t)\}$ als Quelle von Zufallsfunktionen mit statistischen Eigenschaften, die denjenigen des EEGs sehr nahe kommen. Die Gesamtheit dieser Zufallsfunktionen nennt man ein statistisches Ensemble.

Anstelle eines kontinunierlichen Zufallsprozesses betrachten wir sogleich die Sequenz $\{X(j)\}$, $j=0, \pm1, \pm2, \ldots$, welche ein Modell des (ideal) abgetasteten EEGs darstellen soll. Von dieser diskreten Zufallsvariable wird verlangt, dass sie stationär sei, d.h. ihre Erwartungswerte seien unabhängig vom Index j,

$$E\{X(j)\} = E(X), \qquad (1)$$

und es existiert eine Art innerer Zusammenhang des Signals, was sich im Moment zweiter Ordnung

$$\gamma(k) = E\{(X(j)-E(X)).(X(j+k)-E(X))\} \qquad (2)$$
$$= E\{X(j)X(j+k)\} - E^2(X)$$
$$j = 0,\pm1,\pm2,\ldots \qquad k = 0,\pm1,\pm2,\ldots$$

darin zeigt, dass $\gamma(k) \neq 0$ ist für $k \neq 0$. k ist der Abstand des Wertepaares. $\gamma(0)$ ist die Varianz. $\gamma(k)$ heisst auch Autokovarianz. Stationarität heisst also, dass die Autokovarianz nur von k, nicht von j abhängt.

Man bezeichnet als Spektraldichte die Funktion

$$p(\omega) = \frac{1}{2\pi} \sum_{k=-\infty}^{+\infty} \gamma(k) e^{-ik\omega}, \quad (-\pi \leq \omega \leq \pi)$$
$$= \frac{1}{2\pi} \left(\gamma(0) + 2\sum_{k=1}^{\infty} \gamma(k)\cos(k\omega)\right) , \qquad (3)$$

letzteres weil $\gamma(k)=\gamma(-k)$. $p(\omega)$ oder besser $\hat{p}(e^{i\omega})$ ist die z-Transformierte der Sequenz $\gamma(k)$ auf dem Einheitskreis. In unserem Fall, wo X(j) die diskrete Darstellung einer kontinuierlichen Zeitfunktion ist, dürfen wir $p(\omega)$ auch als Fouriertransformierte von $\gamma(k)$ auffassen.

Uns interessiert der Fall, wo aus einer endlichen Sequenz X(0), X(1),...,X(m-1) sowohl Autokovarianz als auch Spektraldichte näherungsweise berechnet (geschätzt) werden können. Wir sprechen von einem Zeitfenster der Länge m. Es gelte ausserdem $m=2^{\mu}$, μ ganz.

b) Schätzung der Autokovarianz

Der Schätzwert des wahren Mittelwertes E(X) ist

$$\bar{X} = \frac{1}{m} \sum_{j=0}^{m-1} X(j) \; . \qquad (4)$$

Dies ist ein guter Schätzwert, denn er ist, gegenüber E(X), nicht

mit einem systematischen Fehler behaftet, er ist "unbiased"[1]:

$$E(\bar{X}) = \frac{1}{m} \sum_{j=0}^{m-1} E(X(j)) = E(X) \ .$$

Als Schätzfolge für $\gamma(k)$ nimmt man

$$\begin{aligned} \bar{\gamma}(k) &= \frac{1}{m-k} \sum_{j=0}^{m-k} X(j)X(j+k) - (\bar{X})^2 \\ &= \bar{\gamma}^{0}(k) - (\bar{X})^2 \ . \end{aligned} \tag{5}$$

Man vergleiche das mit (2)! Im Fall eines Null-Mittelwertes $E(X)=0$ ist die Schätzfolge sogar $\bar{\gamma}^{0}(k)$. Man spricht von einer Unbiased-Schätzung von $\gamma(k)$:

$$E(\bar{\gamma}(k)) = E(\bar{\gamma}^{0}(k)) = \frac{1}{m-k} ((m-k)\gamma(k)) = \gamma(k) \ . \tag{6}$$

c) Schätzung der Spektraldichte - das Periodogramm

Definition der endlichen diskreten Fouriertransformation.

$$A(n) = \frac{1}{m} \sum_{j=0}^{m-1} X(j) e^{-i\frac{2\pi jn}{m}} \ , \ n=0,1,\ldots,m-1 \ . \tag{7}$$

Definition des Periodogramms:

$$I(n) = \frac{m}{2\pi} |A(n)|^2 = \frac{m}{2\pi} A(n)\tilde{A}(n) \tag{8}$$

wobei $\tilde{A}(n)$ das Konjugiert-Komplexe von $A(n)$ ist.

---

[1] biased (engl.) = mit einer systematischen Abweichung behaftet. Ich benütze, mangels eines deutschen, im folgenden den englischen Ausdruck.

I(n) ist gerade, d.h. I(n) = I(m-n), falls X(j) Observablen (reell) sind. Dann ist auch I(m/2) = 0.

Man kann deshalb definieren

$$I_+(n) = 2I(n) \ , \ n=0,1,\ldots, \ m/2-1 \ , \tag{9}$$

womit gezeigt werden kann, dass unter der Voraussetzung E(X)=0 gilt

$$I_+(n) = \frac{1}{\pi} \ (\bar{\gamma}^0(0)+2 \sum_{k=1}^{m-1} \bar{\gamma}^0(k)\cos(\frac{2\pi nk}{m})) \ . \tag{10}$$

Da $p(\omega)$ gemäss (3) in ω symmetrisch ist, kann man die <u>einseitige Spektraldichte</u> $p_+(\omega)$ definieren:

$$p_+(\omega) = \frac{1}{\pi}(\gamma(0)+2 \sum_{k=1}^{\infty} \gamma(k)\cos(k\omega)), \ 0 \leq \omega \leq \pi \tag{11}$$

Nun weiss man, dass $\bar{\gamma}^0(k)$ gegen $\gamma(k)$ konvergiert für $m \to \infty$. Die Vermutung liegt nahe, dass $I_+(n)$ gegen $p_+(\omega)$ konvergiert an den Stellen $\omega=\omega_n=\frac{2\pi n}{m}$ für $m \to \infty$. Dies trifft aber nicht zu! $I_+(n)$ ist zwar eine asymptotische Unbiased-Schätzfolge von $p_+(\omega)$, aber <u>keine konsistente</u>, d.h. die Varianz $E\{(I_+(n)-p_+(\omega_n))^2\}$ geht nicht gegen Null für grosse m, ja sie hängt gar nicht von m ab.

Das ist nachstehend erläutert für den Fall von <u>weissem Gauss-Rauschen</u>. Für dieses sind die X(j) normal verteilt und wechselseitig unkorreliert. Die Autokovarianz verschwindet also, $\gamma(k)=0$ für $k \neq 0$, und mit (3) wird die Spektraldichte $p_+(\omega) = \frac{\gamma(0)}{\pi} = \text{const.}, 0 \leq \omega \leq \pi$, eine Konstante.

Einige Ueberlegungen zeigen aber, dass das Periodogramm in diesem Fall für $n \neq 0$ eine Chi-Quadrat-Verteilung mit zwei Freiheitsgraden (Real- und Imaginärteil sind unabhängig normal verteilt!) ist:

$$\text{Prob} \ (I_+(n) > y) = \exp(-y\pi/\gamma(0)) \ . \tag{12}$$

Mithin sind Erwartungswert und Varianz die Konstanten,

$$E(I_+(n)) = \frac{\gamma(0)}{\pi}, \quad var(I_+(n)) = \frac{\gamma^2(0)}{\pi^2} \triangleq p^2(\omega_n) \tag{13}$$

und die Punkte sind wechselseitig unkorreliert

$$corr\,(I_+(n_1)I_+(n_2)) = 0\ . \tag{14}$$

$I_+(0)$, der einer Chi-Quadrat-Verteilung mit einem Freiheitsgrad folgt, ist der einzige Punkt, der durch einen endlichen Mittelwert $E(X(j)) \neq 0$ beeinflusst wird. In diesem Fall ist

$$E(I_+(0)) = \frac{\gamma(0)}{\pi} + \frac{mE^2(X(j))}{\pi} \tag{15}$$

Das Periodogramm $I_+(n)$ ist im Fall des weissen Rauschens demnach keine gute Schätzung der Spektraldichte $p(\omega)$.

d) Konsistente Schätzung der Spektraldichte

Zur Verbesserung der Stabilität des Periodogramms gibt es drei verschiedene Verfahrensweisen:

I) Der traditionelle Umweg über die verkürzte und abgedämpfte Autokovarianz mit anschliessender diskreter Fouriertransformation.

II) Glättung des Periodogramms, welches aus der zusammenhängenden Abtastfolge X(j) hervorgeht.

III) Sektionierung der Abtastfolge X(j), Berechnung des Periodogramms für jeden Abschnitt, Mittelung dieser Kurz-Periodogramme.

Alle drei Methoden führen zum Varianz- oder Leistungsspektrum als konsistente Schätzfunktion der Spektraldichte. Währenddem I) und II) auf dieselbe Schätzfunktion führen, erzielt man mit III) ein leicht abweichendes Resultat.

Die Methode I) stammt aus der Zeit, wo die Fouriertransformation mit einem bedeutenden Aufwand verbunden war, denn es werden bloss soviele Punkte transformiert, wie im Leistungsspektrum tatsächlich auftreten. Seit der Einführung des FFT-Algorithmus wird sie kaum mehr verwendet, da sie nur ganz spezielle Fensterfunktionen zulässt (Bartlett- und Hanning-window).

Die Methode II) ist die seit der FFT-Revolution meistgebrauchte und sei hier ganz knapp skizziert. Man glättet das Periodogramm durch gewichtete Mittelung über benachbarte Werte I(n),

$$\bar{I}_r(n) = \sum_{p=-k}^{k} H_r(p) I(n-p) \quad . \tag{16}$$

$H_r(p)$ ist die zu bestimmende Gewichtssequenz. Im Fall von weissem Gauss-Rauschen gilt mit (16)

$$E(\bar{I}_r(n)) = \frac{\gamma(0)}{\pi} \sum_{p=-k}^{k} H_r(p) \tag{17}$$

und

$$\mathrm{var}(\bar{I}_r(n)) = \frac{\gamma^2(0)}{\pi^2} \sum_{p=-k}^{k} H_r{}^2(p) \tag{18}$$

(vgl. mit (13)!)

Damit die Schätzung unbiased wird, muss die Summe in (17) Eins werden. Unter dieser Bedingung kann man (18) für gegebene "Bandbreite" 2k minimalisieren und erhält so

$$H_0(p) = \begin{cases} \frac{1}{2k+1} & (-k \leq p \leq k) \\ 0 & \text{sonst} \end{cases} \tag{19}$$

Das führt auf die Varianz

$$\mathrm{var}(\bar{I}_0(n)) = \frac{\gamma^2(0)}{\pi^2(2k+1)} \tag{20}$$

die gegenüber (13) reduziert ist. Es gibt aber so nur noch

$\frac{m}{2(2k+1)}$ unabhängige Schätzwerte von $p(\omega_n)$, und benachbarte Spektrallinien sind <u>korreliert</u>.

Die <u>Methode III)</u> schliesslich ist die für unser Vorhaben weitaus interessanteste. Die Transformation kurzer Sequenzen kann auf einer Maschine mit kleinem Speicherumfang durchgeführt werden. Deshalb ist die Methode wie geschaffen für Hardware-Prozessoren. Noch wichtiger aber ist, dass III) auf Instationarität sensibel ist, sodass eine <u>spektrale Erfassung transienter Vorgänge</u> ins Auge gefasst werden kann. Diese Möglichkeit ist noch nicht völlig erforscht.

Die Zeitreihe $X(j)$, $j=0,1,\ldots, m-1$, mit $E(X) = 0$ wird folgendermassen in K <u>überlappende Segmente</u> zerlegt:

$$\begin{aligned} X_1(j) &= X(j), \quad j=0,\ldots,L-1 \\ X_2(j) &= X(j+D), \quad j=0,\ldots,L-1 \\ &\vdots \\ X_K(j) &= X(j+(k-1)D), \quad j=0,\ldots,L-1 \end{aligned} \tag{21}$$

Diese Segmente sollen die ganze Zeitreihe überdecken, d.h. $(K-1)D+L=m$ (vgl. Fig. 24).

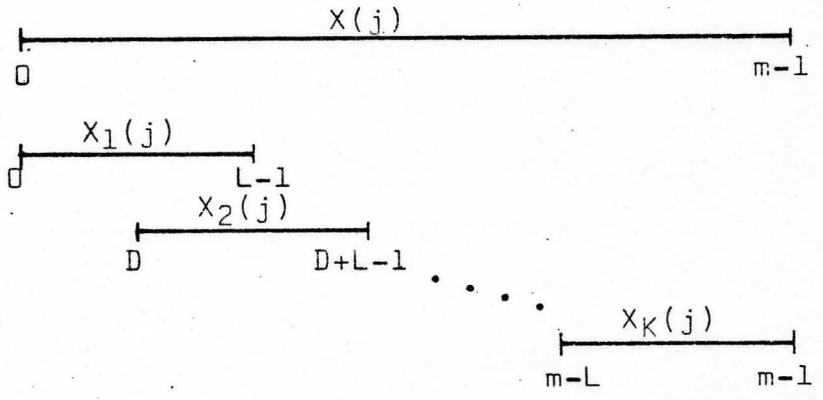


Fig. 24 Darstellung der Sektionierung der Zeitreihe X(j) in K überlappende Segmente.

Wir nehmen an, L sei eine Potenz von 2, z.B. 256 oder 512, was einem Zeitfenster von 2 bzw. 4 sec entsprechen wird.

Ein solcher "Ausschnitt" aus der Zeitreihe X(j) kann auch als Produkt der Zeitreihe mit einer Fensterfunktion W(j) aufgefasst werden,

$$X(j)W(j) \xleftarrow{DFT} A(n)*\hat{W}(n)=A^*(n) \qquad (22)$$

Frequenzseitig entspricht dem Produkt die Faltung der fouriertransformierten Faktoren. Für ein Rechteckfenster ist $\hat{W}$ eine sin x/x-Funktion, deren Seitenbuckel nicht besonders rasch konvergieren, was eine starke Korrelation benachbarter Spektrallinien bewirkt (="leakage"). Die Randabdämpfung der Eingangssequenz mit einer modifizierten Fensterfunktion (z.B. Dreieckfenster) verbessert die Konvergenz des "Spektralfensters" und verringert entsprechend die Korrelation. Allerdings leidet darunter die Stabilität der Schätzung.Durch Ueberlappung der Zeitepochen und Mittelung über sukzessive Spektren kann dem entgegengewirkt werden. Die Kunst besteht in der Optimierung dieses Kompromisses, wozu neben theoretischen Erwägungen eine gehörige Portion praktische Erfahrung gehört.

Wir wählen also ein Datenfenster W(j), j=0,..., L-1 und bilden die Sequenzen $X_1(j)\cdot W(j),\ldots, X_K(j)\cdot W(j)$. Daraus entstehen via FFT die modifizierten rohen Spektren $A_1^*(n),\ldots, A_K^*(n)$ und die modifizierten Periodogramme

$$I_k^*(\omega_n) = \frac{L}{2\pi U} |A_k^*(n)|^2 , \quad k = 1,2,\ldots, K . \qquad (23)$$

mit $\omega_n = \frac{2\pi n}{L}$, n = 0,..., L/2

$$\text{und } U = \frac{1}{L}\sum_{j=0}^{L-1} W^2(j). \tag{24}$$

Für ein modifiziertes Periodogramm ist der Erwartungswert

$$E(I_k^*(\omega_n)) = \int_{-\pi}^{\pi} h^*(\omega)p(\omega-\omega_n)d\omega\ , \tag{25}$$

wobei
$$h^*(\omega) = \frac{1}{2\pi LU}\left|\sum_{j=0}^{L-1} W(j)e^{i\omega j}\right|^2 \tag{26}$$

mit
$$\int_{-\pi}^{\pi} h^*(\omega)d\omega=1 \tag{27}$$

die zu W(j) gehörige, englisch "basic spectral window" genannte Funktion ist. Man sieht in (26), dass $h^*(\omega)$ die erste Nullstelle für $\omega=\frac{2\pi}{L}$ annimmt.

Die spektrale Schätzfunktion erhält man durch Mittelung über die K Periodogramme

$$\hat{p}\ (\omega_n) = \frac{1}{K}\sum_{k=1}^{K} I_k^*(\omega_n)\ . \tag{28}$$

mit Schätzpunkten im Abstand $\frac{2\pi}{L}$. Unter der Voraussetzung, dass X(j) ein Gauss-Prozess ist und die Spektraldichte $p(\omega)$ flach ist innerhalb des Schätzwertes, erhält man gemäss (3) folgende Variabilität:

i) D=L (keine Ueberlappung) und Rechteckfenster:

$$\text{var}(\hat{p}(\omega_n)) = \frac{p^2(\omega_n)}{K} = \frac{p^2(\omega_n)\ .\ L}{m} \tag{29}$$

ii) D=L/2 (halbe Ueberlappung) und Parzenfenster:

$$\text{var}(\hat{p}_1(\omega_n)) = \frac{p^2(\omega_n)}{K}\left(1+\frac{2}{9}-\frac{2}{9K}\right) \approx \frac{11p^2(\omega_n)}{9K} \qquad (30)$$

Da K aber doppelt so gross ist, wie in (29), hat man in ii) eine um den Faktor 11/18 verbesserte Varianz gegenüber i). Diese Werte gelten für $\omega_n \neq 0$ und π, für welche Randpunkte die Varianz doppelt so gross ist. Der Vergleich mit (17)-(20) zeigt, dass II) und III) punkto Varianz etwa gleichwertig sind.

Das Parzenfenster hat die analytische Form

$$W_1(j) = 1 - \left| \frac{j - \frac{L-1}{2}}{\frac{L+1}{2}} \right|^2 , \; j=0,1,\ldots,L-1 , \qquad (31)$$

ist also eine Dreieck-Funktion. Dazu gehört der Spektralpartner

$$h_1^*(\omega) \approx \frac{2\pi}{LU}\left\{\frac{L+1}{\pi}\,\frac{\sin^2((L+1)\omega/4)}{((L+1)\omega/4)^2}\right\}^2 \qquad (32)$$

eine besonders rasch konvergierende Glocke mit verschwindenden Seitenbuckeln.

Falls die Daten einen endlichen Mittelwert $\bar{X} \neq 0$ besitzen, muss dieser vor der Verarbeitung subtrahiert werden. Die Spektren nach Methode II) erscheinen selbst nach rigoroser Glättung ziemlich lebhaft, da nach (16) die Linienanzahl m/2 des Periodogramms nicht reduziert wird. Bei III) hingegen arbeitet man mit der beschränkten Linienzahl L/2, so dass eine glatte Kurve entsteht, die dem Informationsgehalt besser entspricht.

e) Folgerungen

Für einen festverdrahteten EEG-Prozessor mit beschränktem Speicherumfang ist die Methode der Sektionierung und Ueberlappung einer längeren Zeitreihe die günstigste. Sie ist als einzige auf Instationarität sensibel und gibt ebenso gute Schätzwerte wie die anderen

beiden Verfahrensweisen. Die halbe Ueberlappung unter Verwendung eines Parzen-Fensters ist zumindest ein guter Ausgangspunkt. Bezüglich Instationarität und Transientenanalyse kann nur praktische Erfahrung den Weg zur besten Strategie weisen. Es wäre deshalb günstig, von Vektorlängen von 512 Punkten ausgehend, auch möglichst kurze Sektionen transformieren zu können. Die Schieberegister sollten mithin möglichst viele (z.B. 8) Anzapfungen aufweisen.

2.2. Abtastung (vgl. (17))

a) Theorie:

Ideales Abtasten heisst anschaulich, ein kontinuierliches Signal $x(t)$ nur zu bestimmten Zeitpunkten zu beobachten. Wir nehmen hier an, dies könne mit beliebiger Genauigkeit geschehen. Abtasten heisst mathematisch, das kontinuierliche Signal mit einem "Delta-Kamm" $g(t)=\sum_{k=-\infty}^{\infty}\delta(t-k\Delta t)$ zu multiplizieren, wobei $\delta(x)=1$ ist falls $x=0$, sonst ist $\delta(x)=0$. So entsteht eine abgetastete Funktion

$$x^*(t) = x(t)g(t) = \sum_{k=-\infty}^{\infty} x(\Delta t)\ \delta(t-k\Delta t)\ . \qquad (33)$$

Wir haben drei Zeitfunktionen $x^*(t)$, $x(t)$, $g(t)$, für welche die Fouriertransformierten existieren und $X^*(f)$, $X(f)$, $G(f)$ genannt werden. Das Produkt (33) geht über in eine Faltung

$$X^*(f) = X(f).G(f)$$

$$= \int_{-\infty}^{\infty} X(\phi)\ G(f-\phi)\ d\phi\ . \qquad (34)$$

Die Fouriertransformierte $G(f)$ ist wiederum ein "Delta-Kamm" mit Impulsen der Höhe Δt Abstand $1/\Delta t$. Die Faltung bewirkt eine "Ausblendung" der Spektren $X(f)$ an den Stellen $n/\Delta t$, $n = 0,\pm1,\pm2,\ldots$ (vgl. Fig. 25).

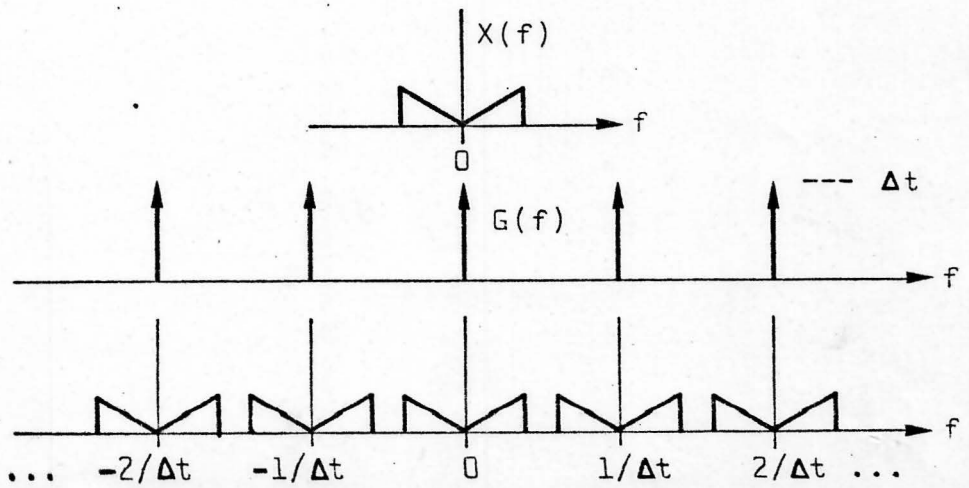


Fig. 25 Illustration der Abtastung im Frequenzbereich

$x(t)$ kann aus der diskreten Repräsentation $x^*(t)$ wiedergewonnen werden, wenn man letztere durch einen Tiefpass $H_0(f)$ laufen lässt, für welchen gilt

$$H_0(f) = \begin{cases} 1/\Delta t & |f| \leq W/2 \\ 0 & \text{sonst} \end{cases} \qquad (35)$$

mit $W<1/\Delta t$. Mit andern Worten, falls für das Abtastintervall gilt

$$\Delta t < \frac{1}{W} \qquad (36)$$

und das Signal $X(f)$ bandbegrenzt ist,

$$X(f) = 0 \text{ für } |f| < W/2 \,, \qquad (37)$$

ist die Abtastung fehlerfrei. Im andern Fall entstehen durch Ueberlagerung der Spektren in Fig. 25 Spiegelfrequenzen ("aliasing"), welche von den richtigen Frequenzkomponenten nicht unterschieden werden können. Man nennt

$$f_N = 1/2\Delta t \qquad (38)$$

die Nyquistfrequenz, welche die Grenzfrequenz des Spektrums $X(f)$ bezeichnet. Da es keine ideale Filter $H_0(f)$ gibt, muss der Filter-Cutoff wesentlich unterhalb der Nyquistfrequenz liegen, damit bei f_N ausreichend gedämpft wird.

b) Folgerungen:

Für EEGs erscheint die Wahl einer Abtastfrequenz f_t=128 Hz vernünftig. Die Nyquistfrequenz liegt dann bei 64 Hz, so dass der Cutoff-Punkt des Tiefpasses bei etwa 50 Hz gewählt werden kann. EEG-Spektren werden höchstens bis 50 Hz aufgezeichnet. Ein Eingangsvektor aus 512 Punkten entspricht einem 4 Sekunden-Stück. Das

bringt eine sehr feine Frequenzauflösung von 0,25 Hz. Die Frage der Anti-Aliasing-Filtrierung und der A/D-Konversion braucht uns vorläufig nicht zu kümmern, da die Datensammlung vom EEG-Labor mit Hilfe des PDP-12-Computers durchgeführt wird.

2.3. Quantisierungsfehler

a) Zahldarstellung

"For fixed-point arithmetic, ist is natural in a signal processing context to consider a register as representing a fixed point fraction."

Alan V. Oppenheim in (20)

Das Produkt zweier b-stelliger Zahlen hat 2b Stellen. Da die Multiplikation bei FFT iteriert wird, kann aber nur mit b Stellen weitergerechnet werden. Daher ist es nötig, b Stellen abzuwerfen.

Dieser Prozess lässt sich am leichtesten durchführen, wenn alle verwendeten Zahlen kleiner als Eins sind, das heisst, wenn der Binärpunkt vor der höchsten Stelle steht. Das Produkt ist dann immer kleiner als die Faktoren und überschreitet somit die Kapazität der Maschine nicht.

Damit ist die Zahldarstellung festgelegt. Sowohl bei der Simulation als auch in der verdrahteten Maschine verwendet man mit Vorteil Bruchzahlen. Die Vorzeichenstelle steht links vom Binärpunkt. Die Darstellung der negativen Zahlen ist aber damit noch nicht entschieden; man hat die Wahl zwischen Absolutwert mit Vorzeichen, Einerkomplement und Zweierkomplement. Dazu möge wiederum Oppenheim zu Worte kommen:

"The choice of representation for negative numbers in a particular system is usually based almost entirely on hardware considerations."

Damit stehen wir vor einem Entscheid, der nicht durch Simulation unterstützt wird. Wir stehen vor der Frage der Detailstruktur der Arithmetik und der Einbettung des Prozessors in den übergeordneten Computer. Zahldarstellung und Hardwarestruktur beeinflussen sich

wechselseitig, so dass nicht das eine ohne das andere diskutiert werden kann. Der Leser, welcher im folgenden die logische Rechtfertigung einzelner Schlüsse vermissen könnte, möge nicht übersehen, dass die Intuition bei der Entwicklung komplexer Systeme eine ausschlaggebende Rolle spielt. Es gibt dann aber nur eine Rechtfertigung a posteriori (d.h. durch die Maschine selber), die nicht in dieser Arbeit erbracht werden kann.

Folgende Umstände sind bei der Entscheidung zur Darstellung negativer Zahlen zu berücksichtigen (vgl. auch (23) und (24)):

- Das System PDP-8/A, das, wie im nächsten Kapitel gezeigt ist, mit Vorteil als übergeordneter Steuercomputer verwendet werden kann, arbeitet im Zweierkomplement. Die Wortlänge ist 12 Bit.

- Im Einerkomplement gewinnt man die konegative Zahl dadurch, dass man alle 1 in 0 vertauscht und umgekehrt, also durch einfache Negation.

- Addition und Subtraktion sind im Zweierkomplement am einfachsten, da der Uebertrag aus Bit 0 fallengelassen wird. Da diese Operationen bei FFT vergleichsweise wenig Zeit beanspruchen, ist auch das Einerkomplement vorteilhaft. Ein algebraisches Addierwerk für Zahlen mit Vorzeichen und Absolutwert ist hingegen sehr kompliziert, da vor und nach der Operation eine Komplementwandlung nötig wäre, was den Aufwand verdreifacht.

- Wichtiger für den Entscheid ist die Ausgestaltung des Multiplizierwerks. Multiplikation mit Absolutwerten ist am einfachsten. Bei rein parallelen Werken entfällt dann jede Ablaufsteuerung. Für konegative Darstellung ist die Operation wesentlich komplizierter.

Unter Berücksichtigung aller Interdependenzen gelange ich zu folgenden Schlüssen:

- Für den Prozessor gilt - mit Ausnahme des Multiplizierwerks - die Darstellung im Zweierkomplement. Der Verkehr mit dem PDP-8/A-Computer wird so möglichst vereinfacht. Die Additions-Arithmetik ist problemlos. Die Subtraktion reduziert sich, nach vorangehender Komplementierung, auf eine Addition. Das iterative scaling ist, je nach dem Vorzeichen, leicht verschieden, aber so noch sehr einfach.

- Vor der Multiplikation werden die Operanden RS und IS in Absolutwerte mit Vorzeichen verwandelt. Das entspricht etwa dem Aufwand für eine Addition. Die Gewichte RW und IW sind schon in Absolutwertdarstellung gespeichert. Die reellen Produkte werden ins Zweierkomplement rückverwandelt und mit Hilfe einer Ablaufsteuerung und der parallelen **S**-Arithmetik derart kombiniert, dass ein komplexes Produkt im Zweierkomplement entsteht.

- Die Konzeption der Ablaufsteuerung sowie die Auswahl der Schaltungen für Addierwerk, Komplementwandler und Multiplizierwerk sollen demnächst untersucht werden. Es sind hierbei keine grossen Schwierigkeiten zu erwarten. Arbeitsunterlage wird hauptsächlich (23) sein.

b) Eingangsfehler (vgl. (17))

Wie schon aus Abschnitt 2.2. hervorgeht ist mit korrektem Abtasten kein Informationsverlust verbunden. Erst die Quantisierung, die mit jeder A/D-Konversion einhergeht, bringt Fehler. Sie sind mit den Fehlern verwandt, die, der beschränkten Wortlänge zufolge, in der Arithmetik entstehen. Weil aber in der Arithmetik zusätzliche Effekte auftreten, die vom Algorithmus abhängen, sei die Diskussion der Eingangsquantisierung vorangestellt.

Wir betrachten das ideale Zeitsignal X(f) und das durch die Quantisierungssprünge veränderte Zeitsignal $X_q(f)$, beide im Frequenzbereich (nach kontinuierlicher Fouriertransformation). Die Differenz beider ergibt das Störsignal.

$$Q(f) = X(f) - X_q(f) \tag{39}$$

welches als breitbandiges Rauschen aufzufassen ist. Es kann gezeigt werden, dass der mittlere quadratische Fehler

$$\overline{(X(f) - X_q(f))^2} = \overline{Q^2(f)} = P_q(f) \tag{40}$$

mit dem Quantisierungsschritt q durch

$$P_q(f) \approx q^2/12 \tag{41}$$

verknüpft ist. Für diesen gilt andererseits

$$q = 1/2^{b-1} , \tag{42}$$

wo b die Wortlänge inkl. Vorzeichen bedeutet. In Dezibel ist dieser Fehler deshalb

$$10.\lg P_q(f) \approx -6b - 4 , \tag{43}$$

Die 0 dB-Grenze entspricht der vollen Aussteuerung vom Betrage 1.

Der PDP-12-Computer besitzt einen 10-bit A/D-Konverter. Wir werden, wie oben erwähnt, vorläufig diesen Konverter benützen. Nach (43) werden wir also über einen Dynamikbereich von ungefähr 60 dB verfügen können. Für biogene Signale ist das mehr als genug!

c) Rechenfehler (vgl. (20) und (17))

Zur Diskussion des bei einem Festkomma-FFT-Prozessors zu erwartenden Rechenfehlers müssen drei Wortlängen auseinandergehalten werden:

- Die Datenwortlänge b_D
- Die Wortlänge der Gewichtsfaktoren b_W
- Die Wortlänge der Zwischenresultate b_T

Die Wortlängen verstehen sich inklusive Vorzeichenbit. Ausserdem ist festzuhalten, dass wir uns einer iterativen Scalingmethode bedienen (engl. automatic array scaling = AAS), die sich, verglichen mit einmaliger Skalierung, besonders günstig auf die Genauigkeit der fixed-point-Prozedur auswirkt.

Betrachten wir zunächst den FFT-Algorithmus ohne Skalierung. Die Grundoperation desselben wird in der Literatur "butterfly" genannt, dessen Zusammensetzung in unserem Falle

$$X_{m+1}(i) = X_m(i) + X_m(j)$$
$$X_{m+1}(j) = (X_m(i) - X_m(j))\, W \qquad (44)$$

ist. Bei Addition und Subtraktion müssen die Resultate nicht gekürzt werden, deshalb stellen sie keine zusätzlichen Rauschquellen dar. Anders die Multiplikation, bei welcher das genaue Resultat, wie eingangs erwähnt wurde, länger ist als die Länge der Operanden. Die Verkürzung auf die Wortlänge $b=b_T=b_W=b_D$ bedeutet eine erneute Quantisierung, stellt eine erneute unkorrelierte Rauschquelle dar. Zur komplexen Multiplikation tragen 4 reelle Multiplikationen bei, deren jede eine spektrale Rauschdichte von

$$P_q(f) \approx \frac{2^{-2(b-1)}}{12} \qquad (45)$$

darstellt (vgl. Gleichung (41)). Die Rauschvarianz für die komplexe

Multiplikation beträgt demgemäss

$$\sigma_B^2 = 4 \cdot \frac{2^{-2(b-1)}}{12} \tag{46}$$

Aus der Formel der diskreten Fouriertransformation

$$A(n) = \sum_{j=0}^{N-1} X(j)\, W_N^{-nj}, \quad W_N = \exp(2\pi i/N) \tag{47}$$

ist direkt ersichtlich, dass N erartige Rauschquellen auf jede einzelne Spektrallinie projizieren. Das ist bei FFT nicht anders, immer vorausgesetzt, dass keine Zwischenskalierungen vorgenommen werden. Die Rauschvarianz eines komplexen Ausgangselementes beträgt infolgedessen

$$\sigma_E^2 = N\sigma_B^2 = N\,\frac{2^{-2(b-1)}}{3} \tag{48}$$

* Komplizierter sind die Verhältnisse bei iterativer Skalierung (AAS), worunter also die arithmetische Verschiebung der Datenwörter $X_m(i)$ und $X_m(j)$ um eine Stelle nach rechts zu verstehen ist. Diese Massnahme verhindert overflow in der anschliessenden Addition bzw. Subtraktion. Um das zu zeigen, setzen wir voraus, dass für die Eingangsvektoren gilt $|X(j)| \leq 1$, und zwar für alle Elemente $X(j)$. Nach der Skalierung ist folglich $|X'(j)| \leq \frac{1}{2}$. Summe und Differenz kann die Beträge höchstens verdoppeln, $|S(j)| \leq 1$. Komplexe Multiplikation mit einem Einheitsvektor (vgl. Fig. 1) ändert diese Betragseigenschaft nicht, $|S(j).e^{i\phi}| = |S(j)| \cdot |e^{i\phi}| \leq 1$. Folglich wird bei der ersten Iteration der Bruchzahlenbereich nirgendwo überschritten, ja es gilt sogar strenger $|X_1(j)| \leq 1$. Das wiederholt sich also für alle Iterationsebenen und gilt mithin für das Resultat. Wichtig ist die Voraussetzung

$$|X(j)| \leq 1 \tag{49}$$

besonders für den Fall, wo nicht bloss die Realteile gefüllt werden,

wo z.B. zwei reelle Segmente zu einem komplexen vereinigt werden. Treten zufällig zwei Partner der Grösse Eins zusammen, $X(j) = 1+i$, dann wird der Betrag $|X(j)| = \sqrt{2} > 1$. Die strengere Voraussetzung (49) muss in diesen Fällen also sorgfältig berücksichtigt werden.

Zum Rechenfehler bei iterativer Skalierung können die folgenden Ueberlegungen gemacht werden. Es ist plausibel, dass das Rauschen (46), welches durch die Multiplikation entsteht, durch die nachfolgende Skalierung gedämpft wird. Andererseits stellt die Skalierung selber eine zusätzliche Rauschquelle dar, weil das letzte bit abgeworfen wird. Trotzdem entsteht, verglichen mit (48), ein viel günstigeres Resultat. Oppenheim (20) gibt als Rauschvarianz aus Multiplikation und Skalierung

$$\sigma_{B'}^2 = \frac{5}{6}\, 2^{-2(b-1)} \qquad (50)$$

an, was etwas höher als (46) ist. Die fortgesetzte Abschwächung dieser Varianzen bewirkt am Ausgang für grosse N die, verglichen mit (48), viel kleinere Rauschvarianz

$$\sigma_E^2 = 2\sigma_{B'}^2 = \frac{5}{3}\, 2^{-2(b-1)} \qquad (51)$$

Eine im Intervall (-a, a) gleichmässig verteilte Zufallsvariable Z besitzt die Varianz $\sigma_Z^2 = \frac{1}{3} a^2$. Auf (51) übertragen heisst das, dass am Ausgang ein Fehlerintervall $(-\sqrt{5}\cdot 2^{-(b-1)}, +\sqrt{5}\cdot 2^{-(b-1)})$ zu erwarten ist. Folglich dürfte sich der Fehler in den letzten zwei, höchstens drei bits auswirken. Wie ich im 1. Kapitel zeige, hat unsere Simulation dies bestätigt.

Die praktischen Konsequenzen derartiger Ueberlegungen sind bei Glisson et.al. (17) ausführlich beschrieben. Diese Gruppe misst den mittleren quadratischen Fehler mit einer Testanordnung Fig. 26.

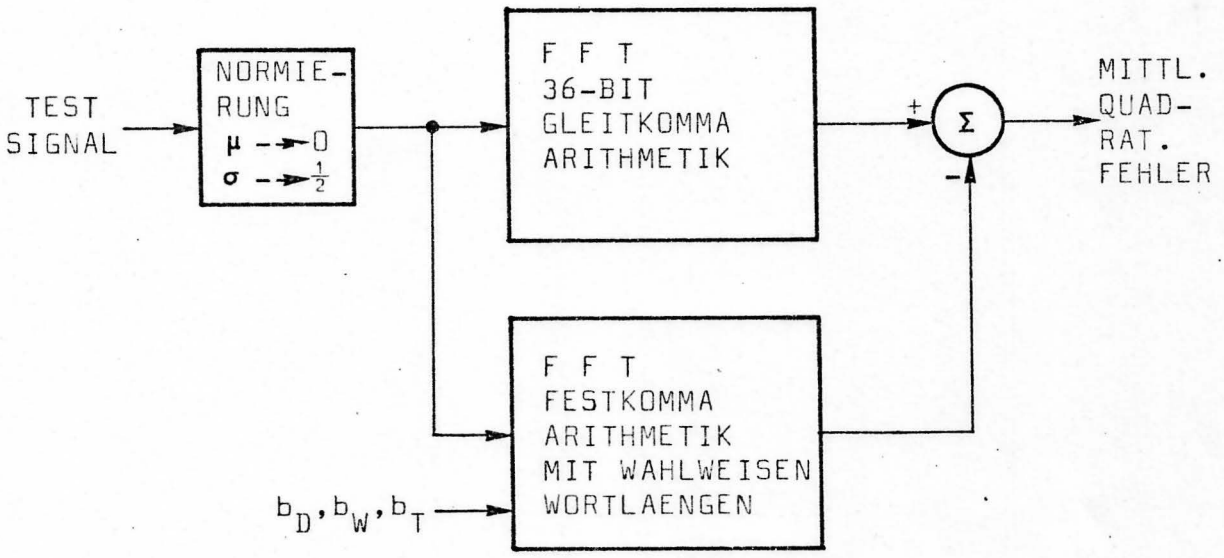


Fig. 26 Testanordnung zur Messung des mittleren quadratischen Fehlers bei Festkommaarithmetik.

Der Fehler wird ermittelt aus der Differenz einer quasiidealen schnellen Fouriertransformation und einem Festkomma-AAS-FFT, bei welchem die drei Wortlängen b_D, b_W, b_T gegenseitig variiert wurden. In etlichen Graphen können die Resultate abgelesen werden; damit wird es möglich, den Prozessor zu optimieren. In Fig. 26 wird am Eingang ein Testsignal mit der Standartabweichung $\sigma = 1/2$ verwendet.

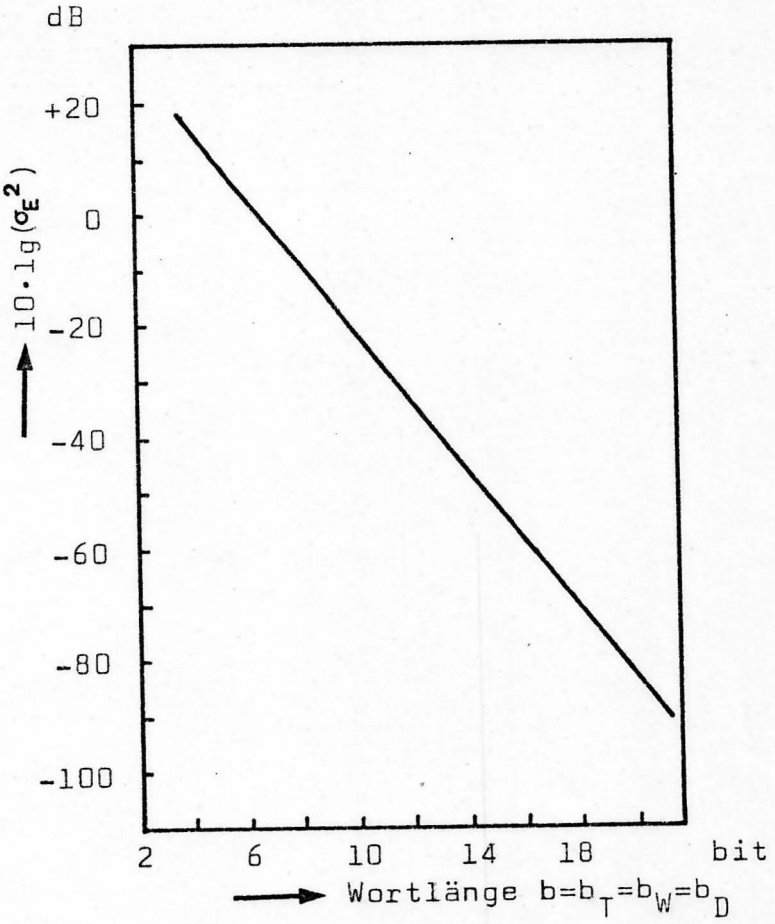


Fig. 27 Mittlerer quadratischer Fehler in Dezibel gegen Wortlänge

Zunächst treffen wir auf die Graphik Fig. 27, wo der Fehler in Dezibel gegen die Einheitswortlänge $b=b_T=b_W=b_D$ aufgetragen ist. Die Kurve stimmt zwar in der Steigung, nicht aber in der Lage mit (51) überein. Dieser Unterschied hängt womöglich damit zusammen, dass Oppenheim bei der Skalierung rundet, Glisson aber nicht. Es ist nicht nötig, ja mit Bezug auf Geschwindigkeit und Hardwareaufwand keineswegs optimal, $b_T=b_W=b_D$ zu wählen.

Zur Optimierung verwenden wir als erstes b_D= 10 bit. Soviel liefert uns der A/D-Konverter. Ausserdem zeigt Glisson, dass es nichts einbringt, b_W grösser als b_D zu machen. Schliesslich bringt Fig. 28 den gewünschten Zusammenhang zwischen b_T und dem mittleren quadratischen Fehler, unter der Bedingung $b_D=b_W$= 10 bit.

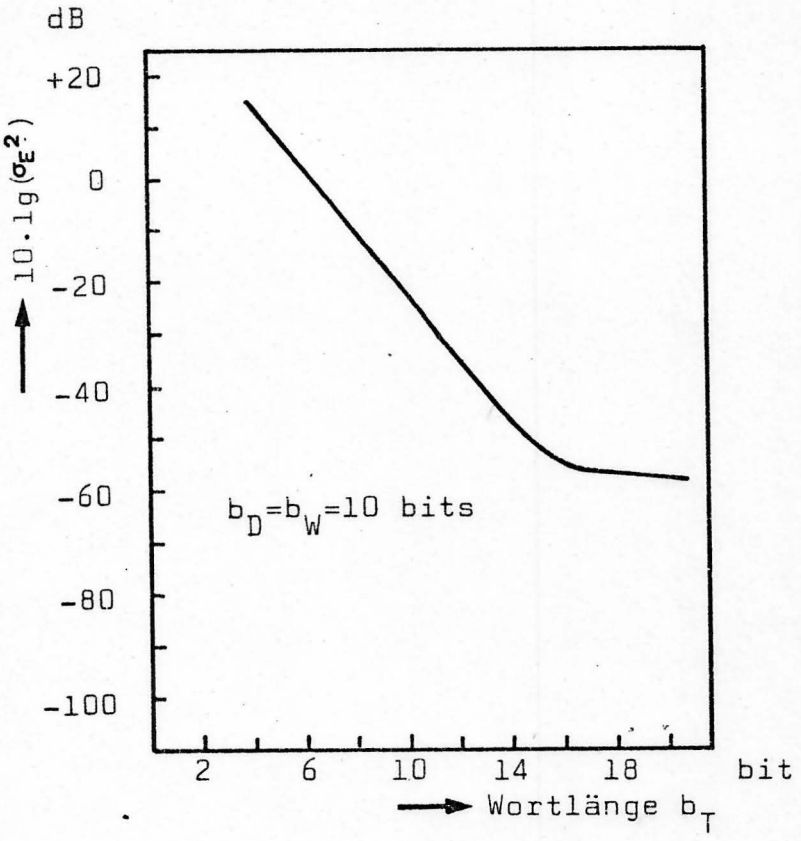


Fig. 28 Mittlerer quadratischer Fehler in Dezibel gegen Wortlänge b_T

Eine 8-bit-Zahl entspricht einerGenauigkeit von ca. $^1/_2$ %, was erfahrungsgemäss für biogene Signale völlig ausreicht. Der Quantisierungsschritt einer b-bit-Bruchzahl ist $q=1/2^{b-1}$, und der mittlere quadratische Fehler, der durch diesen Quantisierungsschritt entsteht, ist in Dezibel

$$10.\lg(q^2/12) = -6b -4 \ . \qquad (52)$$

Für b=8 beträgt er 50 dB. Gemäss Fig. 28 bringt die Wortlänge b_T=14 bit die gewünschte Genauigkeit.

Es bleibt nun allerdings fraglich, ob nicht bereits eine arithmetische Wortlänge b_T=12 bit unsere Genauigkeitsansprüche erfüllen könnte, wenn die Skalierung mit Rundung eingebaut würde. Aufgrund

meiner Simulationserfahrungen bin ich der Ansicht, dass 12 bit ausreicht. Dieser Punkt muss aber, vielleicht am besten mit Hilfe einer der Fig. 26 entsprechenden Testanordnung, noch abgeklärt werden.

d) Folgerungen

- Es gelten die Merkpunkte am Schluss von Abschnitt a).

- Die Wortlänge der Eingangssequenzen b_D und der Gewichtsfaktoren b_W beträgt je 10 bit.

- Die Wortlänge der Zwischenresultate b_T beträgt wahrscheinlich 12, höchstens aber 14 bit. Damit werden die ersten 8 bit genau. Der Rest kann vor dem Auslesen abgedeckt werden. Diese Wortlänge bestimmt die"Breite" der Ein- und Ausgansregister, sowie die Länge aller Zwischenregister in der Arithmetik. Insbesondere wird das Resultat der Multiplikation auf diese Länge gekürzt.

- Die Skalierung unter Rundung durchzuführen, ist möglicherweise vorteilhaft.

- Die komplexen Eingangswerte X(j) müssen der Bedingung (49) $|X(j)| \leq 1$ gehorchen. Nur so ist man sicher, dass im Zuge der Transformation der Bruchzahlenbereich nirgendwo gesprengt wird.

3. KONZEPT EINES EEG-PROZESSORS

Im Verlaufe unserer Umfrage (15) in medizinischen Forschungsstätten haben wir feststellen können, dass einem auf die spezifischen Bedürfnisse zugeschnittenen Echtzeit-Spektralanalysator von verschiedenen Seiten Interesse entgegengebracht wurde. In Zürich allein gibt es sechs Kliniken oder Institute, die alle mit kleineren Prozessrechnern ausgerüstet sind. Unser Projekt könnte - laut einem Gutachten von Herrn Prof. Angst - "ohne Zweifel gerade diesen Institutionen ... einen wesentlichen Ausbau der Datenverarbeitung ermöglichen". Wir konnten aber feststellen, dass die Ansprüche der verschiedenen Laboratorien an einen On-Line-Prozessor ziemlich voneinander abweichen, so dass es kaum möglich und sinnvoll wäre, schon jetzt allen potentiellen Benützern planerisch entgegenzukommen.Es führt rascher zum Ziel, uns zunächst nicht vom Prinzip der "Vollständigkeit", sondern von dem der "Vollkommenheit" leiten zu lassen, und in enger Verbindung mit dem analytisch fortgeschrittenen Institut (Kinderspital, EEG-Labor) eine dort brauchbare Maschine zu entwickeln. Wenn von Anfang an darauf geachtet wird, dem System Ausbau- und Anpassungsfähigkeit zu erhalten, dann ist es verhältnismässig leicht, den Umfang des Pflichtenheftes nachträglich zu vergrössern. Die Forderung "Ausbaufähigkeit" freilich beeinflusst die Entwicklung von Grund auf. Insbesondere werden wir den eigentlichen Fouriertransformator möglichst leistungsfähig konzipieren. Dennoch bleibt eine der wichtigsten Prämissen die, jede Kosteninflation zu vermeiden, und dem beschränkten Budget medizinischer Laboratorien Rechnung zu tragen.

Infolgedessen gilt es, ein Optimum zwischender Ausbau- und Anpassungsfähigkeit einerseits und der Preisgünstigkeit andererseits anzustreben.

3.1. Grundgedanken

Vom schnellen Fouriertransformator an und für sich bis zum mehrkanaligen Realtime-Prozessor gibt es mehr zu entscheiden und zu planen, als man im vornherein anzunehmen geneigt ist. Man denke z.B. an die A/D-Konversion, Zwischenspeicherung, Segmentierung, Ueberlappung, Gewichtung, Transferierung ins Eingangsregister im Zeitbereich, und im Frequenzbereich an die Mittelung der Spektren, Berechnung des Periodogramms, des Leistungsspektrums, eventueller Kreuzspektren, Kohärenzfunktionen, Peakstatistik und Parameterextraktion. Hinzu kommt die Ablaufsteuerung der drei Betriebsphasen des FFT-Prozessors selber, die Steuerung der digitalen Frequenzverformung nach Oppenheim, etc.

Diese Aufgaben sind so vielfältig und zahlreich, dass man nicht umhin kann, vom Gedanken des Selbstbaus abzurücken und sich zu fragen, ob nicht ein Kleincomputer all dies übernehmen könnte. Kleincomputer sind teuer; man schreckt zurück, schaut sich um und stösst auf die Microprozessoren. Bei aller Unvoreingenommenheit muss man aber feststellen, dass Microprozessoren mühsam zu programmieren und etwas langsam sind, einen mageren Befehlsschatz haben und ausserdem meist mit zu kleiner Wortlänge angeboten werden. Es ist nicht auszuschliessen, dass jenseits vom Prototyp dereinst Microcomputer unseren Bioprozessor überwachen könnten. Ihre Verwendung im jetzigen Aufbaustadium jedoch müsste uns zu sehr einschränken.

Von Herrn Dr. Dumermuth erhielt ich den Hinweis, dass die Firma Digital Equipment gegen Ende 1974 einen neuen äusserst preisgünstigen Minicomputer PDP-8/A aus der erfolgreichen PDP-8-Familie auf den Markt bringt. Er kann im Sinne des OEM-Agreements auf Einzelkarten bezogen werden und kostet dann minimal 572 Dollar; das ist die Preislage der Microprozessoren! Der PDP-8/A verfügt

über den vollen PDP-8-Befehlsschatz, RAM, ROM und PROM-Memories von 1 bis zu 32 K bytes, ein einzigartiges Interface-System namens OMNIBUS und 1,5/usec Zykluszeit. Die nachstehende Spezifikationsliste enthält hierüber noch einige zusätzliche Angaben.

PDP-8/A PRODUCT SPECIFICATIONS

Word Length:	12 bit
Cycle Time:	1.5 μs
Hardware Registors:	2 (AC and MQ)
Auto-Index Registors:	8 (Loc 10-17) Operating Times
Memory:	RAM 1K, 2K, 4K 2.3 μs ROM 1K, 2K, 4K 1.5 μs PROM 1K 3.4 μs
Memory Expansion:	Up to 32K (see Power Consumption Chart)
Input/Output:	Programmable and interrupt-driven I/O. Direct memory access via DATA BREAK for high-speed data transfers (667,000 words/sec).
Addressing Capability:	1 instruction may address 256 locations directly/4096 indirectly.
Instruction Set:	Same powerful instructions as the PDP-8/E.
Software:	Complete compatibility with the PDP-8 Family software to include OS/8, CAPS-8, RTS-8.

Instruction Execution	Add	3.0 μs
	AC—Mem.	3.0 μs
	Zero AC	1.5 μs
	Input DATA	1.5 μs
Options:	Two option configurations are available on hex boards.	
DKC8-AA (I/O Option Board)	KM8-A (Extended Option Board)	
Front panel control Serial line unit Parallel I/O Real-time clock	Power Fail/Auto Restart Memory Extension Bootstrap Loader	
Physical Size:	CPU on one hex board—15½ inches x 8½ inches Memory on one quad board—8½ inches x 10½ inches Box 10 inches x 10½ inches x 19 inches	
Operating Environment:	Ambient Temperature 5-50°C Relative Humidity 8-90% (non-condensing)	
Instruction Execution Time: (Assumes 1.5μs memory times)		

Dieser Minicomputer würde uns in die Lage versetzen, ohne Kosteninflation ein verhältnismässig autonomes, vielseitiges, leistungs- und ausbaufähiges EEG-Analyse-System zu entwickeln. Im Gegensatz zu einer Vollintegration der Hardware ins bestehende PDP-12-System am Kinderspital, hätten wir dann die Möglichkeit, ohne viel Mehrarbeit weitere Benützerkreise anzusprechen. Von den sechs medizinischen Forschungsstätten in Zürich besitzen deren vier PDP-Computer, mit welchen der PDP-8/A kompatibel ist. Diesem käme mithin nicht bloss die Aufgabe zu, die spezielle Hardware des EEG-Prozessors zu überwachen, sondern darüberhinaus wäre er auch das

geeignete Kontaktorgan zu den Prozesscomputern in den Laboratorien. Ohne diese Anlagen allzusehr zu belasten, könnte man so über die gesamte Peripherie (Teletype, Bildschirme, Band- und Plattenspeicher, A/D-Konverter, Plotter) verfügen, sobald das relativ bescheidene Problem des Datentransfers zum Hauptcomputer gelöst ist. Im Blick auf eine eventuelle spätere Kommerzialisierung muss noch betont werden, dass der PDP-8/A mit allen 60 PDP-8-Optionen und Peripheriegeräten via OMNIBUS direkt verbindbar ist. Wohl kaum ein anderes Produkt vereinigt auf sich alle diese Vorteile und Wahlfreiheiten, so dass ich die hier skizzierte "Einbettung" mit allem Nachdruck vertreten möchte.

In nachstehender Figur 28 ist das Grundkonzept graphisch veranschaulicht. Die Darstellung enthält die vorläufige, für den Prototyp geltende Anordnung.

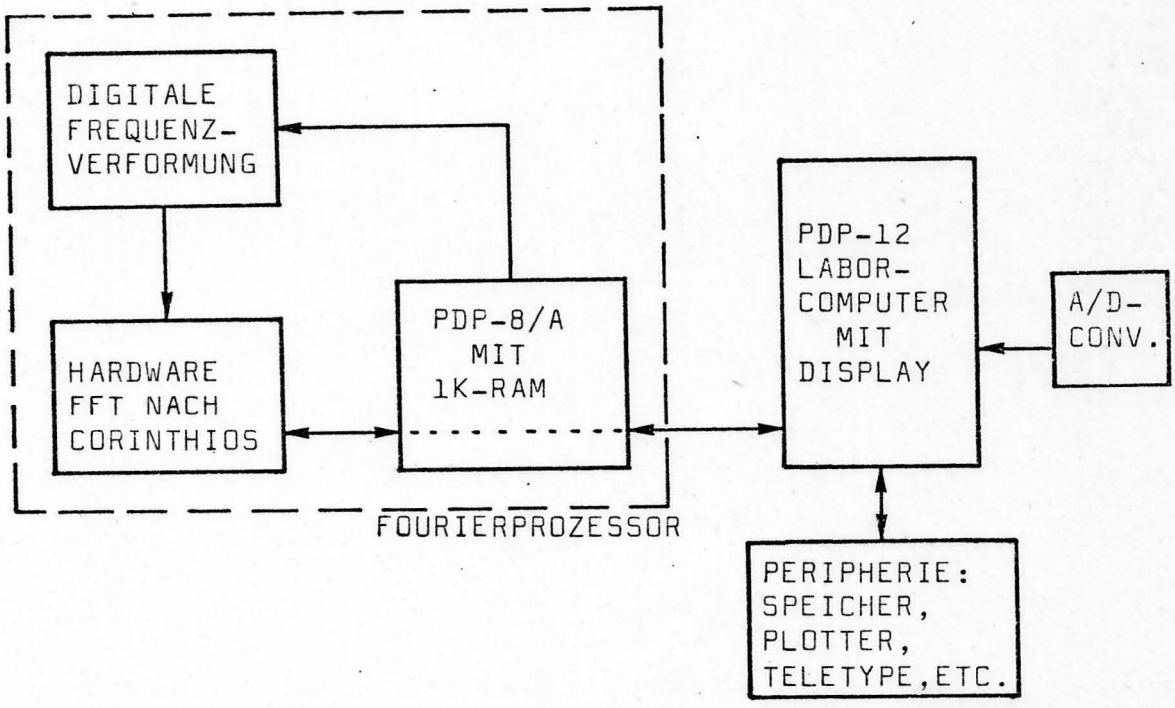


Fig. 28

Gesamtarchitektur des EEG-Analyse-Systems

Datenerfassung und A/D-Konversion leistet der PDP-12 in Verbindung mit dem leistungsfähigen DECpack-Disk-System.

Im PDP-12 werden auch die sich überlappenden Datensegmente gebildet. Ueber einen Interfacekanal mit ausreichender Transferrate laufen die Datenvektoren in den Fourierprozessor. Ob sie zuerst ins PDP-8/A-RAM gelesen werden oder direkt in die Schieberegister des FFT-Prozessors einlaufen, ist noch nicht abgeklärt. Im Corinthios-Prozessor oder jedenfalls mit dessen Multiplizierwerk werden die Segmente seitlich abgedämpft (windowing); die Gewichtsfaktoren stecken in einem speziellen ROM-Speicher. Anschliessend wird fouriertransformiert. Die Summierung der rohen Spektren, sowie eine allfällige Linearkombination derselben zur Entmischung eines komplexkombinierten Eingangs (siehe Anhang A , Seite159) sollte im PDP-8/A geschehen. Daselbst, evtl. mit Hilfe des FFT-Multiplizierwerks, erfolgt schlussendlich die Berechnung des glatten Periodogramms. Alle prozessorinternen Abläufe dirigiert der PDP-8/A, insbesondere auch den wahlweisen Umweg über die digitale Frequenzverformung. Das Varianzspektrum einer beliebigen Anzahl Segmente, oder allenfalls ein Kreuzspektrum,läuft in den PDP-12 zurück und wird dort gespeichert oder ausgegeben.

Es ist ohne weiteres erkennbar, dass im Echtzeitbetrieb, zumal bei grosser Ueberlappung und geringer Akkumulation der Einzelspektren (Transientendetektion!), von den Uebertragungskanälen hohe Durchflussraten gefordert werden müssen. Nehmen wir, zur Abschätzung derselben, einmal folgendes an:

Abtastfrequenz	128 Hz
Wortlänge	12 bit
Anzahl Kanäle	10
Ueberlappung	80 %

Damit erhalten wir eine Transferrate vom PDP-12 in den Fourierprozessor von 128.12.10.5 = 76'800 bits per Sekunde. Da ein Strom transformierter Daten wieder zurück geschleust wird und ausserdem Zeit für Checks und (bei zu geringer Bufferung) für Berechnungen gebraucht wird, ist eine Transferrate von 1 Million bits je Sekunde kein Lu-

xus.. Im nächsten Abschnitt diskutiere ich deshalb die 3 Varianten, welche Digital Equipment für den I/O-Transfer vorsieht, unter diesem Gesichtspunkt.

3.2. Die drei Methoden des Datenaustausches (vgl. (7), (8) und (5))

Zum PDP-8/A-Computer gibt es erst spärliche Informationen. Ich stütze mich im folgenden auf Unterlagen über den PDP-8/E-Computer, welche aber für die ganze PDP-8-Familie, repräsentativ sind.

a) "Programmed Data Transfer"

Der programmierte Datentransfer mit Hilfe eines Satzes sog. IOT-Instruktionen ist das einfachste, gebräuchlichste Mittel zur Datenübertragung zwischen einem externen Gerät und dem Computer. Fig. 29 a) erklärt den Aufbau einer IOT-Instruktion. Fig. 29 b) dient zur Illustration der nachfolgenden Erläuterungen.

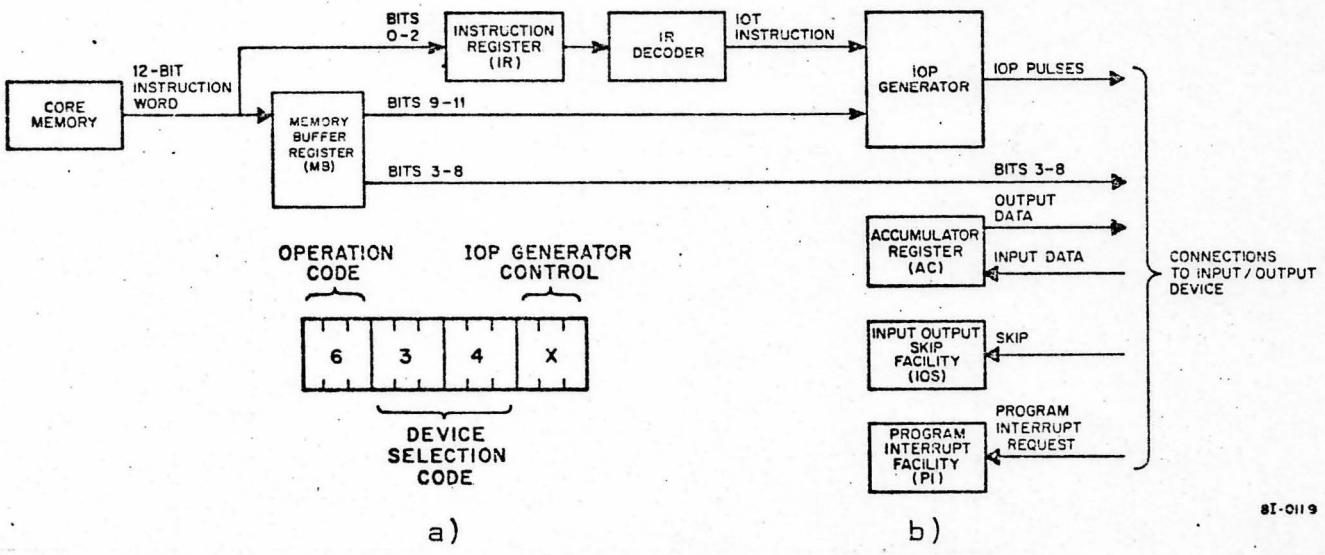


Fig. 29 a) Kodierung der IOT-Instruktionen
b) Blockschema des programmierten Datentransfers

Ein programmierter Datentransfer beginnt, wenn der Computer anhand der ersten drei IOT-bits (0-2) im Instruktions-Register erkennt, dass es sich um einen IOT-Befehl handelt. Der Computer sendet hierauf ein Kontrollsignal zu jeder peripheren Einheit und veranlasst diese, bits 3-8 zu decadieren. Durch diese 6 bits wird eine bestimmte Einheit angewählt, für welche die IOT-Instruktion gilt. Bits 9-11 enthalten die codierte Operationsspezifikation, welche im IOP-Generator eine Sequenz von 1-4 Impulsen auslöst, die zum I/O-Interface des Gerätes laufen. Dort bestimmen diese Impulse je nach Konstruktion des Interfaces, was geschehen soll. Da diese Impulse seriell auslaufen, hängt die Zeit, welche eine IOT-Instruktion benötigt, von der Anzahl der Impulse ab:

IOT-Instruktion	Anzahl IOPs	Zeit in µsec
0	1	1,2
1	2	2,6
2	2	2,6
3	3	3,6
4	2	2,6
5	3	3,6
6	3	3,6
7	4	4,6

Beispielsweise können so 12 bits in 2,6 µsec aus dem Geräte-Register in den Akkumulator transferiert werden. Diese Zahl darf aber nicht bedenkenlos auf eine Serie von Transaktionen übertragen werden. Denn der Akkumulatorinhalt muss jedesmal in den Kernspeicher gelesen werden, bevor neue Daten einlaufen dürfen. Andererseits ist das periphere Gerät oft aus mechanischen Gründen träge (z.B. Teletype), so dass nach einem Transfer zyklisch getestet werden muss, ob der Inhalt des I/O-Geräte-Registers schon erneuert worden ist. Beim Teletype z.B. muss durchschnittlich 250000 mal abgefragt werden, bevor sich der Status verändert. Die theoretische

Transferrate von 380000 bytes/sec reduziert sich so enorm.

Es ist aber für unsere Anwendung denkbar, dass wir die theoretische Rate viel besser ausnützen können, da wir z.B. einen Schieberegister-Inhalt via AC in den Kernspeicher befördern wollen. Hierzu muss natürlich ein I/O-Interface speziell konstruiert werden, was aber, worauf ich unten eingehe, bequem durchzuführen ist. Für einen Schieberegister-Transfer darf infolgedessen etwa mit 100000 bytes/sec gerechnet werden.

b) "Program Interrupt Transfer"

Ein Computer, der mit peripheren Einheiten in Verbindung steht, ist besser ausgenützt, wenn das Programm die I/O-Operation initialisiert und darauf mit dem Programmablauf fortfährt. In diesem Betriebzustand (interrupt-system on) wird in jedem Arbeitszyklus beiläufig geprüft, ob ein I/O-flag anzeigt, dass die graphische Einheit für ein I/O-Transfer bereit ist (interrupt request signal). Dieses Signal unterbricht den Programmablauf und schiebt eine Subroutine ein, die bestimmt, welches Gerät den interrupt verlangt hat, und die anschliessend den Datentransfer durchführt. In diesem Uebergang spielen die Speicherplätze ØØØØ und ØØØ1 von page Ø eine wichtige Rolle. Nach Abschluss des Datentransfers, der im wesentlichen den Operationen gemäss a) entspricht, fährt der Computer an der unterbrochenen Stelle mit dem Hauptprogramm fort.

Es ist ohne weiteres einzusehen, dass, verglichen mit a) mit dieser Methode Daten nicht schneller zum und vom Computer übertragen werden können. Trotzdem wird die z.B. im on-line-Betrieb zur Verfügung stehende Zeit viel besser ausgenützt als mit der Methode a). Das Hauptprogramm läuft mit Ausnahme von Transferperioden ständig. Daten können - sobald sie zur Verfügung stehen - jederzeit übertragen werden. Zwischen verschiedenen Datenquellen können Prioritäten einge-

führt werden, etc. Alle diese Vorteile sind für unseren Fourierprozessor möglicherweise willkommen, so dass dieser Interfacemethode besondere Beachtung geschenkt werden muss.

c) "Data Break Transfer"

Datentransfer unter Programmkontrolle läuft immer über den Akkumulator. Falls die Daten vom oder zum Kern- (bzw. RAM-) Speicher übertragen werden müssen, ist der Umweg über den Akkumulator nachteilig. Ausserdem muss der Akkumulatorinhalt vor dem Transfer gespeichert und nachträglich wieder eingelesen werden. Für extrem schnellen Datenaustausch sind diese Sekundärmanipulationen nachteilig.

Demgegenüber werden im data break Modus die Daten, ohne jede Programmkontrolle, direkt in den Speicher ein- bzw. ausgelesen. Data break ist besonders vorteilhaft im Verkehr mit extrem schnellen peripheren Geräten, wie Platten-, Trommel- und schnelle Bandspeicher, wo lange Datenblöcke verschoben werden müssen. Es ist wahrscheinlich, dass dieser Modus, obwohl hardwaremässig aufwendig, für die Verbindung zwischen den Schieberegistern des FFT-Prozessors und dem PDP-8/A-RAM-Speicher ideal ist. Möglicherweise muss er auch für die Verbindung zwischen PDP-8/A und PDP-12 in Erwägung gezogen werden. Die gegenüber Programm-Transfer kompliziertere Verschaltung der Dialogpartner wird sich hier aber besonders nachteilig auswirken, da wir, durch Separierung des EEG-Analysators vom Laborcomputer, ja eben diese Verbindung vereinfachen möchten.

Von den beiden Versionen "single-cycle" und "three-cycle data break" beschreibe ich nur die erste, da diese die schnellere ist

und für Spezialkonstruktionen empfohlen wird. Fig. 30 zeigt ein vereinfachtes data break - Blockschema; der logische Ablauf geht aus Fig. 31 hervor. Für ein single-cycle data break ist charakteristisch, dass sich die beiden Register "current adress (CA)" und "word count (WC)" im Interface befinden. Der Ablauf zerfällt in drei Phasen.

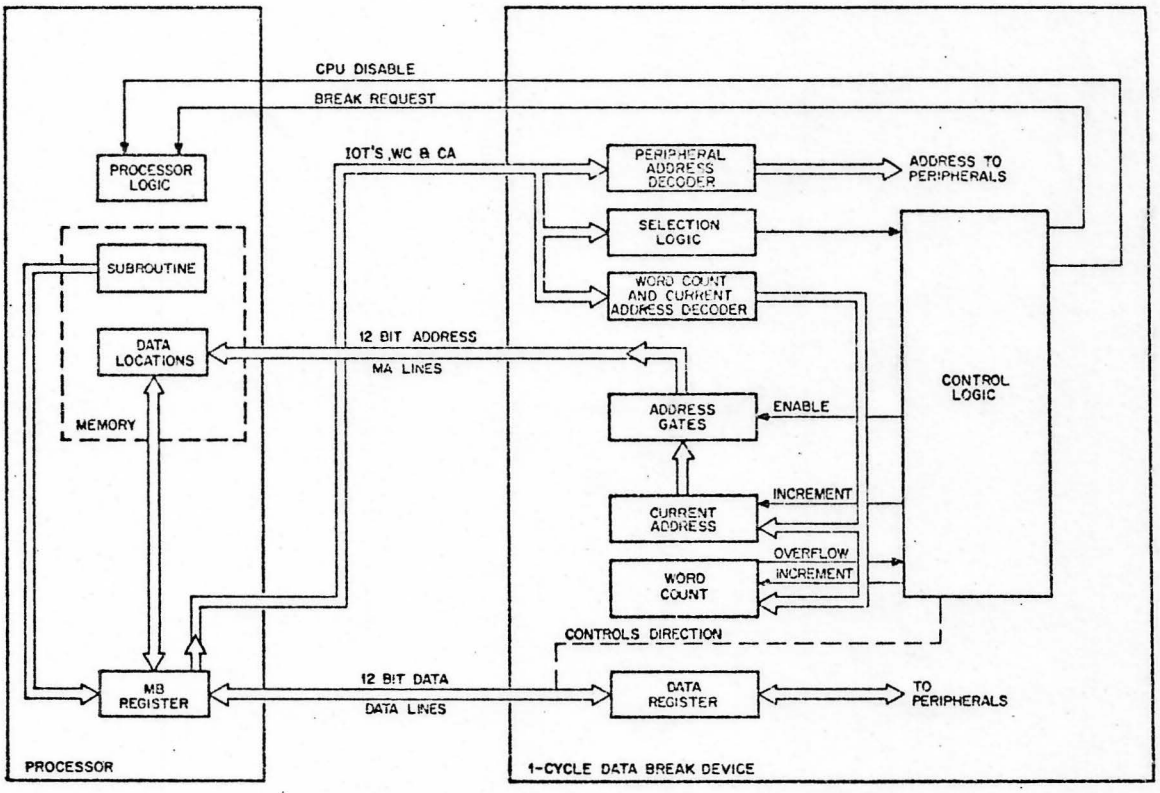


Fig. 30

Vereinfachtes Single-Cycle Data Break Blockschema

Während der ersten Phase "initial set-up" vollzieht das laufende Programm IOT-Instruktionen, welche das WC-Register mit dem 2er-Komplement der Länge des zu transferierenden Blocks füllen. Ausserdem füllen sie das CA-Register mit der Memory-Startadresse, bestimmen die Richtung des Transfers, etc. Die letzte dieser Instruktionen schaltet die Kontrollogik ein.

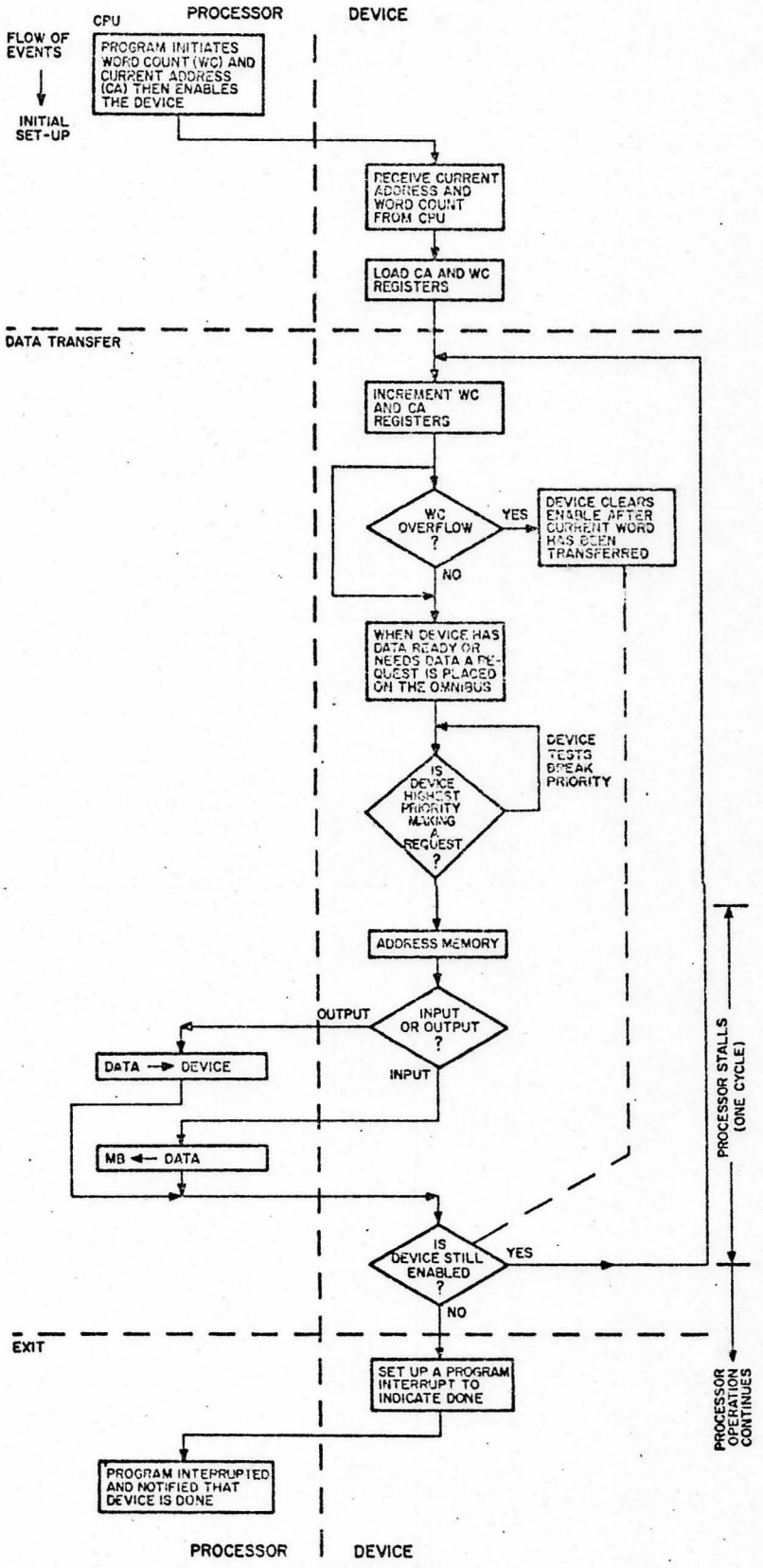


Fig. 31

Single-Cycle Data Break Flussdiagramm

Sobald das periphere Gerät das erste Datenwort ins "data register" schleust (input) bzw. zur Datenaufnahme bereit ist (output), erscheint ein Signal "break request" im OMNIBUS. Erst dann unterbricht der Computer den Programmablauf (AC- und MQ-Register-Inhalte bleiben an Ort), schaltet um in den "direct memory access" - Zustand (DMA), transferiert ein Datenwort, schaltet zurück und fährt mit dem normalen Programmablauf fort, bis das periphere Gerät erneut bereit ist. Im Programmablauf wird so je Datenwort ein Zyklus für die Transaktion "gestohlen". Im günstigsten Fall (trägheitsloser I/O-Partner) ist für die Uebertragung des ganzen Blocks der Computer ununterbrochen im DMA-Zustand. Normalerweise aber ist dies nicht der Fall, und der Computer vollzieht in den Wartezeiten zwischen den Transfers von Einzelworten normale Programmschritte.

Wenn der ganze Block durchgelaufen ist (WC hat overflow), beginnt die dritte Phase ("exit phase") der data break-Operation. Wie Fig. 31 zeigt, wird ein interrupt erzeugt, worauf der Computer mit IOT-Instruktionen testet, ob z.B. Transferfehler vorgekommen sind. Damit ist der I/O-Prozess abgeschlossen.

Je länger der Datenblock ist, welcher im data break-Modus übertragen wird, desto unbedeutender wird der Zeitverlust durch den vorangehenden und abschliessenden IOT-Austausch. Bestenfalls können 4096 aufeinanderfolgende bytes mit einem einzigen kleinen Satz Steuerinstruktionen transferiert werden. Der eigentliche Transfer eines 12 bit-Wortes beansprucht einen Maschinenzyklus von 1,5 /usec. Somit können maximal 667000 Wörter / sec resp. 8 Millionen bits/sec übertragen werden. Im PDP-12 ist diese Rate mit 6,5 Millionen bits / sec angegeben.

3.3. Gesamtplan

Die obigen Erörterungen des Zeitbedarfs und des Charakters von I/O-Operationen ergeben, dass die Separierung des Fourierprozessors vom jeweiligen Laborcomputer nicht an Interface-Problemen scheitern kann. Mit Bezug auf die Transferrate allein, scheinen alle drei Modi für den Verkehr zwischen Laborcomputer und Prozessor geeignet zu sein. Der programmierte interrupt ist vielleicht der günstigste Kompromiss. Damit können Prioritäten gesetzt werden, das laufende Programm wird wenig beeinflusst, und dennoch ist der Hardwareaufwand bescheiden im Vergleich zum data break transfer. Die Kommunikation via Akkumulator ist im Zusammenhang mit dem A/D-Konverter vielleicht vorteilhaft, da man sich die Zwischenspeicherung im PDP-12 Kernspeicher evtl. ersparen kann.

Ein roher planerischer Entwurf des Fourierprozessors, der, unter Verwendung eines PDP-8/A-Minicomputers als Steuerorgan, eine selbständige und anpassungsfähige Einheit werden soll, findet der Leser in Fig. 32. Der OMNIBUS-Datenkanal verbindet alle Untereinheiten - mit Ausnahme des verdrahteten Fouriertransformators. Dieser findet zusammen mit der digitalen Frequenzverformung kaum im PDP-8/A-Standartchassis Platz. DEC offeriert aber ein reichhaltiges Programm von kompatiblen mechanischen und elektronischen Einheiten zum Selbstbau logischer Systeme (vgl. (7) und (9)).
Die Konstruktion eines Prototyps wird sehr vereinfacht, wenn diese vorgefertigten Teile benützt werden. Falls sich zu einem spätern Zeitpunkt bewährte Routinen zur Spektralanalyse herauskristalisieren sollten, ist das ROM-Memory angezeigt. Bufferoperationen im engeren Zusammenhang mit Frequenzverformung und FFT werden wohl mit Vorteil von dynamischen Schieberegistern im verdrahteten Zusatzteil übernommen.

Es folgt die Preisabschätzung der Einbettungselektronik (PDP-8/A-System ohne FFT und Frequenzverformung) in Minimalkonfiguration:

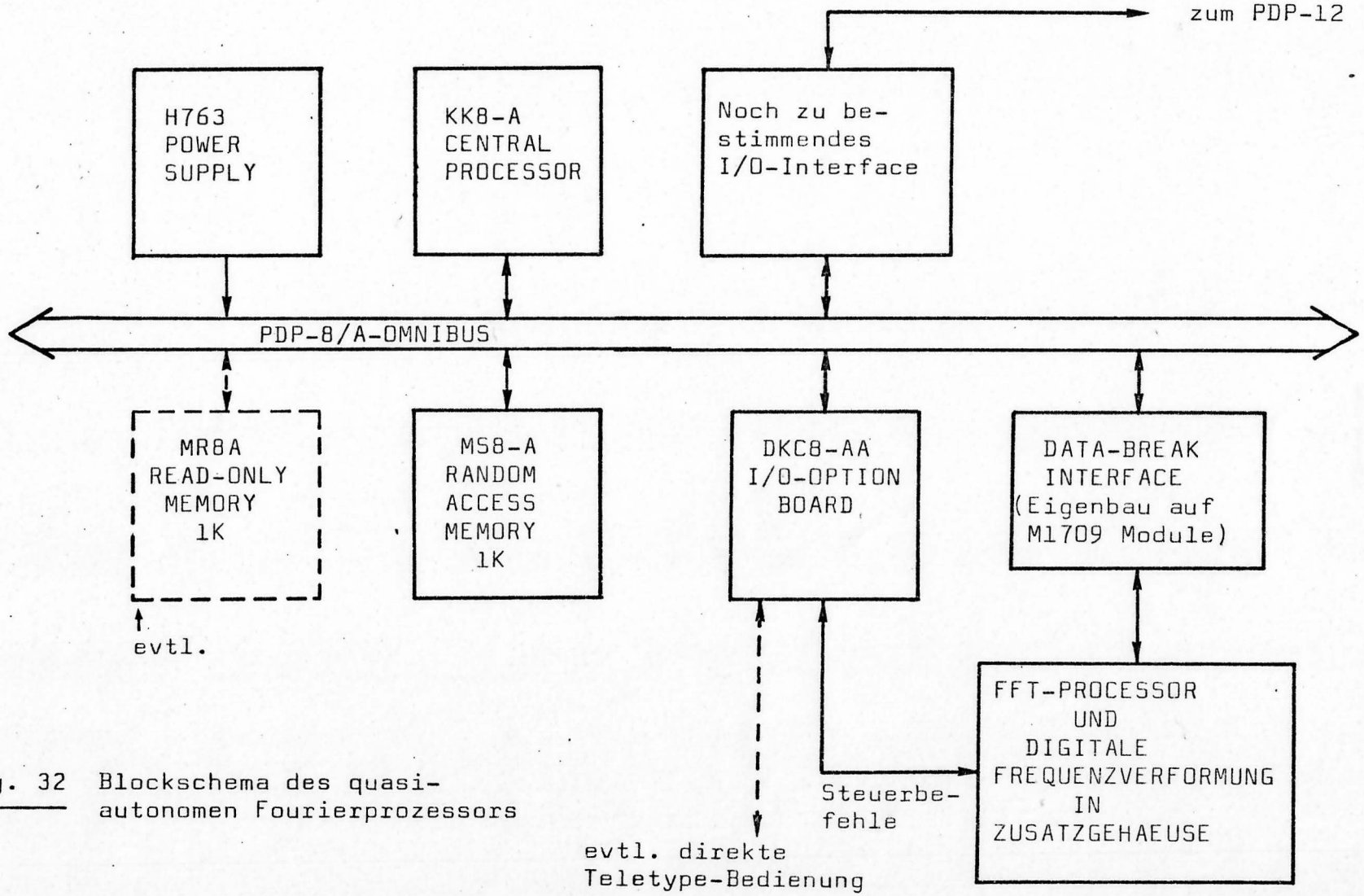


Fig. 32 Blockschema des quasi-autonomen Fourierprozessors

PDP-8/A:	Central Processor, 1k RAM, Chassis mit OMNIBUS und Netzgerät, Operator's- und Programmer's-Console	ca. Fr.	7'200.--
DKC8/AA:	I/O-Option Board enthaltend TTY-Interface, Parallel I/O-Interface, Real-Time Clock, Programmer's Console Control	ca. Fr.	2'100.--
I/O-Interface zum PDP-12 evtl. Kombination von M 1703 und M 1705	zusammen	ca. Fr.	675.--
Data Break-Interface Eigenbau auf M 1709 OMNIBUS Interface Foundation Module		ca. Fr.	600.--
		Fr.	10'575.--

Dieser Preis ist äusserst günstig! Ein Firmenbesuch bei DEC-Zürich hat ausserdem ergeben, dass der Hochschule evtl. Rabatte gewährt werden könnten. Der 100-Stück OEM-Preis dürfte sich etwa um Fr. 5'000.-- bewegen.

Das in Fig. 32 dargestellte System könnte sukzessive zu einem völlig selbständigen, anpassungsfähigen Spektralanalyse-Prozessor für biologische Signalanalyse erweitert werden. Der PDP-8/A ist mit über 60 Optionen und peripheren Geräten direkt kompatibel. Ueber 600 Software-Programme sind erhältlich. Diese Horizonte verheissen ein Gerät, das einem vielseitigen Benützerkreise dienen könnte.

4. DIGITALE FREQUENZVERFORMUNG

Unser Vorsatz, zur EEG-Spektralanalyse und Merkmalsverdichtung einen leistungsfähigen und vielseitigen Spezialprozessor zu entwickeln, führt zwangsläufig zum Problem der Feinauflösung im Spektralbereich. Im fouriertransformierten Elektroenzephalogramm gibt es, wie im Spektrum der Sprache, Bereiche grosser und geringer Bedeutung für die Entschlüsselung der im Signal enthaltenen spezifischen Botschaft. Schon Berger, der Geburtshelfer der klinischen Elektroenzephalographie, erkannte die geheimnisvolle Korrespondenz zwischen Aufmerksamkeit (Vigilanz) und Alphawellen (1). Definitionsgemäss ist der Alpharhythmus auf das verhältnismässig schmale Fenster von 8 bis 13 Hz beschränkt. In der EEG-Grundaktivität wurden ausserdem die Bänder β, ϑ, δ zur Benennung der Signalaktivität um 20, 6 resp. 3 Hz eingeführt. ϑ und δ-Wellen kennzeichnen das Schlaf-EEG, sind aber auch von klinisch-pathologischem Interesse. Eher zu transienten Signalbestandteilen müssen die σ-Wellen (14 Hz) gezählt werden, die in den sog. Schlafspindeln auftreten.

Global, d.h. über das ganze EEG-Fenster von 0 bis 50 Hz beurteilt, nimmt das informative Gewicht der Spektrallinien bei gleichbleibender Auflösung mit steigender Frequenz ab. Da die Berechnung einer Spektrallinie mit einem bestimmten Aufwand verbunden ist, wäre es sinnreich, ein Verfahren anzuwenden, das die Liniendichte zugunsten der niederfrequenten Bereiche mit wachsender Frequenz reduziert; eine Vorrichtung also, welche die Frequenzachse geeignet nichtlinear verformt.

Genau ein solches Verfahren wurde neuerdings von A. Oppenheim (18, 19) angegeben. Er hat es mit "digital frequency warping" bezeichnet, was wir zu deutsch digitale Frequenzverformung (DFV) nennen wollen. Dieses Verfahren formt eine Signalfolge derart um, dass die Fouriertransformierten der umgeformten und der ursprünglichen Folgen durch eine nichtlineare Verzerrung der Frequenzachse zusammenhängen. Eine Fourieranalyse des umgeformten Signals mit gleichmässiger Auflösung entspricht dann einer Analyse des ursprünglichen Signals mit einer ungleichmässigen Auflösung.

Da die beiden Quellen bei Oppenheim didaktisch eher ungünstig abgefasst sind (es treten in der Darstellung Lücken auf, die schwierig zu verifizieren sind), und da eine Diplomarbeit auch studienhalber durchgeführt wird, möchte ich bei meiner Darstellung versuchen, die bei Oppenheim verbliebenen Lücken zu überbrücken.

Folgende Grafik ist geeignet, in den Problemkreis zu führen und gibt anfängliche Definitionen:

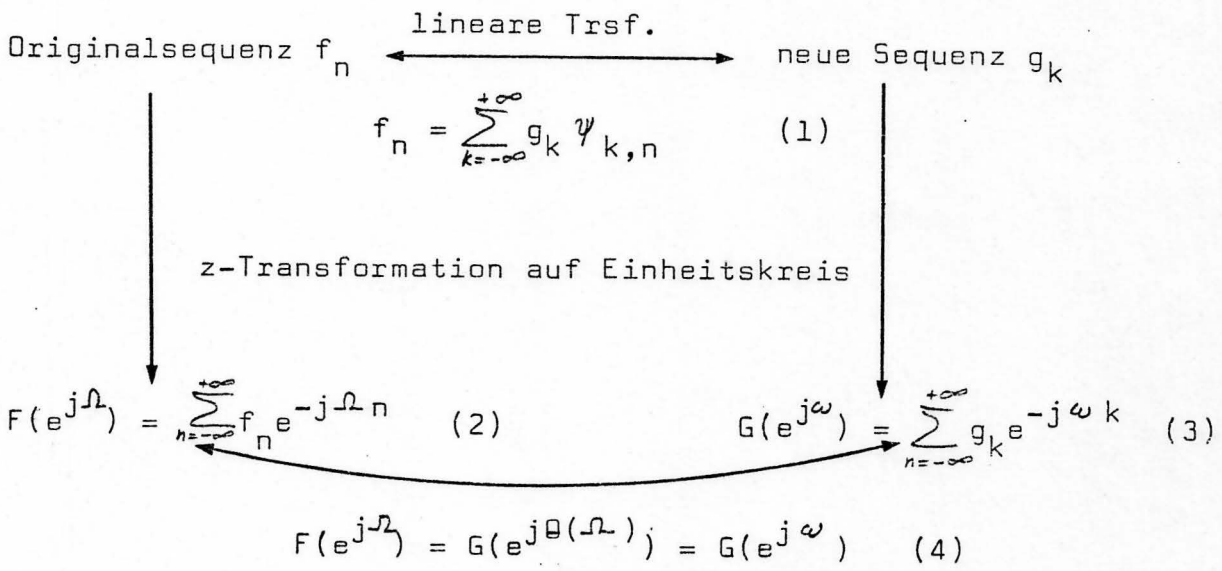


Fig. 1

(1) ist zunächst noch nicht bestimmt. Es wird von f_n und g_k lediglich verlangt, dass sie linear auseinander hervorgehen. Die z-Transformierten[2] der beiden Sequenzen f_n und g_k sollen ausserdem identisch sein, wenn für die Argumente auf dem Einheitskreis folgende Relation gilt

$$\omega = \Theta(\Omega) \qquad (4)$$

Es stellt sich nun die Aufgabe, diese Verzerrung der Frequenzachse zu berechnen.

---

[2] Die z-Trsf. auf dem Einheitskreis darf eigentlich nur dann mit der Fouriertransformierten gleichgesetzt werden, wenn die diskrete Sequenz f_n mit $f(t) = \sum_{-\infty}^{\infty} f_n \delta(t-n)$ eine kontinuierliche Zeitfunktion darstellt. Zu $f(t)$ gehört dann die FT $F(\Omega) = \sum_{-\infty}^{\infty} f_n \exp(-j\Omega n)$.

Was auf dem Einheitskreis im speziellen gilt, dass nämlich die beiden z-Transformierten identisch sind für eine zunächst unbekannte Variablensubstitution, kann auf die ganze z-Ebene erweitert werden. Unter der Abbildung $\hat{z} = m(z)$ ist $F(z) = G(m(z)) = G(\hat{z})$. Für diese Abbildung m fordern wir mithin, dass sie den Einheitskreis in der z-Ebene in denjenigen der $\hat{z}$-Ebene überführt: Für bestimmte Punkte desselben, die durch die Winkelgeschwindigkeiten $\hat{\Omega}$ bzw. Ω bestimmt sind, können wir schreiben

$$e^{j\hat{\Omega}} = m(e^{j\Omega}) = e^{j\vartheta(\Omega)} \quad , \quad \hat{\Omega} = \vartheta(\Omega) \tag{5}$$

Für die linearen Entwicklungskoeffizienten $\psi_{k,n}$ hat die z-Transformierte

$$\Psi_k(z) = \sum_{n=-\infty}^{+\infty} \psi_{k,n} z^{-n} \tag{6}$$

die Eigenschaft, dass

$$\Psi_k(z) = (\Psi_1(z))^k \tag{7}$$

was bei ( ) begründet ist. Die wichtige Relation (7) wird hier als gegeben aufgefasst. Ausserdem ist dort dargelegt, weshalb die Beziehung

$$m(z) = (\Psi_1(z))^{-1} \tag{8}$$

vorausgesetzt werden darf. (7) und (8) folgen aus den Prämissen in Fig. 1, insbesondere aus der Tatsache, dass das Spektrum von g_k eine verformte Variante desjenigen von f_n ist (Gleichung (4)). Mit den Gleichungen (5), (7) und (8) ergibt sich

$$\Psi_k(e^{j\Omega}) = e^{-jk\vartheta(\Omega)} \quad . \tag{9}$$

Die Fouriertransformation[2] $\Psi_k^*(\Omega) = \Psi_k(e^{j\Omega})$ der Folge $\psi_{k,n}$ der Entwicklungskoeffizienten hat Allpass-Charakteristik, denn sie verändert sich mit Ω bloss hinsichtlich der Phase. Ihr Betrag ist gemäss (9) immer gleich Eins. Dies ist eine erste wichtige

Bedingung für $\Psi_k(z)$, welches, einmal bekannt, die Abbildungsvorschrift (1) zwischen den Sequenzen festlegt.

Fernerhin wird gefordert, dass m(z) rational ist und den Einheitskreis von z überführt in den Einheitskreis von $\hat{z}$. In der komplexen Analysis kennt man die Klasse der Möbiustransformationen oder Kreisverwandtschaften, welche genau diesen Forderungen entsprechen (22). Insbesondere die Untergruppe der hyperbolischen Bewegungen

$$m(z) = \eta \frac{z - a}{a^* z - 1} \quad , \quad 0 \leq |a| < 1 \quad , \; |\eta| = 1 \qquad (10)$$

bildet Einheitskreise, sowie deren Inneres und Aeusseres jeweils auf sich ab. Mit der Wahl von $\eta = -1$ und einer geringfügigen Umformung gelangt Oppenheim zur analytischen Darstellung eines Allpass erster Ordnung

$$\hat{z} = m(z) = \frac{1 - az^{-1}}{z^{-1} - a^*} \quad , \qquad (11)$$

welcher das Intervall $-\pi < \Omega \leq \pi$ auf $-\pi < \hat{\Omega} \leq \pi$ abbildet, o.n. Fixpunkte in $e^{\pm j}$ besitzt. Diese letzte Eigenschaft macht die Abbildung 1-1-deutig, sodass also keine Punkte auf dem $\hat{z}$-Einheitskreis existieren, denen entweder keine oder mehrere Urbilder zugeordnet sind.

Die z-Transformierten der Sequenzen $\{\psi_{k,n}\}$ in der Entwicklung (1) sind demgemäss

$$\Psi_k(z) = \left(\frac{z^{-1} - a^*}{1 - az^{-1}} \right)^k \quad . \qquad (12)$$

Das entspricht einem Allpass k-ter Ordung mit k Polen bei z=a und k Nullstellen bei $z = 1/a^*$. Dadurch ist die lineare Beziehung zwischen f_n und g_k völlig bestimmt. Aus (9) und (12) und unter Einschränkung auf reelles a folgt

$$\hat{\Omega} = \theta(\Omega) = \arctan\left[\frac{(1 - a^2)\sin\Omega}{(1 + a^2)\cos\Omega - 2a} \right] \quad . \qquad (13)$$

Beweis: $$e^{-j\vartheta(\Omega)} = \frac{e^{-j\Omega} - a^*}{1 - ae^{-j\Omega}}$$

Subst: $\cos\vartheta(\Omega) = \xi$, $\sin\vartheta(\Omega) = \eta$

$\cos\Omega = x$, $\sin\Omega = y$

$$\xi - j\eta = \frac{x - jy - a^*}{1 - a(x - jy)} = \frac{(x - a^*) - jy}{(1 - ax) + jay}$$

$$= \frac{((x - a^*) - jy)((1 - ax) - jay)}{(1 - ax)^2 + a^2y^2} \qquad (= N)$$

$$= \frac{(x - a^*)(1 - ax) - ay^2}{N} - j\,\frac{y(1 - ax) + ay(x - a^*)}{N}$$

$$\tan\vartheta(\Omega) = \frac{\eta}{\xi} = \frac{y(1 - ax) + ay(x - a^*)}{(x - a^*)(1 - ax) - ay^2}$$

$$= \frac{(1 - a^2)\,y}{(1 + a^2)x - 2a} \qquad = \frac{(1 - a^*a)\,y}{(1 + a^*a)x - \underbrace{a - a^*}_{-2\mathrm{Re}\{a\}}}$$

Schluss durch beidseitiges Anwenden von arctan.

Die graphische Darstellung des Zusammenhangs in (13) gibt Fig. 2 für verschiedene reelle Werte des Parameters a. Der Zusammenhang ist nichtlinear, und die Fouriertransformierte von g_k ist gleich der Fouriertransformierten von f_n berechnet auf einer nichtlinearen Frequenzachse:

$$F(e^{j\Omega}) = G(e^{j\vartheta(\Omega)})$$

Für $a > 0$ wird die Auflösung für niedere Frequenzen vergrössert, was in unserem Fall erwünscht ist.

Nach all diesen Vorbereitungen kann die Frage angepackt werden wie g_k konkret aus f_n gewonnen wird. Es gelte in Anlehung an die Praxis von jetzt an die Einschränkung, dass $f_n = 0$ für $n < 0$ und

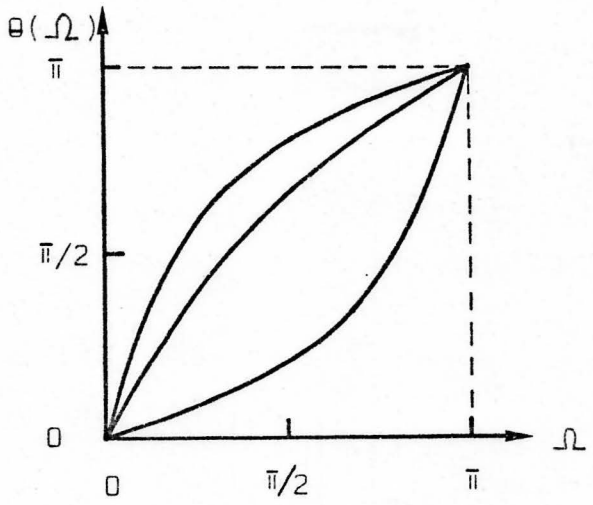


Fig. 2 Graph der Verformungsfunktion (13) für verschiedene reelle Werte von a.

$g_k = 0$ für $k < 0$. Ausgangspunkt ist die gewichtete Orthogonalität zwischen den Sequenzen $\{\psi_{k,n}\}$:

$$\sum_{n=0}^{\infty} n\, \psi_{r,n}\, \psi_{k,n} = \begin{cases} k & \text{falls } k = r \\ 0 & \text{falls } k \neq r \end{cases} \qquad (14)$$

Die Umkehrformel zu (1) ist damit

$$g_k = \frac{1}{k} \sum_{n=0}^{\infty} n\, \psi_{k,n}\, f_n \qquad \text{falls } k > 0$$

$$g_k = 0 \qquad \text{falls } k < 0 \qquad (15)$$

Beweis: Mit Hilfe von (1):

$$\sum_n n\, \psi_{k,n} \left\{ \sum_{k'} g_{k'}\, \psi_{k',n} \right\} = \sum_{k'} g_{k'} \left\{ \sum_n n\, \psi_{k,n}\, \psi_{k',n} \right\}$$

$$= \sum_{k'} g_{k\,k'}\, \delta_{k,k'} = g_{k\,k} \quad .$$

Die Entwicklungskoeffizienten $h_{k\,n} = (1/k)n\, \psi_{k\,n}$ in (15) können

$$Z(nf_n) = -z \frac{d}{dz} Z(f_n) \quad . \tag{16}$$

Die gesamte Systemfunktion wird demgemäss

$$H_k(z) = \frac{1}{k} Z(n \psi_{k,n}) = \frac{1}{k} (-z) \frac{d}{dz} \Psi_k(z) \tag{17}$$

und unter Verwendung von (12)

$$H_k(z) = \frac{(1 - |a|^2) z^{-1}}{(1 - az^{-1})^2} \left[\frac{z^{-1} - a^*}{1 - az^{-1}} \right]^{k-1} \quad k=1,2,... \tag{18}$$

$g_o = \sum_{n=0}^{\infty} h_{o,n} f_n$ kann aus (15) nicht ermittelt werden, es fehlt aus diesem Grunde auch die zugehörige Systemfunktion $H_o(z)$. Aus einem von Oppenheim nicht angegebenen Grunde ist aber

$$\psi_{k,0} = (-a)^k \quad , \quad k = 0,1,2,... \tag{19}$$

insbesondere wird $\psi_{o,o} = 1$. Aus (1) erhalten wir infolgedessen

$$g_o = f_o - \sum_{k=1}^{\infty} g_k (-a)^k \tag{20}$$

und schreiben unter Benutzung von (15)

$$g_o = f_o - \sum_{n=0}^{\infty} f_n n \sum_{k=1}^{\infty} \frac{1}{k} (-a)^k \psi_{k,n} \quad , \tag{21}$$

was mit der Substitution

$$v_n \equiv \sum_{k=1}^{\infty} \frac{1}{k} (-a)^k \psi_{k,n} \tag{22}$$

einfacher wird:

$$g_o = f_o - \sum_{n=1}^{\infty} f_n n v_n = \sum_{n=0}^{\infty} f_n (\delta_n - n v_n) \quad . \tag{23}$$

δ_n ist ein Einheitsimpuls, d.h. $\delta_n = 1$ für $n=0$ und $\delta_n = 0$ für $n \neq 0$. Wir haben mit (23) die bisher noch unbekannte Einheitsimpulsfolge $h_{o,n} = (\delta_n - n v_n)$ erhalten. Gesucht ist deren z-Transformierte. Zunächst suchen wir sie für $n v_n$:

$$\sum_{n=0}^{\infty} n v_n z^{-n} \overset{(16)}{=} -z \frac{d}{dz} \sum_{n=0}^{\infty} v_n z^{-n}$$

$$= -z \frac{d}{dz} \sum_{n=0}^{\infty} \sum_{k=1}^{\infty} \frac{1}{k} (-a)^k \psi_{k,n} z^{-n}$$

$$= -z \frac{d}{dz} \sum_{k=1}^{\infty} \frac{1}{k} (-a)^k \sum_{n=0}^{\infty} \psi_{k,n} z^{-n}$$

$$\overset{(12)}{=} -z \frac{d}{dz} \sum_{k=1}^{\infty} \frac{1}{k} (-a)^k \left(\frac{z^{-1} - a^*}{1 - a z^{-1}} \right)^k$$

$$= -z \sum_{k=1}^{\infty} \frac{1}{k} (-a)^k k \left(\frac{z^{-1} - a^*}{1 - a z^{-1}} \right)^{k-1} \cdot \left(\frac{-(z^{-1} - a^*)(a z^{-2}) - z^{-2}(1 - a z^{-1})}{(1 - a z^{-1})^2} \right)$$

$$= -z \sum_{l=0}^{\infty} (-a) \left(\frac{z^{-1} - a^*}{1 - a z^{-1}} \right)^l (-a)^l \cdot \frac{+a z^{-3} - a a^* z^{-2} - z^{-2} - a z^{-3}}{(1 - a z^{-1})^2}$$

$$= \underbrace{\left\{ \sum_{l=0}^{\infty} \left(\frac{-a z^{-1} + a a^*}{1 - a z^{-1}} \right)^l \right\}}_{\text{geom. Reihe}} \frac{(-a z^{-1})(1 - a a^*)}{(1 - a z^{-1})^2}$$

$$\| \qquad \text{falls } \left| \frac{-a z^{-1} + a a^*}{1 - a z^{-1}} \right| < 1$$

$$\frac{1}{1 - \left(\frac{-a z^{-1} + a a^*}{1 - a z^{-1}} \right)}$$

$$\|$$

$$\frac{1 - a z^{-1}}{1 - a z^{-1} + a z^{-1} - a a^*}$$

$$= - \frac{(1 - a z^{-1}) \cdot a z^{-1} (1 - a a^*)}{(1 - a a^*)(1 - a z^{-1})^2}$$

$$= - \frac{a z^{-1}}{1 - a z^{-1}} = Z(n v_n) . \qquad (24)$$

Die z-Trsf. von $h_{o,n} = (\delta_n - n v_n)$ wird demnach

$$H_o(z) = \sum_{n=0}^{\infty} (\delta_n - n v_n) z^{-n}$$

$$= \sum_{n=0}^{\infty} \delta_n z^{-n} - \sum_{n=0}^{\infty} n v_n z^{-n}$$

$$= 1 + \frac{a z^{-1}}{1 - a z^{-1}}$$

$$H_o(z) = \frac{1}{1 - a z^{-1}} . \qquad (25)$$

Zusammenfassend kann dieses lineare schiebungsinvariante System wie folgt beschrieben werden:

$$H_k(z) = \begin{cases} \dfrac{(1-|a|^2)z^{-1}}{(1-az^{-1})^2}\left[\dfrac{z^{-1}-a^*}{1-az^{-1}}\right]^{k-1} & \text{falls } k>0 \\ \dfrac{1}{1-az^{-1}} & \text{falls } k=0 \end{cases} \qquad (26)$$

Auffallend ist, dass die Systemfunktion ab k=1 kaskadisch wird, d.h. die Vergrösserung von k um einen Schritt zieht die Multiplikation von $H_k(z)$ mit dem festen Faktor $(z^{-1}-a^*)/(1-az^{-1})$ nach sich. Diese Kaskadenstruktur wird einen iterativen Funktionsablauf im Rechner ermöglichen.

Wie sind die Funktionen (26) zu interpretieren? g_k entsteht durch Linearkombination der f_n mit den Einheitsimpulsantworten $h_{k,n}$ als Gewichte. $h_{k,n}$ wird dabei als Sequenz in n mit Parameter k aufgefasst. Zu diesen Sequenzen gehören via z-Transformationen die Systemfunktionen $H_k(z)$ mit demselben Parameter. Nur diese sind bis jetzt explizite gegeben. Aus ihnen könnte über die inverse z-Transformation (Cauchy-Integral) $h_{k,n}$ aber berechnet werden; sie charakterisieren das digitale System also vollständig. Die Kaskade ist in Fig. 3 aufgezeichnet.

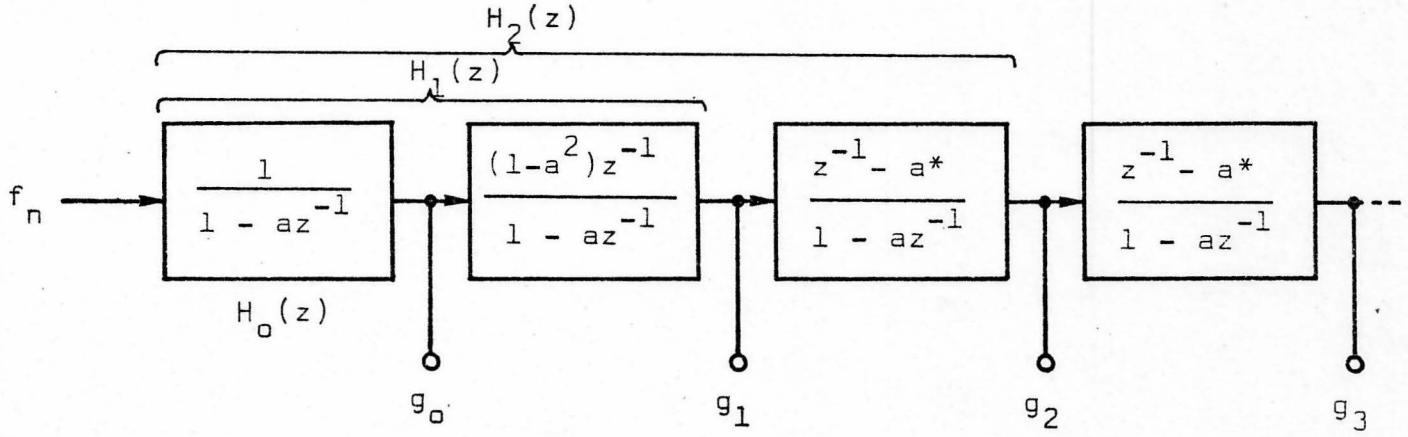


Fig. 3 Allpass-Netzwerk zur digitalen Frequenzverformung. Die Ausgangswerte g_k stehen an, nachdem alle Eingangswerte f_n in absteigender Reihenfolge verarbeitet sind.

$H_o(z)$ ist die Systemfunktion eines Netzwerks erster Ordnung zur Stossantwort $h_{o,n}$, welche g_o erzeugt. g_o steht fertig gerechnet an, wenn die ganze Sequenz f_n verarbeitet ist. Die Stützwerte werden in absteigender Reihenfolge eingelesen.

Dasselbe gilt auch für $H_1(z)$, nur dass auf dem Weg zu g_1 die Zwischenresultate $\tilde{g}_{o,k}$ von g_o mitverwendet werden können, dass die Maschinerie von $H_o(z)$ als Teil des Systems $H_1(z)$ also nicht mehr eigens ablaufen muss. Die Erweiterung auf $H_k(z)$ ist nun gut ersichtlich.

Oppenheim macht einen Vorschlag für die Verwirklichung des Gesamtsystems Fig. 3, wie er in Fig. 4 gezeigt ist.

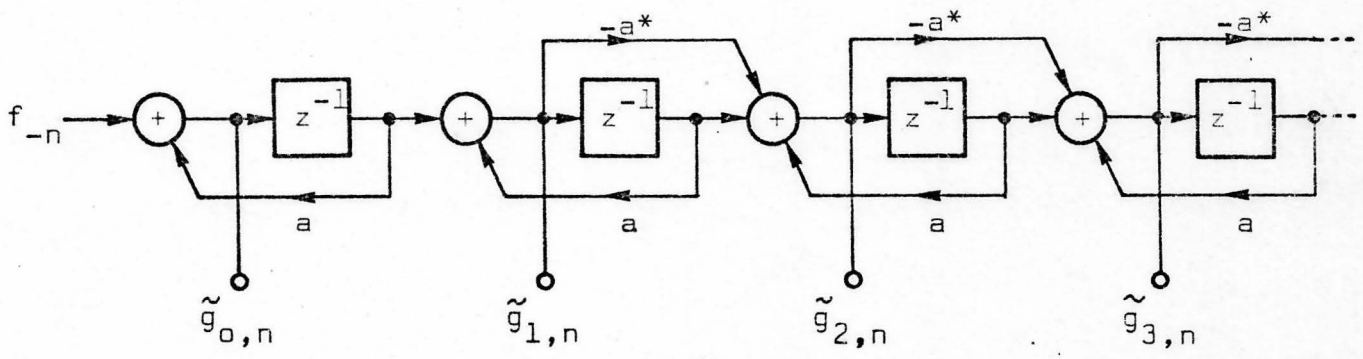


Fig. 4 Eine mögliche Schaltungsanordnung für digitale Frequenzverformung mit Verzögerungsgliedern, Multiplizier- und Addierwerken. (Man beachte: $g_k = g_{k,0}$)

Die Aequivalenz von Fig. 4 mit den Systemfunktionen (26) soll jetzt geprüft werden. Betrachten wir erstens die Erzeugung von g_o. Da die einzelnen Stufen der Kaskade völlig rückwirkungsfrei sind, ist für g_o nur ein beschränktes Teilsystem verantwortlich. Dieses ist in Fig. 5 aufgezeichnet

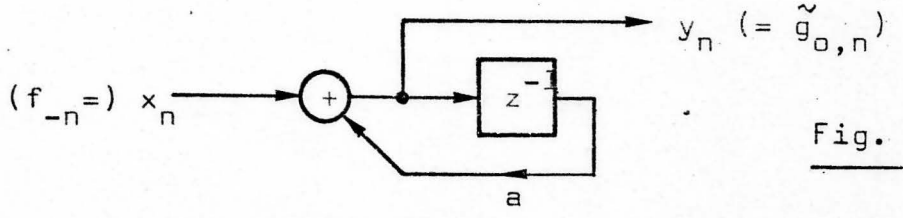


Fig. 5

Die Arbeitsweise kann mit Hilfe der Differenzengleichung

$$y_n = ay_{n-1} + x_n \tag{27}$$

dargestellt werden. Beidseitige z-Transformation mit der Anfangsbedingung $y_{-1} = 0$ liefert

$$\sum_{n=0}^{\infty} y_n z^{-n} = az^{-1} \sum_{n=0}^{\infty} y_{n-1} z^{-(n-1)} + \sum_{n=0}^{\infty} x_n z^{-n}$$

$$Y(z) = az^{-1}(\cancel{y_{-1}}^{0} z + \sum_{m=0}^{\infty} y_m z^{-m}) + X(z)$$

$$= az^{-1} Y(z) + X(z)$$

$$= \frac{1}{(1 - az^{-1})} X(z) = H_o(z) X(z)$$

Man vergleiche mit (26)! Die zweite Kaskadenstufe ist in Fig. 6 gezeigt; sie unterscheidet sich von Fig. 5 nur durch ein zusätzliches Verzögerungsglied am Eingang.

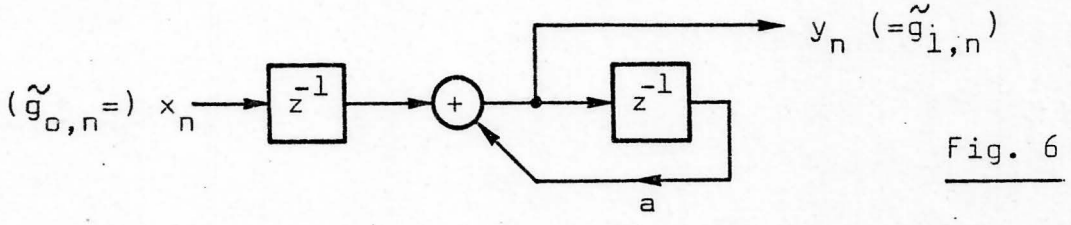


Fig. 6

Differenzengleichung zur zweiten Stufe:

$$y_n = ay_{n-1} + x_{n-1} \tag{28}$$

z-Transformation mit Anfangsbedingung $y_{-1} = x_{-1} = 0$:

$$Y(z) = az^{-1} Y(z) + z^{-1} X(z)$$

$$= \frac{1}{(1 - az^{-1})} X(z) \stackrel{?}{=} H_1(z) X(z)$$

Hier liegt ein Widerspruch zu (26) vor, was möglicherweise einen Fehler von Oppenheim im Netzwerk Fig. 4 anzeigt. Dieser Schönheitsfehler muss vor einer allfälligen Realisierung korrigiert

werden, z.B. durch einen zusätzlichen Term $-|a|^2 x_{n-1}$ in (28).

Schliesslich verifizieren wir noch die dritte Kaskadenstufe, die sich in der Folge ja nur noch wiederholt (Fig. 7).

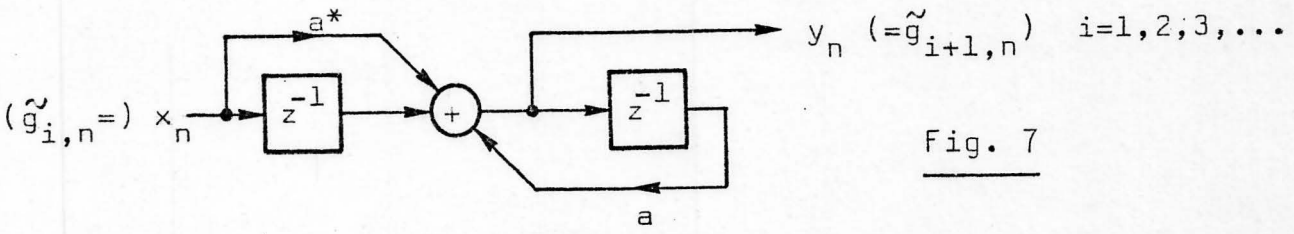


Fig. 7

Differenzengleichung:

$$y_n = ay_{n-1} + x_{n-1} - a^*x_n$$

z-Transformation mir Anfangsbedingung $y_{-1} = x_{-1} = 0$:

$$Y(z) = az^{-1}\,Y(z) + z^{-1}\,X(z) - a^*\,X(z)$$

$$= \frac{z^{-1} - a^*}{(1 - az^{-1})}\,X(z)$$

Damit wäre das Fundament für das Verständnis einer speziellen Hardwarekonfiguration gelegt.

4.1. Plan eines Hardware-Systems für digitale Frequenzverformung

Unter vollkommener Ausnützung der iterativen Möglichkeiten des Systems in Fig. 4 macht Oppenheim einen Vorschlag zu einer verdrahteten Darstellung des Algorithmus. Die Iterationsfolgen lauten zusammengefasst folgendermassen

$$\begin{aligned}
\tilde{g}_{0,n} &= a\tilde{g}_{0,n-1} + f_{-n} \\
\tilde{g}_{1,n} &= a\tilde{g}_{1,n-1} + \tilde{g}_{0,n-1} \\
&\vdots \\
\tilde{g}_{k,n} &= a\tilde{g}_{k,n-1} - a^*\tilde{g}_{k-1,n} + \tilde{g}_{k-1,n-1} \quad , \; k = 2,3,... \\
&\vdots
\end{aligned} \tag{30}$$

Der negative Index bei f_{-n} bedeutet, dass die Folge mit dem höchstindizierten Glied voraus eingelesen werden muss, das in der Praxis an endlicher Stelle, sagen wir bei f_N ist. Mit f_N am Eingang beginnt ein erster "Makrozyklus", während dem alle Schritte (="Mikrozyklen", k läuft) in (30) ausgeführt werden. In der Praxis seien es M Schritte bzw. Folgeglieder $\tilde{g}_{k,n}$. Dann wird der nächste Wert f_{N-1} eingeschleust und wiederum in der Spanne eines Makrozyklus M Mikrozyklen durchgeführt, etc. Man beachte, dass jeweils für die ersten beiden Mikrozyklen gewisse Modifikationen am Einheitsschritt vorgenommen werden müssen und sich erst ab k=2,3,4,... nichts mehr ändert. So werden alle Mikrozyklen durchgespielt, n sukzessive reduziert, bis auch für n=0 alle M Mikrozyklen erledigt sind. Unmittelbar leuchtet ein, dass die Rechenzeit proportional zu M.N ist.

Das Blockschema Fig. 8 ist nunmehr leicht zu begreifen.

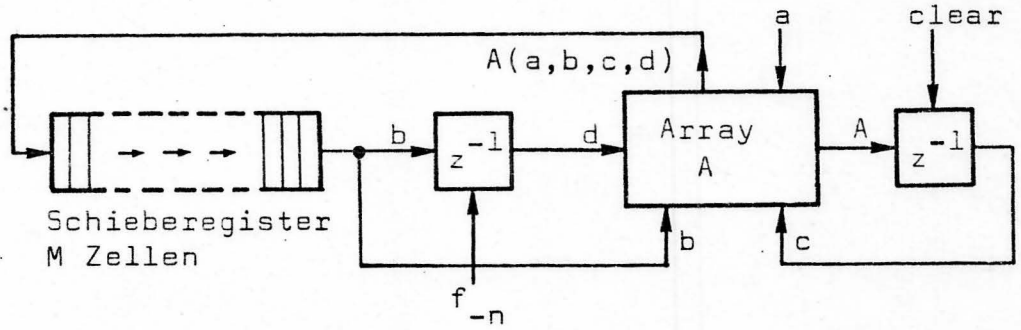


Fig. 8 Blockschema eines möglichen DFV-Rechners

Das Rechenarray A ermittelt aus vier Variablen den Funktionswert

$$A(a,b,c,d) = ab - a*c + d \qquad (31)$$

Die Variablen für die ersten beiden Mikrozyklen sind

$$\tilde{g}_{o,n} = A(a, \tilde{g}_{o,n-1}, 0, f_{-n})$$

$$\tilde{g}_{1,n} = A(a, \tilde{g}_{1,n-1}, 0, \tilde{g}_{o,n-1}) \quad ,$$

für die folgenden

$$g_{k,n} = A(a,\ g_{k,n-1},\ g_{k-1,n},\ g_{k-1,n-1}) \quad ,\quad k= 2,3,\ldots,M \ .$$

Der Inhalt des Schieberegisters wird pro Mikrozyklus um einen Schritt nach rechts verschoben; es hat demnach die Länge M.

Noch ein Wort zum Parameter a. Für eine einfache Frequenzverformung gemäss Fig. 2 ist er reell. Dann kommt im ganzen Rechenablauf keine Vermischung von Real- und Imaginärteilen zustande. Die Grundoperation reduziert sich auf eine reelle Multiplikation

$$A(a,b,c,d) = a(b - c) + d \qquad (32)$$

und der Prozessor kann für reelle Sequenzen ausgelegt werden. Es wird für unsere Zwecke trotzdem vorteilhaft sein, alles komplex zu machen. Ein komplexes Multiplizierwerk ist vom FFT-Prozessor her ohnehin vorhanden. Der DFV-Prozessor wird dem FFT vorgeschaltet und ist punkto Rechenzeit wesentlich ungünstiger als FFT. Komplexe Verarbeitung bringt doppelte Durchflussrate. Der Hauptvorteil aber wurde noch gar nicht erwähnt: nämlich die Möglichkeit, mit einem geeigneten komplexen a in der Umgebung einer beliebigen Frequenz Ω_0 eine Feinauflösung im Spektrum zu erzielen. Diese zusätzliche Eigenschaft ist für den EEGisten hochwillkommen, da z.B. im Alphapeak, wie Dumermuth nachgewiesen hat (11), hochsignifikante Informationen stecken, mit deren Hilfe u.a. auch die genetische Grundlage des EEGs nachgewiesen werden kann.

Die theoretischen Untersuchungen zu dieser interessanten Perspektive der lokalen Feinauflösung, sowie die Simulation des DFV-Prozessors auf einem Minicomputer werden im Anschluss an die Diplomarbeit durchgeführt. Ganz ähnliche Möglichkeiten, die Erzeugung eines Spektrums hoher und gleichmässiger Auflösung über ein schmales Frequenzband, eröffnet die Chirp-z-Transformation. Auch sie soll baldmöglichst in bezug auf ihre Bedeutung für unser Projekt überprüft werden. Gegen Abschluss dieser Arbeit ist ein weiterer Artikel von Oppenheim über DFV erschienen, in welchem die Quantisierungsfehler eingehend untersucht werden (2).

5. LITERATURVERZEICHNIS

1) Berger, H. Ueber das Elektrenkephalogramm des Menschen, I. Arch.Psychiat.Nervenkr., 1929, 87: 527 - 570.

2) Braccini, C., and Oppenheim, A.V. Unequal bandwith spectral analysis using digital frequency warping, IEEE Trans.Acoust.Speech and Signal Process., 1974, ASSP-22, 236 - 244.

3) Cooley, J.W., Lewis, P.A.M., and Welch, P.D. The fast Fourier transform algorithm and its applications. IBM Res. Paper RC-1743, Febr. 1967, 157 p.

4) Corinthios, M.J. A time-series analyzer. Proc.Symp.Comput. Processing in Commun., 1969, 19: 47 - 61. MRI Symposia Ser. New York: Polytechnic Press.

5) DEC: PDP-12 System reference manual, Digital Equipment Corp., 1970.

6) DEC: Laboratory computer handbook 1971, Digital Equipment Corp., 1970, 298 p.

7) DEC: PDP-8/E small computer handbook, Digital Equipment Corp., 1973, PDP-8 handbook series.

8) DEC: Introduction to programming, Digital Equipment Corp., 1973, PDP-8 handbook series.

9) DEC: Logichandbook 1973-74, Digital Equipment Corp., 1973, 520 p.

10) Dumermuth, G. Numerical spectral analysis of the electro-encephalogramm, Handbook of EEG clin. Neurophysiol., 1973, 5A: 33-60.

11) Dumermuth, G. EEG and powerspectra in twins, Helv.pädr.Acta, 1969,

12) Fricker, B. "Carry-save" Multiplizierwerke, (Interne Mitteilung), 1972, 13 p., (siehe Anhang C).

13) Fricker, B. Ein DFT-Prozessor zur mehrkanaligen EEG-Spektral-analyse in Echtzeit ohne Benützung des FFT-Algorithmus, (Interne Mitteilung), 1973, 9p., (siehe Anhang B).

14) Fricker, B. Ein FFT-Prozessor, (Interne Mitteilung), 1973, 28 p., (siehe Anhang A).

15) Fricker, B. Ergebnisse einer Umfrage in neurobiologischen Laboratorien der Region Zürich, Aktenvermerk, 1973, 54 p.

16) Fricker, B. EEG-Spektralanalysator, detailliertes Forschungsprojekt 1974/75, Aktenvermerk, 1974, 18 p.

17) Glisson, T.H., Black, Ch.I., and Sage, A.P. The digital computation of discrete spectra using the Fast Fourier Transform, IEEE Trans.Audio Electroacoust., 1970, AU-18: 271 - 287.

18) Oppenheim, A.V., and Johnson, D.H. Computation of spectra with unequal resolution using the Fast Fourier Transform, Proc.IEEE, 1971, 59: 299 - 301.

19) Oppenheim, A.V., and Johnson, D.H. Discrete representation of signals, Proc.IEEE, 1972, 60: 681 - 691.

20) Oppenheim, A.V., and Weinstein, C.J. Effects of finite register length in digital filtering and the Fast Fourier Transform, Proc.IEEE, 1972, 60: 957 - 976.

21) Pease, M.C. An adaptation of the Fast Fourier Transform for parallel processing, J.Ass.Comput.Mach., 1968, 15: 252 - 264.

22) Peschl, E. Funktionentheorie I, Bibl. Institut, Mannheim, 1967, HTB 131/131a, Kap. I.

23) Shah, A. Binäre Arithmetik, Kap. IX eines Buches über digitale Systeme, (in Vorbereitung).

24) Speiser, A.P. Digitale Rechenanlagen, Springer-Verlag, Berlin, 1965, 454 p.

ANHANG A: EIN FFT-PROZESSOR

Inhaltsübersicht

"It is my great hope that the engineer, the scientific descendants of Wiener and Kolmogorow, will become skilled in the recognition and prediction of the non-linearities of their sick fellow humans and join hands coevally in true biomedical engineering with their physician colleagues in the advance of the healing sciences." [1]

0. EINLEITUNG

Auf dem Weg zur Synthese eines Biosignalprozessors gibt diese Arbeit die Zusammenfassung meines bisherigen Studiums der schnellen Fouriertransformation[2]. Im vorangehenden Aufsatz habe ich gezeigt, dass die Leistungsfähigkeit eines sequentiellen DFT-Prozessors grundsätzlich ausreicht, um eine vielkanalige EEG-Analyse in Echtzeit durchzuführen. Die wachsende Bedeutung der schnellen Fouriertransformation auf dem breiten Feld der statistischen Signalanalyse erheischt jedoch eine genaue Prüfung ihres Wertes für unser Anliegen. Dazu bedarf es zunächst einer geschlossenen Darstellung des FFT-Algorithmus in einer Form, welche den schaltungstechnischen Entwürfen möglichst isomorph ist. PEASE[3] hat einen Matrix-Formalismus entwickelt, der dieser Forderung entspricht. Die formale Eleganz seiner Darstellungsweise hat prompt zu einer Maschinenstruktur geführt, deren Symmetrie und Einfachheit besticht und die sich für unsere Zwecke ausgezeichnet eignet[4]. Man kann sagen, dass der Formalismus von PEASE die heutige MOS-Technologie geradezu antizipiert.

---

[1] John Hanley, M.D., Department of Psychiatry and Space Biology Laboratory, Brain Research Institute UCLA.

[2] engl. Fast Fourier Transform = FFT, Discrete Fourier Transform = DFT

[3] Pease, M.C. An adaptation of the fast Fourier transform for parallel processing. J. Ass. Comput. Mach., 1968, 15: 252 - 264.

BERGLAND[5] gibt vier Klassen möglicher Hardware-Ausführungen des FFT-Algorithmus. Er unterscheidet den sequentiellen Prozessor, den Kaskadenprozessor, den Parallel-Iterativ-Prozessor und den Array-Analyzer. Da wir eine möglichst preisgünstige und einfache Realisierung anstreben, scheiden die drei letzten Klassen aus. Es bleibt der sequentielle Prozessor, dessen Verarbeitungskapazität auf jeden Fall ausreichen wird, da uns ja bereits ein sequentieller DFT-Prozessor zufriedenstellen könnte. Von BERGLAND erfahren wir, dass bis 1970 etwa ein Dutzend verschiedene sequentielle Rechner gebaut wurden. Die typische Struktur der festverdrahteten unter diesen Prozessoren findet man bei SHIVELY[6] und bei KLAHN et al.[7]. Trotz der "in place computation"[8] sind die Speicherkosten hoch, da man auf direkten Zugang zu jedem Speicherplatz angewiesen ist. Als Folge der "in place computation" müssen zudem die Schlussresultate in einem separaten Durchlauf entmischt werden[9].

Beim heutigen Stand der Dinge müssen diese sequentiellen Maschinen als veraltet bezeichnet werden. Es ist - unter Verwendung von MOS-Schieberegistern - möglich, einfacher und preisgünstiger zu bauen. Der in dieser Arbeit vorgestellte Prozessor überrascht in mancher Hinsicht. Er ist einerseits fast ebenso einfach, wie ein DFT-Prozessor; andrerseits kann seine Leistung mit den klassischen sequentiellen Maschinen durchaus verglichen werden.

---

[4] Corinthios, M.J. A time-series analyzer. Proc. Symp. Comput. Processing in Commun., 1969, 19: 47 - 61, MRI Symposia Ser. New York: Polytchnic Press.

[5] Bergland, G.D. Fast Fourier transform hardware implementations. I. An overview. II. A survey. IEEE Trans. Audio Electroacoust., 1969, AU-17: 104 - 119.

[6] Shively, R.R. A digital processor to generate spectra in real time. IEEE Trans. Comput., 1968, C-17: 485 - 491.

[7] Klahn, R., Shively, R.R., Gomez, E. and Gilmartin, M.J. The time-saver: FFT hardware. Electronics, June 1968: 92 - 97.

[8] Für Zwischen- und Schlussresultate ist kein zusätzlicher Speicher nötig, da einmal bearbeitete Datenpaare nicht mehr gebraucht werden und fortlaufend überschrieben werden können. Dies ist ein Merkmal von FFT und steht im Gegensatz zu den Verhältnissen bei DFT.

1. DER FFT - ALGORITHMUS

1.1. Grundidee[10]

Sei $X(j)$, $j = 0,1,2,\ldots,N-1$ eine Folge von N komplexen Zahlen. Dann ist die diskrete[11] Fouriertransformation gegeben durch

$$A(n) = \frac{1}{N} \sum_{j=0}^{N-1} X(j) \exp(-2\pi i n j/N) \qquad (1)$$

wobei $i = (-1)^{\frac{1}{2}}$ ist. Mit der Abkürzung $W_N = \exp(2\pi i/N)$, der N-ten Einheitswurzel, erhält man

$$A(n) = \frac{1}{N} \sum_{j=0}^{N-1} X(j)\, W_N^{-nj} \quad . \qquad (2)$$

Als inverse diskrete Fouriertransformation folgt daraus

$$X(j) = \sum_{n=0}^{N-1} A(n)\, W_N^{nj} \quad . \qquad (3)$$

Dass (2) und (3) tatsächlich ein Transformationspaar darstellt, kann durch Substitution unter Berücksichtigung der Orthogonalitätsrelation

$$\sum_{j=0}^{N-1} W^{nj}\, W^{-mj} = \begin{cases} N, \text{ falls } n = m \bmod N \\ 0, \text{ andernfalls} \end{cases} \qquad (4)$$

leicht gezeigt werden.

Dieser Sachverhalt wird abgekürzt durch die Schreibweise

$$X(j) \longleftrightarrow A(n) \quad . \qquad (5)$$

---

[9] Permutation durch "bit reversing". Bei den binären Adressen werden MSB und LSB vertauscht. Dann wird in aufsteigender Reihenfolge ausgelesen.

[10] Die Darstellung in diesem Abschnitt stützt sich im wesentlichen auf die umfangreiche und genaue Arbeit von Cooley, J.W., Lewis, P.A.M. and Welch, P.D. The fast Fourier transform algorithm and its applications. IBM Res. Paper RC-1743, Febr. 1967, 157 p.

Der Vektor X(j) kann beispielsweise eine Folge von Tastwerten einer komplexen Zeitfunktion darstellen. Dann beschreibt der Vektor A(n) die Amplituden und Phasen der zugehörigen Frequenzwerte. Dies ist bei COOLEY[12] im Detail gezeigt.

Die diskrete Fouriertransformation erfordert N^2 komplexe Operationen, wie aus (2) oder (3) zu entnehmen ist. Eine Operation setzt sich zusammen aus einer komplexen Multiplikation und einer komplexen Addition. In welcher Idee liegt nun der Schlüssel zu der enormen Rechenersparnis der schnellen Fouriertransformation?

Wir betrachten ein komplexes Datenpaket[13] X(j), j = 0,1,2,...,2N-1. Es besteht aus 2N indizierten Punkten. Wir zerlegen es in die Folge der gerade und ungerade indizierten Punkte und dürfen schreiben

$$X(2j) = \sum_{n=0}^{N-1} A_1(n)\, W_N^{jn}, \quad j = 0,1,\ldots,N-1 \qquad (6)$$

und

$$X(2j+1) = \sum_{n=0}^{N-1} A_2(n)\, W_N^{jn}, \quad j = 0,1,\ldots,N-1. \qquad (7)$$

Die Fourierreihe aller 2N Punkte wird gegeben durch

$$X(j) = \sum_{n=0}^{2N-1} A(n)\, W_{2N}^{jn}, \quad j = 0,1,\ldots,2N-1. \qquad (8)$$

Es wird sich zeigen, dass es möglich ist, aus den Koeffizienten $A_1(n)$ und $A_2(n)$ die Koeffizienten A(n) und A(N+n) zu berechnen, und zwar mit einer einzigen komplexen Multiplikation. N komplexe Multiplikationen liefern dann die ganze 2N-Reihe, ein Uebergang, der bei DFT $(2N)^2 - 2N^2 = 2N^2$ komplexe Multiplikationen kosten würde. In dieser Ersparnis liegt der enorme Vorteil der schnellen Fouriertransformation.

---

[11] Wird auch endliche Fouriertransformation genannt.

[12] l.c., p. 30 - 32 und p. 35 - 42.

[13] engl. batch

Berücksichtigt man die Beziehung $W_{2N}^2 = W_N$, dann kann man (8) zerlegen in

$$X(2j) = \sum_{n=0}^{2N-1} A(n)\, W_N^{jn} , \quad j = 0,1,\ldots,N-1 \qquad (9)$$

und

$$X(2j+1) = \sum_{n=0}^{2N-1} A(n)\, W_N^{jn}\, W_{2N}^{n} , \quad j = 0,1,\ldots,N-1. \qquad (10)$$

Wenn wir die Terme mit $n \geq N$ separieren und benützen $W_N^N = 1$ und $W_{2N}^N = -1$, erhalten wir

$$\sum_{n=N}^{2N-1} A(n)\, W_N^{jn} = \sum_{n=0}^{N-1} A(N+n)\, W_N^{jn} \qquad (11)$$

und

$$\sum_{n=N}^{2N-1} A(n)\, W_N^{jn} W_{2N}^{n} = - \sum_{n=0}^{N-1} A(N+n)\, W_N^{jn}\, W_{2N}^{n} . \qquad (12)$$

(11) substituiert in (9) und (12) in (10) und Koeffizientenvergleich mit (6) bzw. (7) liefert das Gleichungspaar

$$A_1(n) = A(n) + A(N+n) , \qquad (13)$$

$$A_2(n) = (A(n) - A(N+n)) \cdot W_{2N}^{n} . \qquad (14)$$

Nach A(n) bzw. A(N+n) aufgelöst, wird daraus

$$\boxed{\begin{aligned} A(n) &= \tfrac{1}{2}\,(A_1(n) + A_2(n)\, W_{2N}^{-n}) \qquad (15)\\ A(N+n) &= \tfrac{1}{2}\,(A_1(n) - A_2(n)\, W_{2N}^{-n}) \qquad (16) \end{aligned}}$$

$$n = 0,1,2,\ldots,N-1.$$

In Worten: Die Koeffizienten A(n) für die erste Hälfte des Frequenzbereichs in der 2N-Punkteserie mit dem Tastintervall $\frac{1}{2}\Delta t$ sind die Mittelwerte der Koeffizienten der zwei N-Punkteserien mit den geraden und ungeraden Indices. Die Multiplikation mit W_{2N}^{-n} bedeutet eine Verschiebung der Punkte mit ungeraden Indices im Zeitbereich um $\frac{1}{2}\Delta t$ nach links. Der neue Satz von Frequenzen, n = N,N+1,...,2N-1, enthält genau die Differenzen der Tastwerte mit geraden und ungeraden Indices. Sind diese alle Null, so kommen keine neuen Frequenzen hinzu. Dies trifft zu, falls das Abtasttheorem[14] schon für Tastintervalle Δt

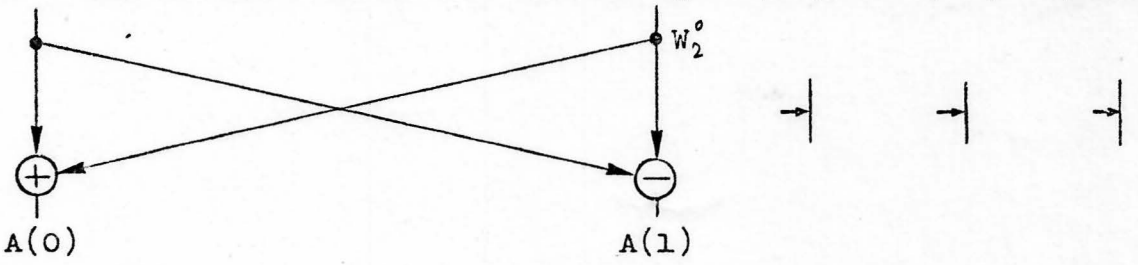
[14] Vgl. Bendat, J.S. and Piersol, A.G. Measurement and analysis of random data. John Wiley,Inc, New York, 1966, 390 p.

erfüllt ist. Dieses Kriterium stellt ferner einen sehr nützlichen Genauigkeitstest dar[15].

Aus den Gleichungen (15) und (16) gewinnt man den FFT-Algorithmus durch folgende Ueberlegungen. Ich wähle zur Veranschaulichung ein Datenpaket von 8 Punkten $X(j)$, $j = 0,1,\ldots,7$, und setze $X(0) = A_1(0)$ und $X(4) = A_2(0)$.

$X(0)$	$X(1)$	$X(2)$	$X(3)$	$X(4)$	$X(5)$	$X(6)$	$X(7)$
$\parallel$				$\parallel$			
$A_1(0)$				$A_2(0)$			

Das entspricht bis auf einen Normierungsfaktor der Fouriertransformation (2) eines einzigen Wertes, also für $N = 1$. $A_1(0)$ und $A_2(0)$ sind die Amplituden der Frequenz Null, d.h. DC-Pegel. Auf dieses Paar werden die Gleichungen (15) und (16) angewandt.



Die Fortsetzung der Operation auf die Paare (1,5), (2,6), (3,7) erzeugt die Reihe

$A(0)$ $A(0)$ $A(0)$ $A(0)$ $A(1)$ $A(1)$ $A(1)$ $A(1)$.

Die Werte der vier je gleich dargestellten Grössen unterscheiden sich natürlich. Man kann das als vier unabhängige Rechenschritte ansehen. Jeder dieser Schritte erzeugt aus zwei $N(=1)$ - Datenpaketen der Frequenz Null ein 2N-Datenpaket der Frequenzen Null und $N(=1)$. Die Reihe wird indiziert.

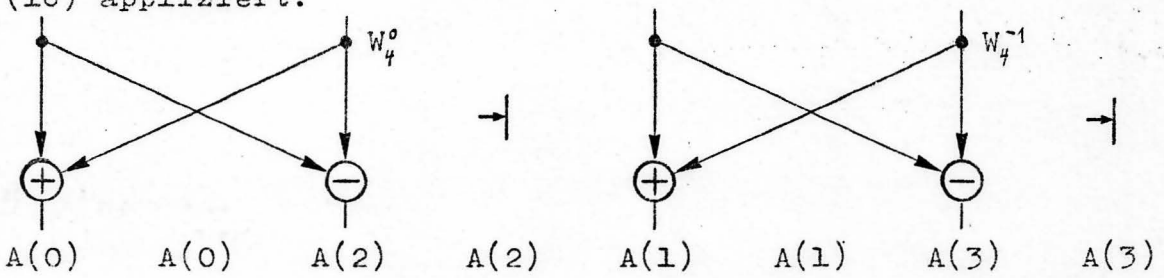
$A_1(0)$ $A_1(0)$ $A_2(0)$ $A_2(0)$ $A_1(1)$ $A_1(1)$ $A_2(1)$ $A_2(1)$

Es ist demnach $N = 2$ zu setzen, da $n = 0,1$.

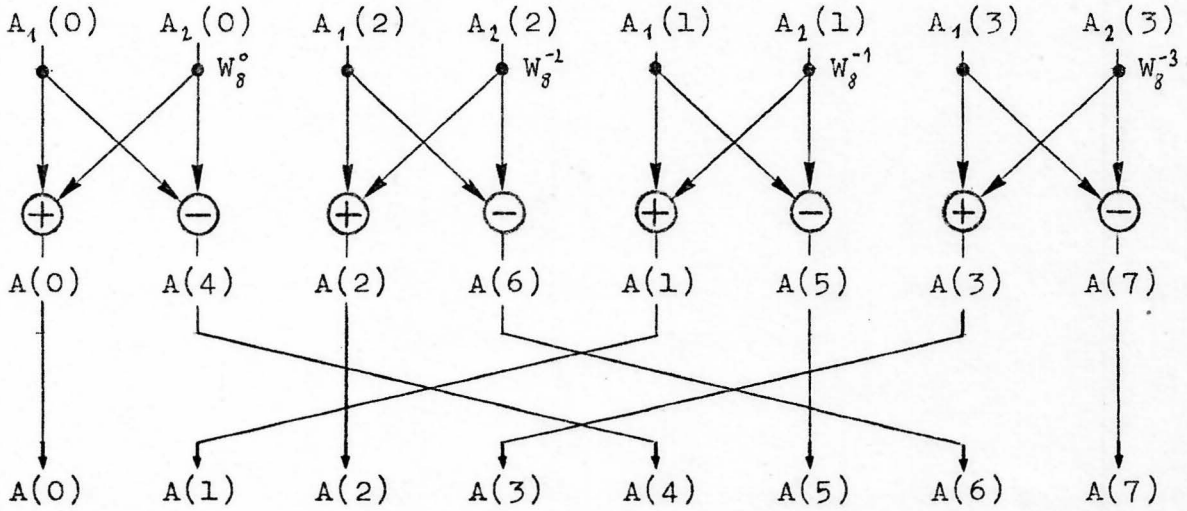
---

[15] Vgl. Cooley, l.c., p. 29 f.

Für n = 0 und n = 1 werden erneut die Operationen (15) und (16) appliziert.



Indizierung und N = 4 gesetzt, liefert



Durch eine abschliessende Permutation werden die Koeffizienten geordnet. Am Schluss ist N = 8, denn n geht von 0 bis 7.

Die Ueberlegungen gelten ganz allgemein für $N = 2$ $(m = 1,2,3,\ldots)$. Dieser Algorithmus erscheint in der Literatur in verschiedenen Modifikationen[16,17,18], die aber keine neuen Gesichtspunkte bringen.

Wir interessieren uns abschliessend für die Zeiteinsparung, die von FFT in Vergleich zu DFT geleistet wird. Die allgemeinen Verhältnisse können von obigem Beispiel leicht abstrahiert werden. Für jede Iterationsebene werden $\frac{N}{2}$ (=4) komplexe Multiplikationen, Additionen und Subtraktionen gebraucht, oder kurz

---

[16] G-AE Subcommittee on Measurement Concepts, What is the fast Fourier transform? IEEE Trans. Audio Electroacoust., 1967, AU-15: 45 - 55; also Proc. IEEE, 1967, 55: 1664 - 1674.

[17] Gold, B. and Rader, C.M. Digital processing of signals, Mc Graw-Hill, Inc., New York, 1969, Chapter 6.

[18] Bertram, S. On the derivation of the fast Fourier transform. IEEE Trans. Audio Electroacoust., 1970, AU-18: p. 55 - 58.

$\frac{N}{2}$ Operationen. Nach $^2\log N$ (=3) Iterationsebenen ist man am Ziel. Insgesamt werden also $\frac{N}{2}{}^2\log N$ (=12) Operationen gebraucht im Vergleich zu N^2(=64) bei DFT. Die Verhältnisse gegenüber DFT werden demnach um einen Faktor

$$G(N) = N^2/(\tfrac{N}{2}{}^2\log N) = 2N/{}^2\log N$$

verbessert.

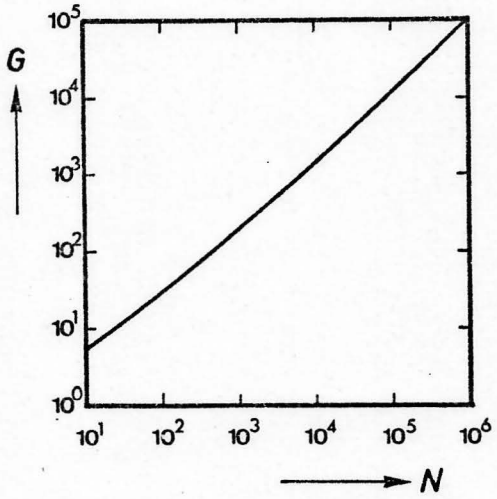


Fig. 1

G(N) ist in Fig. 1 graphisch dargestellt. Der Gewinn wird bei grossen Datenmengen spektakulär, sodass viele Projekte überhaupt erst dank FFT verwirklicht werden können. Man darf dann nicht mehr von einem bloss quantitativen Fortschritt sprechen.

1.2. Eine klassische Darstellung

Im vorangehenden Abschnitt wurde zugunsten der Anschauung auf eine geschlossene Herleitung des FFT-Algorithmus verzichtet. Dies soll hier nachgeholt werden.

Auch bei der schnellen Fouriertransformation führen viele "Wege nach Rom". BERGLAND[19] schreibt: "The number of variations of the FFT algorithm appears to be directly proportional to the number of people using it." - Ich gebe hier eine Herleitung von COOLEY[20].

---

[19] Bergland, G.D. A guided tour of the fast Fourier transform. IEEE Spectrum, 1969, 6: 41 - 52. Dort wird auf zwanzig Arbeiten über FFT-Algorithmen hingewiesen.

[20] l.c., p.19 f.

Er beginnt mit der endlichen Fourierreihe

$$X(j) = \sum_{n=0}^{N-1} A(n)\, W_N^{nj} \qquad (1)$$

wobei $N = 2^M$ und $W_N = \exp(2\pi i/N)$ sein soll.

Es ist gleichgültig, von welcher Transformationsrichtung man ausgeht. Es ist sogar Geschmackssache, wie man die Fourierreihe definiert. Auch hier ist die Literatur nicht einheitlich. Die Ansätze unterscheiden sich hinsichtlich des Normierungsfaktors und des Vorzeichens im Exponenten. Die Form (1) ist aber sehr übersichtlich.

Die Indices werden als Binärzahlen dargestellt.

$$j = j_{M-1}\cdot 2^{M-1} + \ldots + j_1 \cdot 2 + j_0\,, \quad j_L = 0 \text{ oder } 1,$$

$$n = n_{M-1}\cdot 2^{M-1} + \ldots + n_1 \cdot 2 + n_0\,, \quad n_L = 0 \text{ oder } 1.$$

$X(j)$ ist eine Funktion von M Argumenten $j_{M-1}, \ldots, j_1, j_0$. Ebenso ist $A(n) = A(n_{M-1}, \ldots, n_1, n_0)$. Mit dieser Notation muss die Fourierreihe (1) folgendermassen geschrieben werden.

$$X(j_{M-1},\ldots,j_1,j_0) = \sum_{n_0=0}^{1}\sum_{n_1=0}^{1}\cdots\sum_{n_{M-1}=0}^{1}\Big\{A(n_{M-1},\ldots,n_1,n_0)\cdot W_N^{j(n_{M-1}\cdot 2^{M-1}+\ldots+n_1\cdot 2+n_0)}\Big\} \qquad (2)$$

Mit $W_N^{2^M} = W_N^N = 1$ wird

$$W_N^{j\cdot 2^{M-L}} = W_N^{(j_{L-1}\cdot 2^{L-1}+\ldots+j_1\cdot 2+j_0)\cdot 2^{M-L}}, \qquad (3)$$

da Terme im Exponenten, die Multipla von $2^M = N$ sind, einfach fallengelassen werden können.

Jetzt werden die Summen (2) iterativ aufgebaut. Man summiert zunächst über n_{M-1} und schreibt unter Berücksichtigung von (3)

$$A_1(j_o, n_{M-2}, \ldots, n_1, n_o) = \sum_{n_{M-1}=0}^{1} A(n_{M-1}, \ldots, n_1, n_o)\; W_N^{j_o \cdot n_{M-1} 2^{M-1}} \qquad (4)$$

Diese erste Iteration liefert ein Datenfeld A_1, das von den Parametern $j_o, n_{M-2}, \ldots, n_1, n_o$ abhängt, das also die Dimension N besitzt. Bei der folgenden Iteration über n_{M-2} werden die Koeffizienten A_1 verwendet.

$$A_2(j_o, j_1, n_{M-3}, \ldots, n_1, n_o) = \sum_{n_{M-2}=0}^{1} A_1(j_o, n_{M-2}, \ldots, n_1, n_o) \cdot W_N^{(j_1 \cdot 2 + j_o) \cdot n_{M-2} 2^{M-2}} \qquad (5)$$

Es entsteht ein Datenfeld A_2 derselben Dimension. Die L-te Iteration lautet

$$\boxed{\begin{aligned} &A_L(j_o, \ldots, j_{L-2}, j_{L-1}, n_{M-L-1}, \ldots, n_o) \\ &= \sum_{n_{M-L}=0}^{1} A_{L-1}(j_o, \ldots, j_{L-2}, n_{M-L}, n_{M-L-1}, \ldots, n_o) \cdot W_N^{(j_{L-1} 2^{L-1} + \ldots + j_o) \cdot n_{M-L} 2^{M-L}} . \end{aligned}} \qquad (6)$$

Zum Schluss entsteht ein Feld $A_M(j_o, \ldots, j_{M-1})$, das in permutierter Form die Koeffizienten $X(j)$ enthält.

Es ist $X(j_{M-1}, \ldots, j_o) = A_M(j_o, \ldots, j_{M-1})$, also ein einfacher Zusammenhang. Die Koeffizienten A_M müssen "bit-reversed" herausgelesen werden.

Man beachte, dass die Parameter $j_o, \ldots, n_o$ der Koeffizienten A_L deren Speicheradressen darstellen. Da ein bestimmter Koeffizient A_{L-1} bloss einmal gebraucht wird, darf er durch das Resultat A_L überschrieben werden. Durch die Indizierung ist dieser Vorgang festgelegt. Man spricht von "in place computation"; damit wird der Aufwand an Speicherplatz auf die Hälfte reduziert. Neben der Zeitersparnis ist das der wichtigste Vorteil der schnellen Fouriertransformation.

BERGLAND[21] gibt eine Veranschaulichung des Algorithmus (6) für N = 8, was wegen der unübersichtlichen Indizierung nicht

---

[21] A guided tour, l.c., p.42 f.

ganz überflüssig ist.

Die Darstellung (6) der schnellen Fouriertransformation ist für Softwareprogrammierung recht interessant, da sie die Speicheradressierung eindeutig festlegt. Sie ist für die Entwicklung eines festverdrahteten Prozessors aber bedeutend weniger geeignet, als die im nächsten Abschnitt entwickelte Operator-Schreibweise.

1.3 Eine Matrix - Darstellung

Der Matrix-Formalismus wurde erstmals von PEASE[22] auf den Cooley-Tuckey-Algorithmus angewandt. PEASE zeigt, dass drei Operatoren genügen, um den Prozess durchzuführen. Ein Operator ***S*** ersetzt äquidistante Paare von Datenpunkten durch ihre Summe und ihre Differenz[23]. Der zweite Operator ***M*** multipliziert die "untere" Hälfte des Datenvektors mit den komplexen Gewichten. Der dritte Operator ***P*** leistet eine bestimmte Permutation und entmischt die Koeffizienten fortwährend. Die Operatoren sind NxN-Matrizen.

Wir folgen der Darstellung von CORINTHIOS[24], einer direkten Weiterentwicklung des Formalismus von PEASE für sequentielle Maschinen[25]. Wiederum soll von der endlichen Fourierreihe ausgegangen werden.

$$X(j) = \sum_{n=o}^{N-1} A(n)\, W_N^{nj}\,, \quad j = 0,1,\ldots,N-1. \qquad (1)$$

Die Konventionen bleiben unverändert. Es ist gleichgültig,

---

[22] l.c.

[23] Ich habe die Operatoren durch DIN-Buchstaben dargestellt.

[24] l.c.

[25] Die Anwendung des Formalismus von PEASE auf Parallel-Prozessoren gibt der Autor selber in
Pease, M.C. Organization of large scale Fourier processors. J. Ass. Comput. Mach., 1969, 16: 474 - 482.

ob man (1) als Hin- oder Rücktransformation auffasst.

Wir bezeichnen die Vektoren mit

$$\vec{x} \equiv \begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ \cdot \\ \cdot \\ \cdot \\ X(N-1) \end{bmatrix} , \qquad \vec{a} \equiv \begin{bmatrix} A(0) \\ A(1) \\ A(2) \\ \cdot \\ \cdot \\ \cdot \\ A(N-1) \end{bmatrix} ,$$

und definieren die Transformationsmatrix T_N durch ihre komplexen Elemente

$$(T_N)_{nj} = W_N^{nj} = \exp(2\,\overline{\imath}\,inj/N) \ .$$

Beide Indices laufen von 0 bis N-1; es handelt sich also um eine quadratische Matrix der Dimension $N = 2^M$.

Weiter definieren wir eine Permutationsmatrix P_k durch

$$P_k \cdot \begin{bmatrix} X_0 \\ X_1 \\ \cdot \\ \cdot \\ \cdot \\ X_{k/2-1} \\ X_{k/2} \\ \cdot \\ \cdot \\ \cdot \\ X_{k-1} \end{bmatrix} \equiv \begin{bmatrix} X_0 \\ X_{k/2} \\ X_1 \\ X_{k/2+1} \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ X_{k/2-1} \\ X_{k-1} \end{bmatrix} , \quad k \text{ sei gerade.}$$

Damit ist ein inverser Operator P_k^{-1} festgelegt mit der Eigenschaft, gerade und ungerade indizierte Elemente zu separieren.

$$P_k^{-1} \cdot \begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ X_{k-1} \end{bmatrix} \equiv \begin{bmatrix} X_0 \\ X_2 \\ X_4 \\ \cdot \\ \cdot \\ \cdot \\ X_{k-2} \\ X_1 \\ X_3 \\ X_5 \\ \cdot \\ \cdot \\ X_{k-1} \end{bmatrix} .$$

Per definitionem gilt $P_K \cdot P_k^{-1} = I_k$, wo I_k die Einheitsmatrix ist. Man darf schreiben

$$T_N = P_N \cdot P_N^{-1} \cdot T_N \equiv P_N \cdot T_N' \tag{2}$$

und definiert damit eine neue Matrix $T_N' \equiv P_N^{-1} \cdot T_N$.

Der iterative Charakter des FFT-Algorithmus zeigt sich im Matrixbild in der Faktorisierbarkeit der Matrix T_N' . Um dies zu leisten, definieren wir

$$K_{N/2}^1 \equiv \begin{bmatrix} 1 & & & & \\ & W_N^1 & & 0 & \\ & & W_N^2 & & \\ & & & W_N^3 & \\ & 0 & & \cdot & \\ & & & \cdot & W_N^{\frac{N}{2}-1} \end{bmatrix} \tag{3}$$

und bezeichnen mit $\emptyset$ die Nullmatrix der entsprechenden Dimension.

Es ist möglich, T_N' folgendermassen zu zerlegen

$$T_N' = \left[\begin{array}{c|c} T_{N/2} & T_{N/2} \\ \hline T_{N/2} \cdot K_{N/2}^1 & -T_{N/2} \cdot K_{N/2}^1 \end{array}\right] . \tag{4}$$

Mit (2) wird

$$T_N = P_N \cdot \left[\begin{array}{c|c} T_{N/2} & \emptyset \\ \hline \emptyset & T_{N/2} \end{array}\right] \cdot \left[\begin{array}{c|c} I_{N/2} & I_{N/2} \\ \hline K_{N/2}^1 & -K_{N/2}^1 \end{array}\right] \tag{5}$$

$$T_N = P_N \cdot \left[\begin{array}{c|c} T_{N/2} & \emptyset \\ \hline \emptyset & T_{N/2} \end{array}\right] \cdot \left[\begin{array}{c|c} I_{N/2} & \emptyset \\ \hline \emptyset & K_{N/2}^1 \end{array}\right] \cdot \left[\begin{array}{c|c} I_{N/2} & I_{N/2} \\ \hline I_{N/2} & -I_{N/2} \end{array}\right] \tag{6}$$

Diese Zerlegung wird (M-1)mal wiederholt, dann ist

$$T_{N/2^{M-1}} = T_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

Unter Verwendung des Kronecker-Produkts[26] kann die Zerlegung von T_N' sehr elegant dargestellt werden.

Das Kronecker-Produkt für Matrizen ist wie folgt definiert. Es seien zwei quadratische Matrizen

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdot & \cdot & a_{1n} \\ a_{21} & a_{22} & \cdot & \cdot & a_{2n} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ a_{n1} & a_{n2} & \cdot & \cdot & a_{nn} \end{bmatrix}, \qquad B = \begin{bmatrix} b_{11} & b_{12} & \cdot & \cdot & b_{1m} \\ b_{21} & b_{22} & \cdot & \cdot & b_{2m} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ b_{m1} & b_{m2} & \cdot & \cdot & b_{mm} \end{bmatrix}$$

der Dimensionen n und m gegeben. Die Matrix $C = A \times B$ heisst Kronecker-Produkt von A und B , falls für ihre Elemente gilt

$$(C)_{ji,\ell k} = a_{ik} \cdot b_{j\ell} \quad .$$

Das Paar (ij) spielt die Rolle des ersten, (lk) die Rolle des zweiten Index von C , wobei

i und k = 1,2,..,n,

j und l = 1,2,...,m .

An einem Beispiel wird die einfache Bedeutung diese Produktes sofort klar.

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \quad , \qquad B = \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} ,$$

$$C = \left[\begin{array}{cc|cc} a_{11}\cdot b_{11}, & a_{12}\cdot b_{11} & a_{11}\cdot b_{12}, & a_{12}\cdot b_{12} \\ a_{21}\cdot b_{11}, & a_{22}\cdot b_{11} & a_{21}\cdot b_{12}, & a_{22}\cdot b_{12} \\ \hline a_{11}\cdot b_{21}, & a_{12}\cdot b_{21} & a_{11}\cdot b_{22}, & a_{12}\cdot b_{22} \\ a_{21}\cdot b_{21}, & a_{22}\cdot b_{21} & a_{21}\cdot b_{22}, & a_{22}\cdot b_{22} \end{array}\right] = \begin{bmatrix} c_{11,11} & c_{11,12} & c_{11,21} & c_{11,22} \\ c_{12,11} & c_{12,12} & c_{12,21} & c_{12,22} \\ c_{21,11} & c_{21,12} & c_{21,21} & c_{21,22} \\ c_{22,11} & c_{22,12} & c_{22,21} & c_{22,22} \end{bmatrix}$$

Die Dimension von C ist demnach n.m .

---

[26] Vgl. Pease, M.C. Methods of Matrix Algebra. Academic Press, New York, 1965, Ch. XIV.
Das Kronecker-Produkt von Matrizen erscheint in der Literatur auch unter dem Namen direktes Produkt.

Gleichung (6) kann jetzt bündiger dargestellt werden.

$$T_N = P_N (T_{N/2} \times I_2) D_N (I_{N/2} \times T_2) \qquad (7)$$

D_n ist eine quasidiagonale NxN-Matrix,

$$D_n \equiv \left[\begin{array}{c|c} I_{N/2} & \emptyset \\ \hline \emptyset & K'_{N/2} \end{array}\right]$$

Allgemein definieren wir mit $k = 2^i$ $(i = 0,1,2,\ldots)$

$$D_{N/k} \equiv \left[\begin{array}{c|c} I_{N/2k} & \emptyset \\ \hline \emptyset & K^k_{N/2k} \end{array}\right]$$

und in Verallgemeinerung von Definition (3)

$$K^k_{N/2k} \equiv \begin{bmatrix} W^0_{N/k} & & & & & 0 \\ & W^k_{N/k} & & & & \\ & & W^{2k}_{N/k} & & & \\ & & & W^{3k}_{N/k} & & \\ & & & & \ddots & \\ 0 & & & & & W^{N/2-k}_{N/k} \end{bmatrix}$$

Man beachte, dass die unteren Indices an den Matrixsymbolen immer die jeweilige Dimension der Matrix angeben.

Der zweite Schritt der Zerlegung von T'_N heisst

$$T_{N/2} = P_{N/2} (T_{N/4} \times I_2) D_{N/2} (I_{N/4} \times T_2) \quad . \qquad (8)$$

Vergleiche das mit (7)! Allgemein dürfen wir mit $k = 2^i$ $(i = 0,1,2,\ldots,M-1)$ und $N = 2^M$ schreiben

$$T_{N/k} = P_{N/k} (T_{N/2k} \times I_2) D_{N/k} (I_{N/2k} \times T_2) \quad . \qquad (9)$$

Nach M Iterationen ist man fertig, $T_{N/k}$ ist dann eine 2x2-Matrix.

Für N = 8 gibt das die bekannten drei Iterationen. Fortgesetzte Substitution mit Anfang bei (7) liefert die geschlossene Darstellung

$$T_N = P_{N/2}\,[\{P_{N/2}\,[\{\ \dots\ P_{N/k}\,[\{\ \dots[\{P_4\ (T_2 \times I_2)\,D_4\,(I_2 \times T_2)\} \times I_2\,]\dots$$

$$\dots\}\times I_2\,]\,D_{N/k}(I_{N/2k}\times T_2)\dots\}\times I_2\,]\,D_{N/2}(I_{N/4}\times T_2)\}\times I_2\,]\,D_N\,(I_{N/2}\times T_2)\ . \qquad (10)$$

In dieser Form ist das Ziel noch nicht ganz erreicht. Für Kronecker-Produkte gibt es keine einfache Hardware-Uebersetzung. Wenn es gelingt, die Kronecker-Produkte in gewöhnliche Matrix-Produkte umzuwandeln, sieht die Sache anders aus. Das ist in der Tat möglich.

Für eine beliebige Matrix $A_{k/2}$ gilt die Beziehung

$$P_k\,(A_{k/2}\times I_2)\,P_k^{-1} = I_2 \times A_{k/2} \qquad (11)$$

beziehungsweise

$$P_k\,(A_{k/2}\times I_2) = (I_2 \times A_{k/2})\,P_k\ . \qquad (12)$$

Die Faktoren $P_{N/k}$ können damit in (10) nach rechts gezogen werden.

$$T_N = [\,I_2 \times \{[\,\overbrace{I_2 \times \{\dots[I_2 \times \{\dots\ [I_2 \times \{(I_2 \times T_2)\,P_4\,D_4\,(I_2 \times T_2)\}]\dots$$

$$\dots\ ]\,P_{N/k}\,D_{N/k}(I_{N/2k}\times T_2)\ \dots\ ]\,P_{N/2}\,D_{N/2}(I_{N/4}\times T_2)\}]\,P_N\,D_N\,(I_{N/2}\times T_2)\ . \qquad (13)$$

(M-2)-mal

Es ist allgemein[27]

$$I \times (A\,B\,C\ \dots) = (I \times A)(I \times B)(I \times C)\ \dots$$

und

$$\overbrace{I_2 \times (I_2 \times (I_2 \times \dots (I_2 \times A)\dots))}^{k\text{-mal}} = I_{2^k} \times A\ .$$

---

[27] Vgl. Pease, l.c.

[28] Bei P_i' ist i ein Laufindex und nicht die Dimension. Dasselbe gilt für M_i. Die Dimensionen dieser Matrizen sind N.

Damit folgt

$$T_N = (I_{N/2} \times T_2)(I_{N/4} \times P_4)(I_{N/4} \times D_4)(I_{N/2} \times T_2) \ldots$$

$$\ldots (I_k \times P_{N/k})(I_k \times D_{N/k})(I_{N/2} \times T_2) \ldots \qquad (14)$$

$$\ldots (I_2 \times P_{N/2})(I_2 \times D_{N/2})(I_{N/2} \times T_2) P_N \; D_N \; (I_{N/2} \times T_2) \quad .$$

Wir definieren die Matrizen

$$S \equiv I_{N/2} \times T_2 \quad ,$$

$$P_i' \equiv I_{2^{i-1}} \times P_{N/2^{i-1}} \qquad [28]$$

und $$M_i \equiv I_{2^{i-1}} \times D_{N/2^{i-1}} \; .$$

Man beachte, dass $P_M' = M_M' = I_N$ ist $(N = 2^M)$.

Für die Gleichung (14) können wir schlussendlich schreiben

$$T_N = S \cdot P_{M-1}' \cdot M_{M-1} \cdot S \cdot \ldots \cdot P_2' \cdot M_2 \cdot S \cdot P_1' \cdot M_1 \cdot S \qquad (15)$$

In abgekürzter Schreibweise ist das

$$\boxed{T_N = \prod_{i=M}^{1} (P_i' \; M_i \; S)} \quad . \qquad (16)$$

Wir haben damit eine Faktorisierung der Transformationsmatrix T_N gefunden, die äusserst einfach in eine sequentielle Hardware-Struktur übersetzt werden kann. Das ist im nächsten Abschnitt dargestellt. Der Permutationsoperator P_i' wird bei jeder Iteration verändert. Durch diese Anpassung erreicht man eine kontinuierliche Entmischung der Vektorpunkte. CORINTHIOS[29] gibt noch eine zweite Faktorisierung an, die mit einem festen Permutationsoperator auskommt. Dafür muss nach der Transformation separat entmischt werden. Wir wollen diese Möglichkeit aber nicht weiter verfolgen, da sie für uns weniger interessant ist.

---

[29] l.c.

2. DER FFT - PROZESSOR

2.1. Elektronische Nachbildung des Algorithmus

Die Ergebnisse von Abschnitt 1.3. können wie folgt zusammengefasst werden.

Von der diskreten Fourierreihe

$$X(j) = \sum_{n=0}^{N-1} W_N^{nj} \cdot A(n) \quad , \quad j = 0,1,\ldots,N-1, \qquad (1)$$

in Summenschreibweise ausgehend, findet man nach geeigneten Definitionen deren Matrixschreibweise

$$\vec{x} = T_N \cdot \vec{a} \quad . \qquad (2)$$

Die Transformationsmatrix T_N kann - und das ist wesentlich - faktorisiert werden,

$$T_N = \prod_{i=M}^{1} (P_i' M_i S) \quad , \quad N = 2^M \quad , \qquad (3)$$

was eine iterative Durchführung der Transformation (2) ermöglicht. Die Iteration zerfällt in M Schritte, die ihrerseits aus drei Operatoren bestehen.

Der erste, der Summe/Differenz-Operator S , verändert sich nicht. Der zweite, der Multiplikations- oder Gewichtsoperator M_i , wird nach jedem Zyklus auf einfache Weise abgeändert. Dasselbe geschieht mit dem Permutationsoperator P_i' .

Ein Beispiel für N = 8 soll den Algorithmus veranschaulichen. Auf Grund der in 1.3. aufgestellten Definitionen finden wir

$$\vec{x} = T_8 \cdot \vec{a}$$

$$= S P_2' M_2 S P_1' M_1 S \cdot \vec{a}$$

$$= \begin{bmatrix} 1 & & 1 & & & & & \\ & 1 & & 1 & & & & \\ 1 & & -1 & & & & & \\ & 1 & & -1 & & & & \\ & & & & 1 & & 1 & \\ & & & & & 1 & & 1 \\ & & & & 1 & & -1 & \\ & & & & & 1 & & -1 \end{bmatrix} \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & W^0 & \\ & & & W^2 \end{bmatrix} \begin{bmatrix} 1 & & 1 & \\ & 1 & & 1 \\ 1 & & -1 & \\ & 1 & & -1 \end{bmatrix} \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & W^0 & \\ & & & W^2 \end{bmatrix} \begin{bmatrix} 1 & & 1 & \\ & 1 & & 1 \\ 1 & & -1 & \\ & 1 & & -1 \end{bmatrix} \begin{bmatrix} A_0 \\ A_1 \\ A_2 \\ A_3 \\ A_4 \\ A_5 \\ A_6 \\ A_7 \end{bmatrix} \quad (4)$$

Die weiss gelassenen Stellen bedeuten Nullen. Ein Normierungsfaktor $1/N = \frac{1}{2}^M$ würde gegebenenfalls als konstanter Faktor $\frac{1}{2}$ zwischen die M Zyklen geschoben.

Der Schritt von dieser Abstraktionebene zur Hardware-Darstellung liegt beinahe auf der Hand. Fig. 2 zeigt eine denkbar einfache Anordnung, die eine Fouriertransformation leistet.

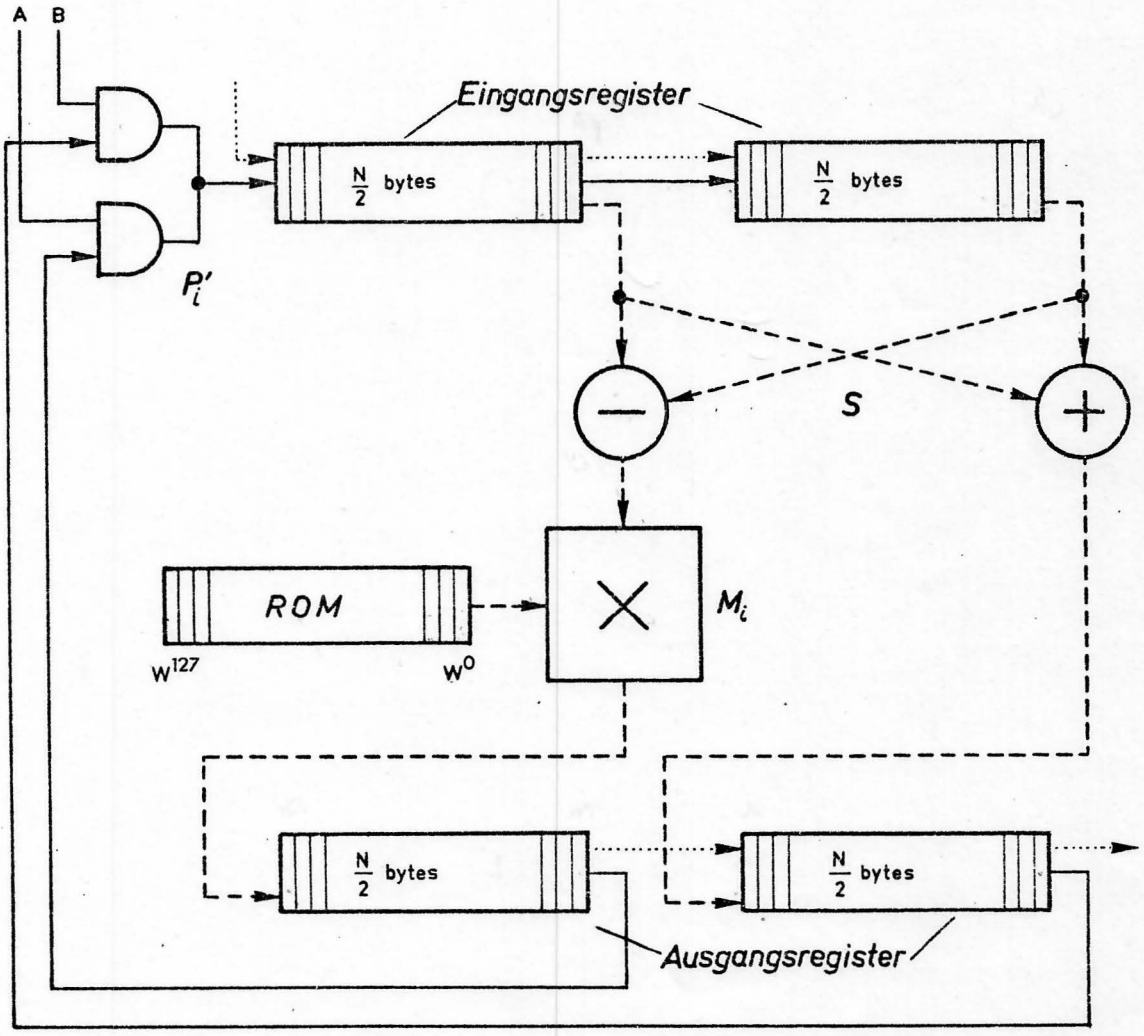


Fig. 2

Ueber den punktierten Pfad .....► läuft der Datenvektor $\vec{a}$ ins Eingangsregister. Gleichzeitig läuft der vorgängige Resultatvektor $\vec{x}$ über .....► aus dem Ausgangsregister zur Weiterverarbeitung in den Laborcomputer. Wir wollen diesen ersten Betriebszustand mit Transferphase bezeichnen.

Als nächstes werden die gestrichelten Wege aktiviert. Die komplexen bytes am Ausgang und Mittelabgriff des Eingangsregisters werden zunächst dem Summen/Differenz-Operator S unterworfen. Parallel-Addierwerke sorgen für einen raschen Durchlauf. Der Normierungsfaktor $\frac{1}{2}$ würde ebenfalls hier appliziert. Der linke Ast läuft über ein Parallel-Multiplizierwerk, das die komplexe Gewichtung mit Einheitswurzeln bewerkstelligt. Beide Aeste münden im Ausgangsregister. Nach N/2 Takten stehen im Eingangsregister lauter Nullen, im Ausgangsregister der Vektor $\vec{a}_1$ der ersten Iterationsfolge.

Jetzt wird auf den ausgestrichenen Pfad geschaltet. Eine N-fache Clockpulsserie schiebt das Paket durch das Permutationsgatter P_i' , das von einer Kontrollogik, wie ich sie in Fig. 3 vorschlage, beherrscht wird. Der Vektor läuft geordnet im Eingangsregister ein und steht für den nächsten Zyklus bereit.

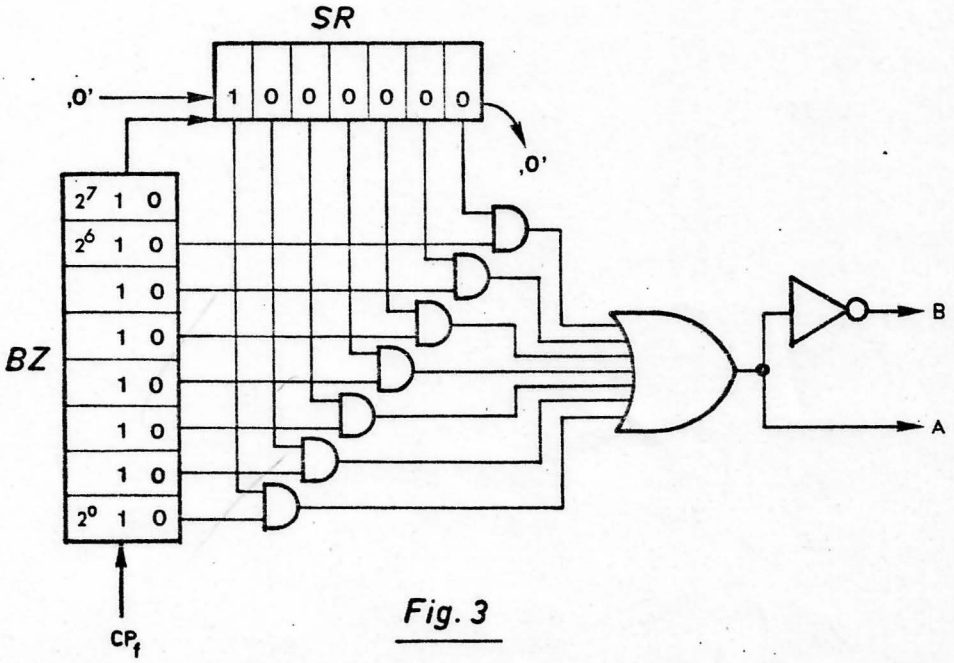


Fig. 3

Wir unterscheiden also drei Betriebszustände; die Transferphase (.....►), die Processphase (----------►) und die Feedbackphase (——————►). Die drei Phasen haben möglicherweise ihre je eigenen Clocks, CP_t, CP_p, CP_f, die auf die Arbeitsgeschwindigkeit der aktivierten Bezirke abgestimmt sind.

Die Steuerlogik der Permutationsgatter (Fig. 3) wird demnach von CP_f dirigiert. Ein Binärzähler BZ untersetzt die einlaufenden N Pulse[30]. Im Schieberegister SR indiziert die Stellung der Marke 1 die Zyklusnummer. Die Marke 1 springt, wenn eine Permutation abgeschlossen ist. Bei der ersten Permutation schalten die Tore A und B nach jedem Clock CP_f um; bei der zweiten nach jedem zweiten CP_f; bei der dritten nach jedem vierten CP_f; etc. Man versichere sich anhand des Beispiels für N = 8, dass die Permutationen so geleistet werden.

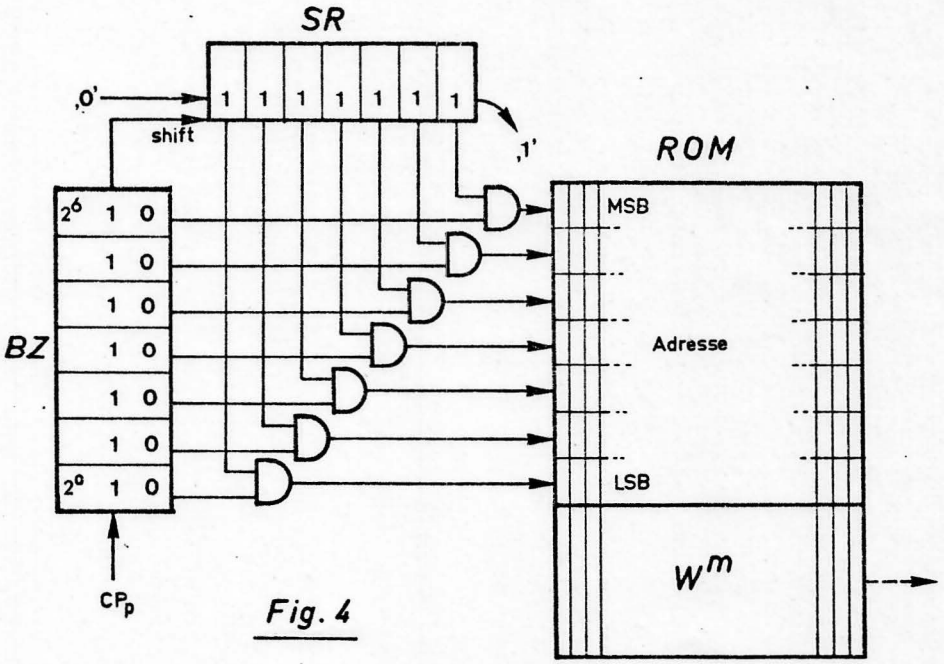


Fig. 4

[30] Die Figuren entsprechen unserer Vergleichsgrundlage N = 256.

Die ROM-Adressierung entscheidet über die korrekte Gewichtung. Sie ist in Fig. 4 dargestellt. Das Prinzip wird unverändert übernommen, nur genügen hier N/2 Clocks zur Erzeugung eines Overflows aus dem Binärzähler BZ. Das Schieberegister SR ist anfangs mit einer Serie Einern gefüllt. Alle AND's sind dann geöffnet. Die Adresse entspricht der Position des Binärzählers.

Im zweiten Zyklus werden alle Gewichte mit geradem Exponenten erzeugt; im dritten Durchlauf stehen Gewichte mit Exponenten aus der 4-er Reihe an; etc.

Für M Processzyklen werden M.N/2 Processclocks der Frequenz f_p gebraucht. M Feedbackzyklen benötigen M.N Feedbackclocks der Frequenz f_f. Es kommen überdies N Transferimpulse der Frequenz f_t hinzu.

Setzen wir $f_f = f_t = 5$ MHz und $f_p = 1$ MHz, dann wird die Transformationszeit in /usec

$$t(N) = M.(N/2).(1/f_p) + (M+1).N.(1/f_f)$$

$$= 8.128.1 + (8+1).256.0,2 = 1485 \text{ /us}$$

Eine Fouriertransformation von N = 256 komplexen Punkten braucht 1,5 msec. Um besser vergleichen zu können, berechnet man in der Gleichung $t(N) = k \cdot N \cdot {}^2\log N$ den Proportionalitätsfaktor k in /usec. Er beträgt mit obenstehenden Annahmen approximativ 0,7 /usec. Ein Blick auf die Vergleichstabelle von BERGLAND[31] zeigt, dass wir ausgezeichnet liegen! Unser Prozessor ist so schnell wie die besten (und teuersten) der dort vorgestellten Maschinen. Ein Blick auf den brandneuen HEWLETT-PACKARD 5470A Fast Fourier Processor, der als extrem schnelle Maschine angepriesen wird, ist geradezu aufreizend. Sein k beträgt 1,5 /usec.

---

[31] FFT implementations, l.c., p.114 f.

2.2. Verarbeitung reeller Zeitreihen

Der in Fig. 2 skizzierte FFT-Prozessor arbeitet aus Symmetriegründen durchwegs im komplexen Zahlenbereich. Er transformiert einen komplexen Vektor $\vec{a}$ auf einen komplexen Vektor $\vec{x}$, oder umgekehrt. Andrerseits ist jede physikalische Grösse zunächst reell; das gilt auch für elektrische Potentialschwankungen der Schädeldecke, die uns hier besonders interessieren. Darin besteht aber keineswegs ein Widerspruch, wie ich im folgenden zu zeigen versuche.

Zunächst ist festzuhalten, dass uns die komplexe Arbeitsweise des Prozessors hochwillkommen ist. Wir werden es in der statistischen Signalanalyse nur in Spezialfällen mit reellen Eingangsvektoren zu tun haben. Denn die Anwendung der diskreten Fouriertransformation erschöpft sich längst nicht in der Ueberführung reeller Zeitreihen in den Frequenzbereich. Die Fouriertransformation ist ein Gefährt zu beliebigen statistischen Höhenflügen und findet auch ausserhalb der Signalverarbeitung mannigfaltige Anwendung. In der Mehrzahl dieser Fälle ist die komplexe Arbeitsweise eine Conditio sine qua non. Aus diesem Grunde ist eine Spezialisierung der Maschine auf reellen Eingang, wie sie zum Beispiel BERGLAND[32] vorschlägt, nicht empfehlenswert.

COOLEY[33] zeigt hingegen einen einfachen Zusatzalgorithmus, der unsere Maschine im Falle reeller Eingangswerte vor der Qualifikation Speicherverschwendung bewahrt. Das Eingangsregister wird nicht bloss mit einem reellen Datenvektor der Dimension N gefüllt, sondern mit deren zwei $X_1(j)$ und $X_2(j)$. $X_1(j)$ wird in die reellen, $X_2(j)$ in die imaginären byte-Positionen geschoben.

---

[32] Bergland, G.D. A fast Fourier transform algorithm for real-valued series. Commun. ACM, 1968, 11: 703 - 710.

[33] l.c., p.27 ff.

Es ist $X_1(j) \longleftrightarrow A_1(n)$

$X_2(j) \longleftrightarrow A_2(n)$,

und wir erhalten $X(j) \longleftrightarrow A(n)$, wobei

$$X(j) = X_1(j) + iX_2(j)$$

und $$A(n) = A_1(n) + iA_2(n) \quad . \tag{1}$$

Wir möchten aus dem Resultat $A(n)$ die Partialvektoren $A_1(n)$ und $A_2(n)$ zurückgewinnen, die als Fouriertransformierte der Eingangswerte $X_1(j)$ und $X_2(j)$ anzusehen sind.

Da $X_1(j)$ und $X_2(j)$ reell sind, erfüllen sie die Symmetrieeigenschaften

$$A_1(n) = \tilde{A}_1(N-n)$$

$$A_2(n) = \tilde{A}_2(N-n) \quad .^{34} \tag{2}$$

Daraus folgt bereits, dass es genügt, $A_1(n)$ und $A_2(n)$ für $n = 0,1,2,\ldots,\frac{N}{2}-1$ zu kennen. Wir ersetzen in (1) n durch N-n, nehmen von beiden Seiten das konjugiert Komplexe und erhalten mit der Eigenschaft (2)

$$\tilde{A}(N-n) = \tilde{A}_1(N-n) - i\tilde{A}_2(N-n) = A_1(n) - iA_2(n) \quad . \tag{3}$$

(1) und (3) kann man auflösen nach

$$\boxed{\begin{aligned} A_1(n) &= \frac{1}{2}[A(n) + \tilde{A}(N-n)] \\ A_2(n) &= \frac{1}{2i}[A(n) - \tilde{A}(N-n)] \end{aligned}} \tag{4}$$

$$(n = 0,1,2,\ldots,\frac{N}{2} - 1) \quad .$$

(4) kann im Anschluss an die Fouriertransformation entweder im Laborcomputer berechnet werden, falls der Processor als Interface gebaut wird, oder, falls er eine selbständige Einheit darstellt, durch einen vierten Betriebszustand unter Verwendung

---

34 $\tilde{A}$ bedeutet das konjugiert Komplexe von A.

der Addierwerke S. Allerdings müsste, wie aus (4) hervorgeht, die Schieberichtung einer Registerhälfte gewechselt werden, was die Verwendung dynamischer Schieberegister ausschliesst. Die Hardwarefrage muss aber ohnehin noch genauer abgeklärt werden.

Der Prozessor transformiert also entweder N komplexe Punkte, oder aber zweimal N reelle Punkte in einem Iterationsverfahren. Es ist als Drittes aber auch möglich, eine reelle Punkteserie der Länge 2N zu verarbeiten. Als Resultat erscheint dann eine komplexe Folge der Länge N. Den nötigen Zusatzalgorithmus findet man bei COOLEY[35]. Eine Maschine, die für N = 256 Punkte gebaut ist, könnte 512 Punkte verarbeiten. Diese Wahlmöglichkeiten sind allenfalls recht interessant und vorteilhaft.

2.3. Vergleich zur DFT - Alternative

Um den Kreis zu schliessen, der von diesen Blättern umspannt wird, möchte ich abschliessend den DFT- und den FFT-Prozessor einander gegenüberstellen.

Ein Blick auf die Blockschemata zeigt, dass für die beiden Maschinen etwa derselbe schaltungstechnische Aufwand getrieben werden muss; dies ist wohl das überraschendste Ergebnis. Auch der DFT-Prozessor könnte eingangsseitig komplex organisiert werden, worauf wir angesichts des erweiterten Anwendungsbereichs wohl kaum verzichten würden.

Der DFT-Prozessor leistet eine Konversion von 256 Punkten in 32 msec. Der FFT-Prozessor brauch dafür 1,5 msec, ist also gut zwanzigmal schneller. Entsprechend erweitert sich mit FFT die EEG-Kanalzahl um einen Faktor zwanzig.

---

[35] l.c., p.30 ff.

Mit FFT würde der Dimension elektrotopologischer Studien prozessorseitig keine Grenzen gesetzt. Oder die zeitliche Ueberlappung der Datenpakete könnte sehr weit getrieben werden, was möglicherweise interessant wäre. Oder drittens, der Prozessor stünde während der on-line Analyse für statistische Weiterverarbeitung höherer Ordnung zur Verfügung. Varianzspektra, Bispektra, Bicohärenz in Echtzeit[36], Kreuzspektren und Kohärenz zwischen sämtlichen Paaren einer quadratischen Elektrodenmatrix der Dimension 7 in Echtzeit , das sind Streiflichter auf ferne Horizonte, denen der FFT-Prozessor als Herz der Anlage gewachsen wäre. Mit DFT können solche Ziele nicht angestrebt werden. Man muss sich abermals vor Augen führen, dass die Neurophysiologie mit voller Kraft auf jene Horizonte u.a. zusteuert und eine leistungsfähige Elektronik braucht. Knapp zu kalkulieren, scheint mir hier ungünstig. Und diese Auffassung verstärkt sich, wenn noch andere Anwendungsgebiete erwogen werden. Das EEG ist noch nicht das schnellste Biosignal. Das Myogramm ist 100mal schneller und liegt seinerseits im unteren Audiobereich. On-line Klang-, Geräusch- und Sprachanalysen eröffnen ein ganz grosses Anwendungsgebiet für schnelle Fourierprozessoren, das seine Aktualität noch längst nicht eingebüsst hat, obwohl Orgelklänge schon in den Dreissigerjahren spektrometriert wurden. Pattern recognition, von Modellen des visuellen Systems über optische Leser bis zu morphologischen Vergleichen in Archäologie und Paläographie, begründet einen anderen Forschungskreis, in dessen Zentrum die zweidimensionale Fouriertransformation steht mit besonders hohem Datenanfall.

Die Reihe liesse sich verlängern. Es ist nicht meine Absicht, dies hier zu tun. Auch muss einer Entscheidung zwischen den Alternativen die Diskussion unter den Beteiligten vorangehen. Ich hoffe, dass sie auf der Grundlage meiner beiden Berichte fruchtbar geführt werden kann.

---

[36] Vgl. Dumermuth, G., Huber, P.J., Kleiner, B. and Gasser, Th. Analyses of the interrelations between frequency bands of the EEG by means of the bispectrum. Electroenceph. clin. Neurophysiol., 1971, 31: 137 - 148.

3. Literaturverzeichnis

Bendat, J.S. and Piersol, A.G. Measurement and analysis of random data. John Wiley, Inc., New York, 1966, 390 p.

Bergland, G.D. A fast Fourier transform algorithm for real-valued series. Commun. ACM, 1968, 11: 703 - 710.

Bergland, G.D. A guided tour of the fast Fourier transform. IEEE Spectrum, 1969, 6: 41 - 52.

Bergland, G.D. Fast Fourier transform hardware implementations. I. An overview. II. A survey. IEEE Trans. Audio Electroacoust., 1969, AU-17: 104 - 119.

Bertram, S. On the derivation of the fast Fourier transform. IEEE Trans. Audio Electroacoust., 1970, AU-18: 55 - 58.

Cooley, J.W., Lewis, P.A.M. and Welch, P.D. The fast Fourier transform algorithm and its applications. IBM Res. Paper RC-1743, Febr. 1967, 157 p.

Corinthios, M.J. A time-series analyzer. Proc. Symp. Comput. Processing in Commun., 1969, 19: 47 - 61. MRI Symposia Ser. New York: Polytechnic Press.

Dumermuth, G., Huber, P.J., Kleiner, B. and Gasser, Th. Analyses of the interrelations between frequency bands of the EEG by means of the bispectrum. Electroenceph. clin. Neurophysiol., 1971, 31: 137 - 148.

G-AE Subcommittee on Measurement Concepts. What is the fast Fourier transform? IEEE Trans. Audio Electroacoust., 1967, AU-15: 45 - 55; also Proc. IEEE, 1967, 55: 1664 - 1674.

Gold, B. and Rader, C.M. Digital processing of signals, McGraw-Hill, Inc., New York, 1969, Chapter 6.

Hanley, J. Practice and problems of biological signal processing. Proc. 2nd Symp. on nonlinear Estimation Theory San Diego, 1971, 2: 403 - 413.

Klahn, R., Shively, R.R., Gomez, E. and Gilmartin, M.J. The time-saver: FFT hardware. Electronics, June 1968, p. 92 - 97.

Pease, M.C. Methods of Matrix Algebra. Academic Press, New York, 1965, Ch. XIV.

Pease, M.C. An adaptation of the fast Fourier transform for parallel processing, J. Ass. Comput. Mach., 1968, 15: 252 - 264.

Pease, M.C. Organization of large scale Fourier processors. J. Ass. Comput. Mach., 1969, 16: 474 - 482.

Shively, R.R. A digital processor to generate spectra in real time. IEEE Trans. Comput., 1968, C-17: 485 - 491.

ANHANG B: EIN DFT-PROZESSOR

Ein DFT-Prozessor
zur mehrkanaligen EEG-Spektralanalyse
in Echtzeit
ohne Benützung des FFT-Algorithmus.

Inhaltsübersicht

0. Einleitung

Die Wiederentdeckung der schnellen Fouriertransformation (FFT) durch Cooley und Tuckey im Jahre 1965[1] hat eine Flut von Publikationen ausgelöst, die sich mit den Modifikationen und Anwendungen dieses Algorithmus befassen. Im Februar 1969 wurden bereits 100 Arbeiten gezählt[2]. Diese Zahl wird sich inzwischen vervielfacht haben. FFT ist zu einem Schlagwort mit geradezu numinosem Akzent geworden. In dieser Tatsache spiegelt sich ein Zeitgeist, der irgendwo "the Age of Electronic Analysis and Synthesis" genannt wurde.

Parallel dazu führte der Hardware-seitige Fortschritt zu quantitativen Verbesserungen um Grössenordnungen. Die MSI- und LSI-Technologie ermöglichte die RAM- und ROM-Speicher in der heute vorliegenden multizellulären Form. Noch vor wenigen Jahren undenkbare MOS-Schieberegister mit über tausend Bits und Clockfrequenzen von mehreren Megahertz werden zu spektakulären Preisen angeboten. In den Lagern der Halbleiterproduzenten liegt ein Schatz potentieller Anwendungmöglichkeiten verborgen, an dem bislang nur geritzt wurde.

Auf die Fragestellungen, welche die Neurophysiologie oder im allgemeinen die Physiologie in diesem Zusammenhang aufgeben, wird seitens der Technik noch immer zuwenig geachtet, oder die Konvergenz wird von allerlei Sprachschranken gestoppt. Die Trächtigkeit dieses Bodens kann aber nicht genügend hoch veranschlagt werden, wie überhaupt die Bedeutung der Wissenschaften im Umkreis der Hirnforschung. Man muss sich

---

[1] Cooley, J.W. and Tuckey, J.W. An algorithm for the machine calculation of complex Fourier series. Math. Computation, 1965, 19: 297 - 301.

[2] Singleton, R.C. A short bibliography on the fast Fourier transform. IEEE Trans. Audio and Electroacoustics, 1969, 17: 166 - 169.

vergegenwärtigen, dass man in jüngster Zeit beginnt, sich dem uralten Rätsel der Hirnvorgänge mit empirisch-quantitativen Methoden zu nähern. Schon vor 100 Jahren versuchten Helmholtz und seine Schüler von diesem Ansatz auszugehen, allerdings war die Zeit noch nicht reif und der Erfolg entsprechend gering. Inzwischen haben Naturwissenschaft und Technik eine völlig neue Situation geschaffen. So wie die Physik um die Jahrhundertwende dem Problem der Materie, so steht heute die Phalanx sämtlicher Disziplinen der Sphinx des Geistes gegenüber. Auch Techniker sind aufgerufen mitzutun und ihre Methoden bereitzuhalten. -

Ein festverdrahteter Prozessor zur diskreten Fouriertransformation (DFT) bildet das Kernstück eines Systems für simultane Vielkanal-EEG-Spektralanalyse in Echtzeit[3]. Da in der Literatur der letzten Jahre die gewöhnliche diskrete Fouriertransformation fast nie zur Sprache kommt, konzentrierte ich mich zunächst auf die Möglichkeit, den Prozessor im Sinne von FFT zu organisieren. Gegen Ende der letzten Semesterferien habe ich hierzu zwei Pläne aufgestellt. Die Wiederaufnahme des Fadens in den laufenden Semesterferien führte auf eine Publikation, die ohne Benützung des FFT-Algorithmus einen Echtzeit-Analysator für Sprachsignale beschreibt[4]. Der vorliegende Bericht gilt der Frage, ob die gewöhnliche DFT-Struktur unter Verwendung von MOS-Schieberegistern und MOS-Festwertspeichern als Zentraleinheit des EEG-Analysators dienen könnte. Die nächste Arbeit wird einen Vergleich zu der entsprechenden FFT-Variante beinhalten. Späteres soll sich auf die Ein- und Ausgabeprobleme, die A/D-Wandlung, den Multiplexbetrieb und die Organisation des EEG-Analysators

---

[3] Die Anregung zu diesem Projekt stammt von Dr. G. Dumermuth, Privatdozent für Kinderheilkunde und Elektroencephalographie an der Universitäts-Kinderklinik Zürich.

[4] Bially, T. Audio frequency spectrum analysis using MOS memory elements. IEEE Trans. Audio and Electroacoustics, 1970, 18: 201 - 203.
Herr M. Maurer, Assistent am Institut für Technische Physik ETH, hat mich auf diese Arbeit und Möglichkeit hingewiesen.

als Ganzes[5] beziehen.

1. Das EEG - Signal

Es kann sich nicht darum handeln, in diesem Abschnitt auf die statistischen Eigenschaften oder gar auf die biologische Bedeutung des Elektroencephalogramms einzugehen. Ich möchte lediglich zusammenfassen, welche Kenngrössen sich auf die Organisation des Prozessors auswirken und in welcher Art sie das tun. Die Angaben stützen sich auf ein mit Dr. Dumermuth ausgearbeitetes Pflichtenblatt, verschiedene schriftliche Quellen und eigene Ueberlegungen und sind als vorläufige Richtlinien gedacht.

Die in der EEG-Kurve aufgezeichnete Potentialschwankung an der Schädeloberfläche kann als Zufallsvariable aufgefasst werden. Bei nicht zu langen Datenstücken[6] und bestimmten physiologischen Bedingungen ist das Signal stationär. Es besteht aus zahlreichen unkorrelierten Frequenzkomponenten, deren Teilvarianzen sich zur Gesamtintensität linear superponieren. Die Darstellung dieser Teilvarianzen als Funktion der Frequenz ergibt das Varianzspektrum. Bisher war das Interesse der Fachwelt fast ausschliesslich auf die Frequenzgebiete zwischen 0 und 30 Hz gerichtet. Die berühmten α-, β-, ϑ- und δ-Rhythmen finden sich alle in diesem Intervall. Auch phasische Abläufe sind kaum schneller. Die Untersuchung höherer Frequenzgebiete ist zudem erschwert durch die Aktivität der Kopfmuskulatur. Das Myogramm hat ein besonders breites Spektralfenster, das sich von ca. 20 bis über 1000 Hz erstreckt.

---

[5] Inklusive Fragen wie Anti-aliasing-Filter, "leakage" und "picked-fence"-Effekt und deren Bewältigung durch geeignete Fenster ("windowing").

[6] einige Sekunden

[7] Im Hinblick auf FFT wählt man diese Zahlen am besten aus der Reihe 2^m.

Berufen wir uns auf das klassische Frequenzfenster zwischen 0 und 30 Hz und rechnen mit einem Anti-aliasing-Filter, das oberhalb etwa 40 Hz scharf abschneidet, dann scheint eine Samplingfrequenz f_s = 128 Hz vernünftig zu sein[7]. Die Nyquistfrequenz liegt also bei f_N = 64 Hz. Eine abgetastete frequenzbandbegrenzte Zeitfunktion f(nT) wird durch DFT in eine komplexe Frequenzfunktion $F(k\Omega)$ umgewandelt[8]. Ist f(nT) reell und schreibt man $Re\{F(k\Omega)\} = R(k\Omega)$, $-Im\{F(k\Omega)\} = I(k\Omega)$, so ist $R(k\Omega)$ eine gerade, $I(k\Omega)$ eine ungerade Funktion mit Bezug auf die Nyquistfrequenz. Um das Spektrum zu erhalten, benützen wir die Spektrallinien unterhalb f_N. Die Spektrallinien oberhalb von f_N bringen keine neuen Informationen und können weggelassen werden. In dieser Weise werden N Tastwerte f(nT) in N/2 Cosinus-Koeffizienten $R(k\Omega)$ und N/2 Sinus-Koeffizienten $I(k\Omega)$ transformiert.

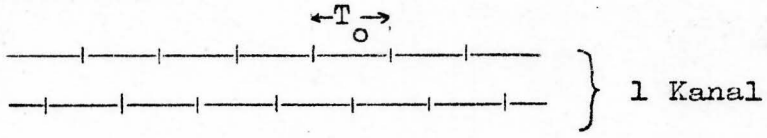
Aus Gründen der Stationarität ist es vernünftig, eine Tastperiode $T_o = (N-1)T$ = 2 sec anzusetzen. Die Frequenzauflösung ist $1/T_o$ = 0,5 Hz. Ein solches Paket umfasst also N = 256 Punkte. Diese Zahl bestimmt die Länge der Schieberegister und der Speicher. Es können so beliebig viele Tastperioden zwecks Mittelung akkumuliert werden. Auch an eine Schachtelung der Tastfenster lässt sich denken. Die Frequenzauflösung bleibt bei all diesen Manipulationen erhalten.

Wie ich unten zeigen werde, verarbeitet der vorliegende DFT-Prozessor 256 Punkte in 32 msec. Pro Sekunde braucht er also 16 msec für einen Kanal. Also können 62 Kanäle simultan verarbeitet werden! Auch bei einer Schachtelung in

---

[8] Näheres im Buch:
Gold, B. and Rader, C.M. Digital Processing of Signals. McGraw-Hill, Inc., New York, 1969, 269p.

folgender Form



verbleibt eine Verarbeitungskapazität von 31 Kanälen. Das gibt immer noch eine anständige Matrix für topologische Korrelationsanalysen.

Jeder EEG-Fachmann weiss, dass es mit dem Auflösungvermögen seines Polygraphen nicht sehr gut steht. Er braucht schon eine besonders geschickte Assistentin, um in die Gegend des Tintenstrichs hinunterzukommen, dessen Dicke bei ca. 0,4 % des gesamten Exkursionsbereiches liegt. Dieselbe Auflösung erreicht man mit einer 8-bit-Dualzahl. Zusammen mit dem Vorzeichenbit erweist sich eine 8-bit-Dualzahl[9] als passende Darstellung von EEG-Tastwerten. Genauer zu sein ergibt keinen Sinn und verteuert den Prozessor sofort. Die anschliessende Beschreibung stützt sich auf diese Zahldarstellung.

2. Der Prozessor

Die diskrete Fouriertransformation von N Stützstellen ist mit $\Omega = \frac{2\pi}{NT}$ gegeben durch

$$F(k\Omega) = \sum_{n=0}^{N-1} f(nT)\, e^{-j\Omega Tnk}$$

$$= \sum_{n=0}^{N-1} f(nT) \cos\frac{2\pi nk}{N} - i \sum_{n=0}^{N-1} f(nT) \sin\frac{2\pi nk}{N}$$

$$= R(k\Omega) - I(k\Omega).$$

Bially[10] gibt in seiner Arbeit drei verschiedene Strukturen

---

[9] Definition: 8 bit = 1 byte

[10] l.c., p.202 und 203. Diese Strukturen sind bei ihm dargestellt in den Figuren 1, 2 und 5.

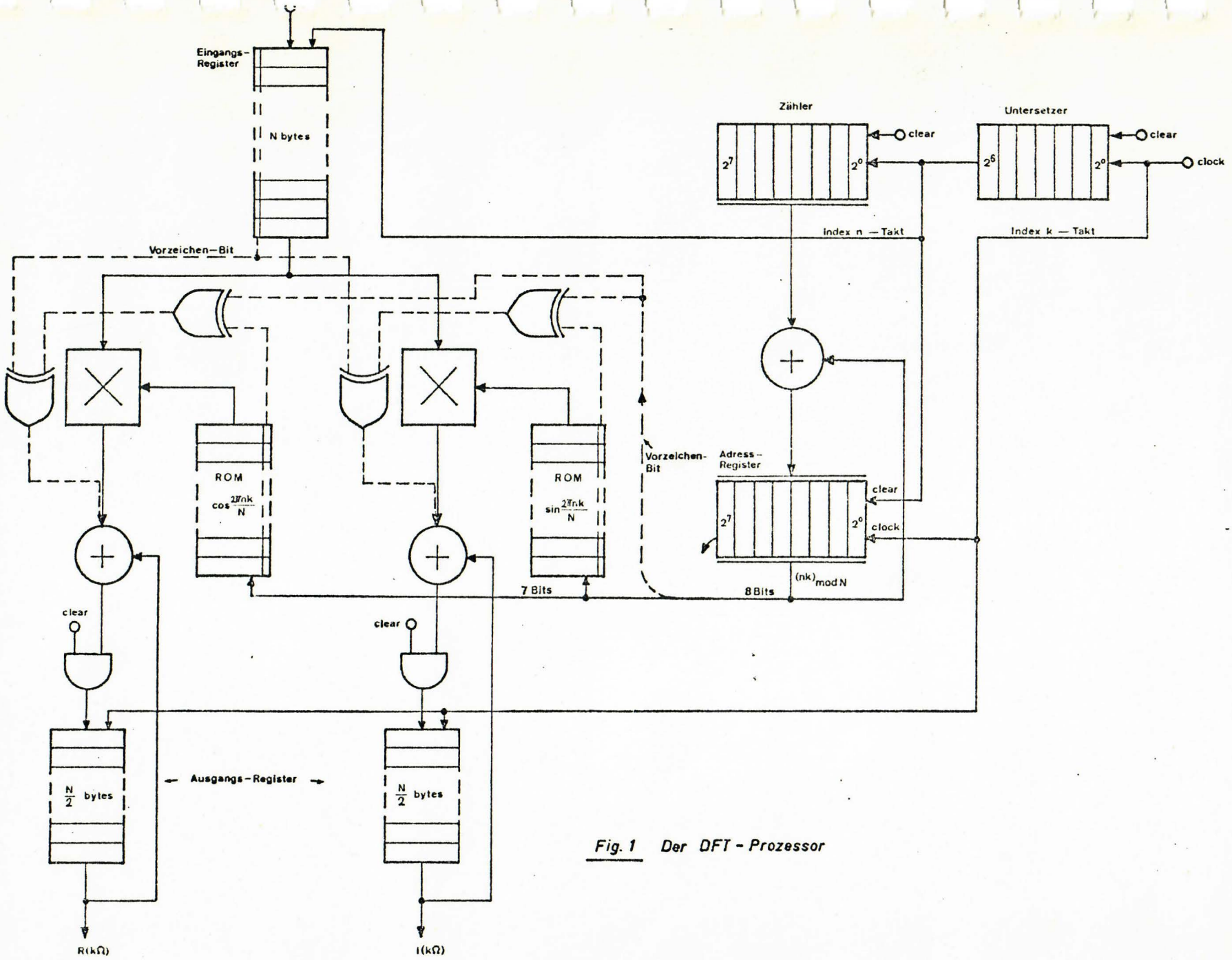


Fig. 1 Der DFT - Prozessor

an, welche diese Transformation durchführen können. Sie werden zunehmend schneller aber auch zunehmend komplizierter. Mir scheint die Anordnung, welche bei Bially in Fig. 2 angedeutet ist, für unsere Zwecke die richtige zu sein. Sie besticht durch eine schöne Symmetrie und einen einfachen Signalfluss, und sie hat den Vorteil, genau zu sein. In der schnellsten Version wird nämlich eine ROM-Organisation verwendet, welche aus Symmetriegründen nicht die exakten Sinus- und Cosinuswerte enthält. Der Fehler, der dadurch entstehen würde, könnte sich bei N = 256 unangenehm auswirken. Zudem benötigt diese Variante ein sehr kompliziertes, undurchsichtiges Selektionsnetzwerk, wodurch der Vorteil grösserer Schnelligkeit teilweise zunichte gemacht würde.

Ich habe die für unsere Zwecke passende Struktur in Fig. 1 aufgezeichnet. Sie wird von zwei Taktfolgen durchpulst, welche dem Fortschreiten der Indices k und n entsprechen. Der Index k wird durch den 1 MHz-Clock repräsentiert. Im ersten Durchlauf des Untersetzers, also nach 128 k-Takten, ist das linke Ausgangsregister R(kΩ) gefüllt mit den Werten

$$f(0.T).\cos(\tfrac{2\pi}{N}\cdot 0\cdot 127)$$
$$\vdots$$
$$f(0.T).\cos(\tfrac{2\pi}{N}\cdot 0\cdot 2)$$
$$f(0.T).\cos(\tfrac{2\pi}{N}\cdot 0\cdot 1)$$
$$f(0.T).\cos(\tfrac{2\pi}{N}\cdot 0\cdot 0),$$

also mit lauter Werten f(0).

Beim 128sten Puls springt der Zähler von 0000000 auf 0000001, und das Adress-Register beginnt in Einerschritten zu wandern. Am Eingangsregister steht jetzt der nächste Stützwert f(1) an. Nach einer weiteren Untersetzerperiode enthält das linke Ausgangsregister folgendes:

$$f(0) + f(1\cdot T)\cdot\cos\left(\frac{2\pi}{N}\cdot 1\cdot 127\right)$$
$$\vdots$$
$$f(0) + f(1\cdot T)\cdot\cos\left(\frac{2\pi}{N}\cdot 1\cdot 2\right)$$
$$f(0) + f(1\cdot T)\cdot\cos\left(\frac{2\pi}{N}\cdot 1\cdot 1\right)$$
$$f(0) + f(1\cdot T)\cdot\cos\left(\frac{2\pi}{N}\cdot 1\cdot 0\right)$$

Wie das weitergeht liegt auf der Hand. Der Stand des Zählers bestimmt die Schrittweite des Adressregisters, das nach jeder Untersetzerperiode gelöscht wird. Am Schluss, das heisst nach 128.256 = 32768 Clock-Impulsen[11] ist die Berechnung der Fourierkoeffizienten fertig. Im linken Ausgangsregister steht

$$\sum_{n=0}^{N-1} f(nT)\cdot\cos\left(\frac{2\pi}{N}n\cdot 127\right) = R((N-1)T)$$
$$\vdots$$
$$\sum_{n=0}^{N-1} f(nT)\cdot\cos\left(\frac{2\pi}{N}n\cdot 2\right) = R(2\cdot T)$$
$$\sum_{n=0}^{N-1} f(nT)\cdot\cos\left(\frac{2\pi}{N}n\cdot 1\right) = R(1\cdot T)$$
$$\sum_{n=0}^{N-1} f(nT)\cdot\cos\left(\frac{2\pi}{N}n\cdot 0\right) = R(0\cdot T).$$

Im rechten Ausgangsregister steht das Entsprechende mit Sinusfunktionen. Das Vorzeichen dieser Serie kann falls nötig während des Auslesens gekehrt werden. Die DFT-Rücktransformation unterscheidet sich an diesem Punkt von der Hintransformation.

---

[11] Bei einer Clock-Frequenz f_c = 1 MHz ergeben sich so die oben erwähnten 32 msec für die Verarbeitung eines Datenpakets.

In den ROM-Speichern stehen nur N/2 Werte. Das entspricht der halben Periode der Winkelfunktionen. Die Adressierung erfordert demnach 7 bits. Das achte bit wird als Vorzeichenbit betrachtet und mit den Vorzeichen der ROM-Koeffizienten und der Tastwerte f(nT) entsprechend verknüpft. Die Multiplikation wird - wie ersichtlich - mit den Absolutwerten durchgeführt. Die bytes stehen überall parallel an. Auch die Multiplizierwerke können rein kombinatorisch ausgelegt werden[12]. Man darf für ein Multiplizierwerk mit ungefähr m . n = 7 . 7 = 49 AND-Gates und ebensovielen Volladdierern rechnen. Dieser Aufwand ist durchaus zu vertreten. Rechenzeit und logischer Ablauf der Multiplikation werden so sehr problemlos. Das Vorzeichen wird erst bei den Addierwerken hinzugenommen.

Die Einbettung dieses Prozessors in die Gesamtstruktur des EEG-Analysators werde ich erst nach dem Vergleich mit der FFT-Alternative beschreiben. Man sieht aber schon jetzt, dass dabei keine unüberwindlichen Schwierigkeiten auftauchen werden. Auch die Kostenfrage werde ich später genau untersuchen. Die Halbleiterpreise für die in Fig. 1 angegebene Schaltung dürften gegenwärtig in der Gegend von Fr. 1000 liegen.

---

[12] Vergleiche Shah, A. Binäre Arithmetik, (Kap. IX aus dem Buch über digitale Schaltungstechnik, zur Zeit im Druck.)

ANHÄNG C: "CARRY-SAVE" MULTIPLIZIERWERKE

Bericht zur Realisierung
eines Serie-Parallel-Multiplizierwerkes
als Vorbereitung der Synthese
eines schnellen Fouriertransformators.

Inhaltsübersicht

1. Einleitung

Digitale Multiplizierwerke können in drei strukturell verschiedene Klassen eingeteilt werden:

- serielle Multiplizierwerke
- Parallel-Serie-Multiplizierwerke
- parallele Multiplizierwerke.

Sie unterscheiden sich in zwei wichtigen Kriterien, der Rechenzeit und der Anzahl benötigter Grundelemente (GE). Diese beiden Kriterien verhalten sich zueinander reziprok.

Wir vergleichen im folgenden drei typische Vertreter obiger Klassen bezüglich Rechenzeit und Materialaufwand. Dabei seien zwei 16-Bit Operanden (m = n = 16) vorausgesetzt.

Das serielle Multiplizierwerk[1] ist im Aufbau einfach, benötigt aber eine grosse Rechenzeit. Die Anzahl Grundelemente ist[2]

n + 3	Flip-Flop (FF)
1	Volladdierer (VA)
1	AND-Gate (AND).

Mit n = 16 also 21 GE.

Die Rechenzeit beträgt

$$t_{tot} = (m + n)(n + 1)(t_s + t_{FF} + t_{pdAND}).$$

Typische Zeiten sind $t_c = t_s = t_{FF} = 40ns$, $t_{pdAND} = 20$ ns.
Damit wird $t_{tot} = 54{,}4$ /us.

---

[1] Vergleiche A. SHAH: Binäre Arithmetik.

[2] Eingangs- und Ausgangsregister nicht gezählt.

Das Parallel-Serie-Multiplizierwerk wird unten genauer besprochen. Zum Vergleich können die Zahlen

61 GE, $t_{tot} = (n + m)(t_{FF} + t_s) = 2{,}5\ /us$

herangezogen werden.

Parallel-Multiplizierwerke erfordern einen sehr grossen Materialaufwand. Dafür bieten sie die Vorteile einer nichtsequentiellen Logik[3]. Es handelt sich um rein kombinatorische Schaltungen, aus VA und Gattern bestehend. Die zweidimensional periodische Struktur begünstigt die Herstellung in LSI-Technik, sodass der Nachteil eines grossen Materialaufwands weniger ins Gewicht fällt. Für (16,16) Bit werden rund 500 GE benötigt. Die Rechenzeit ist etwa

$t_{tot} = (n - 1)t_c + (n - 1)t_s = 1{,}2\ /us.$

Noch schneller rechnen sog. Baumstrukturen, bei allerdings exzessivem Materialaufwand.

Diese Uebersicht zeigt die relativ günstige Platzierung der Serie-Parallel-Multiplizierwerke in dem recht breiten Spektrum möglicher Realisierungen.

---

[3] Keine Uhrimpulse, keine Steuerlogik, keine FF.

2. Varianten von "carry-save" Multiplizierwerken

Es werden vier Varianten diskutiert, denen wir die Nummern 1, 1', 2, 2' gegeben haben.

Der "carry-save" Akkumulator bietet im Vergleich zum gewöhnlichen Akkumulator eine Zeitersparnis um etwa einen Faktor 8, bei nur geringem Mehraufwand. Die vier Varianten sind daher alle "carry-saving".

1) Paralleler Ausgang, einfache VA. Nach Einlesen des Multiplikators werden durch einen Steuerimpuls Z die anstehenden Ueberträge im "ripple-carry" Modus addiert. Die Schaltung geht aus Fig. 1 hervor.

Material[4]:

Tore:	$n + (2n - 2) = 44$	} 105 GE
FF:	$(n - 1).2 + m = 46$	
VA:	$n - 1 = 15$	

Rechenzeit[5]:

$t_{FF} = t_s = t_c = 20$ ns, $t_{pdGate} = 10$ ns.

$$t_{tot} = m(t_{FF} + t_s + 2t_{pd}) + (n - 2)(t_c + 2t_{pd}) + t_s = 1{,}54 \text{ /us}$$

1') Einfache VA, serieller Ausgang, d.h. die Verrechnung der Schlussüberträge erfolgt sequentiell, es werden also m + n Uhrimpulse benötigt. Der serielle Ausgang wird auf ein 32-Bit Register gegeben. Besoders schön an dieser Variante ist die Homogenität des Operationsablaufs.

---

[4] Im folgenden ohne Eingangsregister und Steuerlogik, hingegen mit Ausgangsregister.

[5] Maximalzeit unter Verwendung der schnellen TTL-Serie 74HN von Texas Instruments.

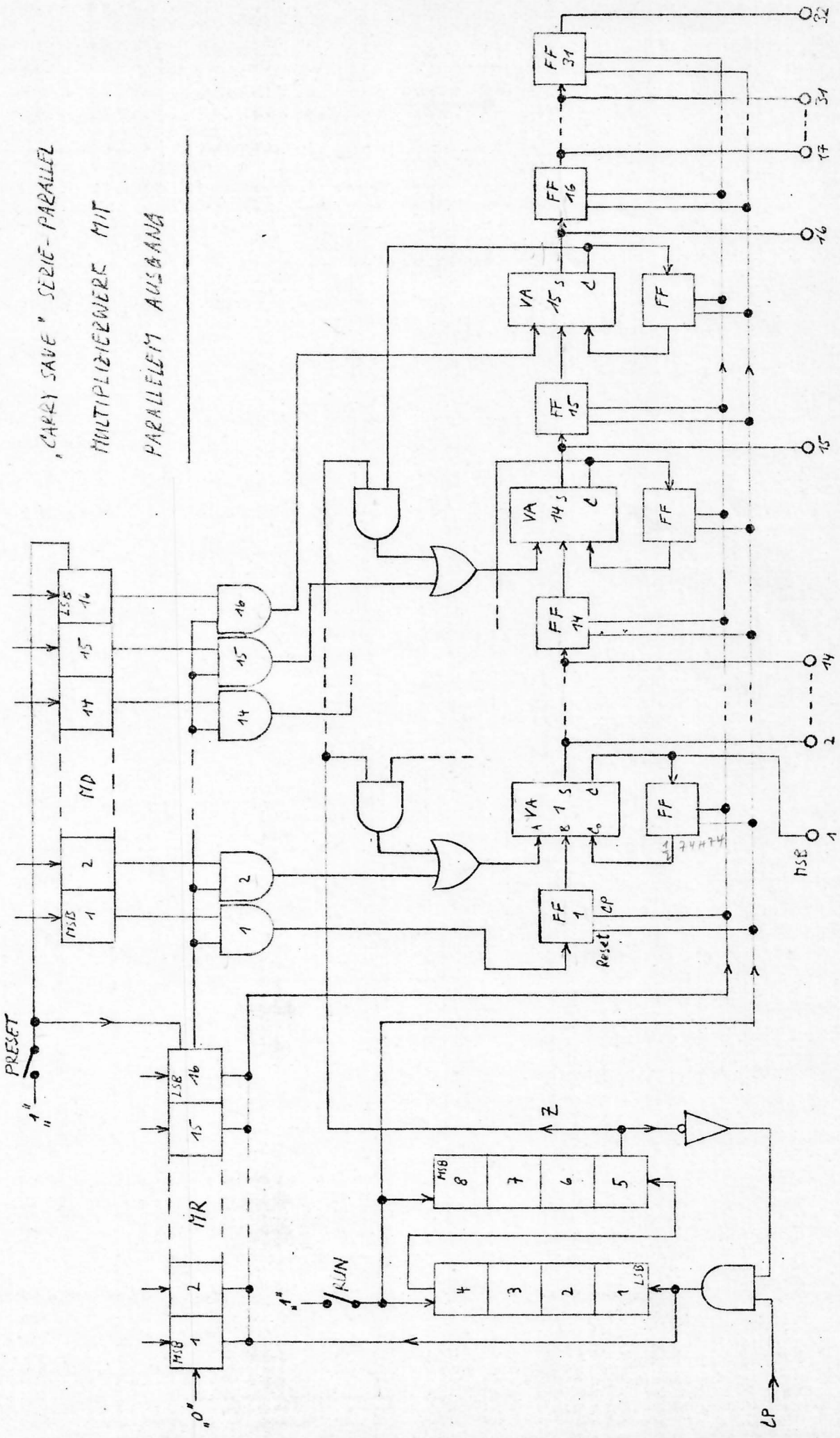

„CARRY SAVE" SERIE-PARALLEL
MULTIPLIZIERWERK MIT
PARALLELEM AUSGANG
MD
MR
VA
FF
PRESET
RUN
LP
Reset
MSB
LSB
10.8.72 EF

Fig. 1

Material:

Tore: $n = 16$
FF: $(n-1).2 + m + n = 62$
VA: $n - 1 = 15$

} 93 GE

Zeit:

$$t_{tot} = (n + m)(t_{FF} + t_s + t_{pd}) = 1{,}60 \text{ /us}$$

2) Wie in Fig. 2 gezeigt, ist das Merkmal dieser und der folgenden Variante die Verwendung von (5,3)-Parallelzählern, beispielsweise SN 7483 N von Texas Instruments. Auffallend ist die Kette zweier FF in der "carry-save" Schleife des Uebertrags. Dieser Uebertrag besitzt jetzt das Gewicht 100_2, muss daher um zwei Clockperioden verzögert berücksichtigt werden, wogegen der Ausgang Σ_2 das Gewicht 10_2 besitzt, und wie der Uebetrag unter 1) behandelt wird.

Material:

Tore: $n = 16$
FF: $6.n/2 - 2 + 32 = 78$
(5,3)-VA: $n/2 = 8$

} 102 GE

Zeit:

$t_{FF} = 20$ ns, $t_{pd} = 10$ ns, $t_s = t_c = 40$ ns

$$t_{tot} = (m + n)(t_s + t_{FF} + t_{pd}) = 2{,}24 \text{ /us}$$

2') Diese letzte Variante unterscheidet sich von 2) durch einen parallelen Ausgang. Die Berücksichtigung der Ueberträge nach Verarbeitung des Multiplikators kann hier allerdings nicht mehr rein kombinatorisch erfolgen, da die Ausgänge P (siehe Fig. 2) nicht direkt vom VA gespiesen werden.

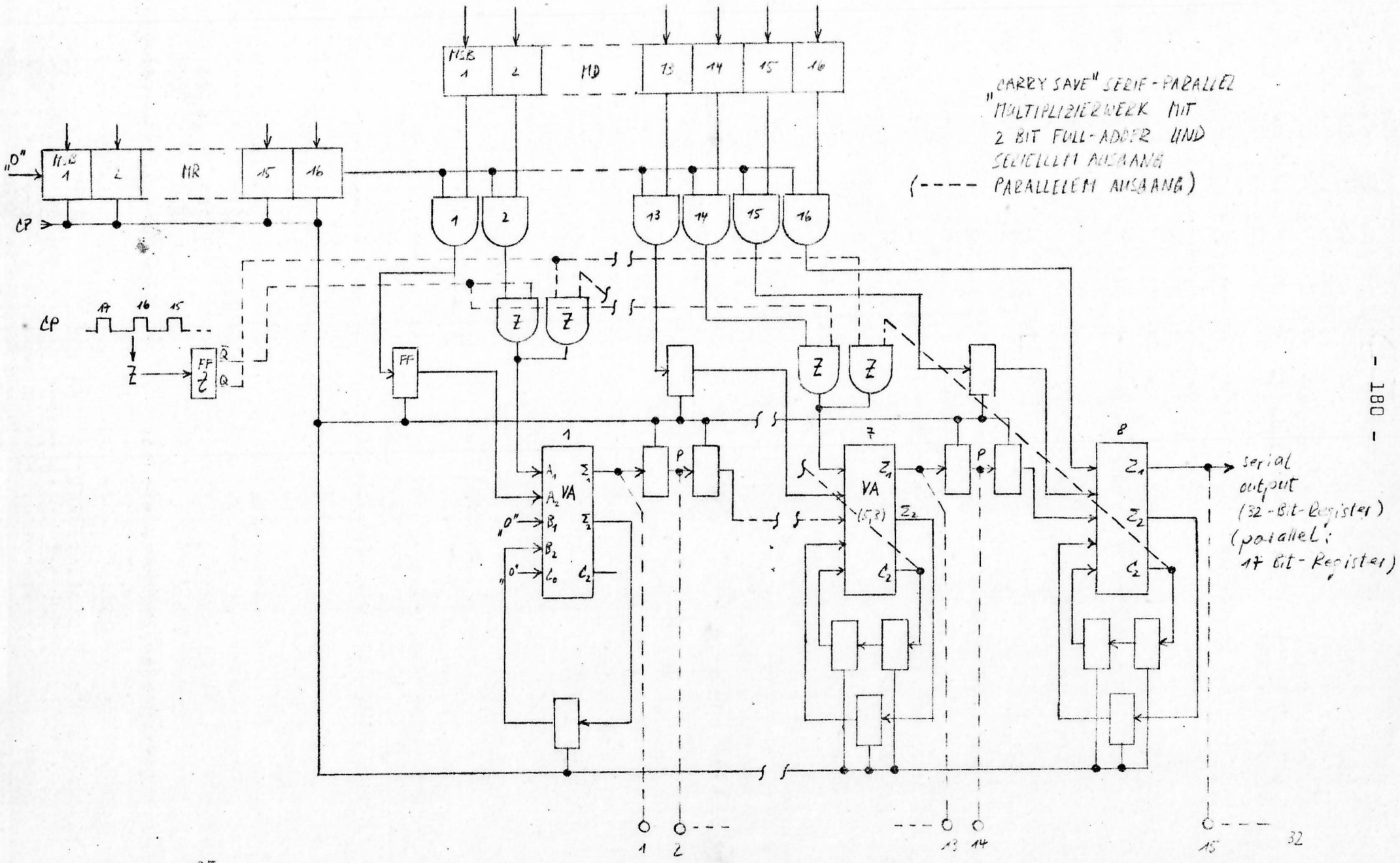


Fig. 2

- 180 -

11.8.72 BF

Die Information an den Punkten P muss mit den nachfolgenden Ueberträgen[6] der Ausgänge Σ_2 kombiniert werden. Die Ueberträge[7] C_2 müssen hingegen zur Information an den Ausgängen Σ_1 der vorhergehenden VA addiert werden. Beides erzielt man dadurch, dass man am Schluss de m-ten Uhrimpulses durch einen Steuerimpuls die Gatter Z auf die Ueberträge C_2 umschaltet und in dieser Position in passendem Zeitabstand einen weiteren Uhrimpuls hinzufügt, sodass P und Σ_2 am je nachfolgenden VA anstehen. Das Ausgangsregister besitzt deshalb 17 Bits.

Material:

Tore: $(n/2 - 1).2 + n = 30$
FF: $(n/2 - 1).6 + 3 + (m + 1) = 62$
(5,3)-VA: $n/2 = 8$

} 100 GE

Zeit:

$$t_{tot} = m(t_s + t_{FF} + 2t_{pd}) + n/2 \cdot (t_{pd} + t_c) + (t_{FF} + t_s)$$

$$= 1{,}74 \text{ } /\text{us}$$

Aus diesen Kalkulationen geht die Variante 1') als beste hervor. Sie benötigt am wenigsten GE, besticht durch ihre klare Funktionsweise und liegt mit der Rechenzeit von 1,60 /us nur unbedeutend über derjenigen von 1). Ein weiterer Vorteil liegt darin, dass Rückstellimpulse für die FF des Akkumulators wegfallen, da dieser beim Auslesen des Resultats mit Nullen gespült wird. All dies bewog uns zur Wahl von 1') als Grundlage des Prototyps, der im folgenden beschrieben ist.

---

[6] Diese Uebeträge haben das Gewicht 10_2.

[7] Mit Gewicht 100_2.

3. Konstruktion eines (16,16)-Bit Parallel-Serie-Multiplizierwerks

Fig. 3 zeigt ein Detailschema, an dem der Funktionsablauf besser studiert werden kann, als am Gesamtschema Fig. 4. Oben sind die beiden Eingangsregister zu erkennen. Die Eingangsdaten, die auf einer Schalterbank bestimmt werden, gelangen in die Register, sobald die Taste PRESET betätigt wird. Die Taste RUN stellt den Binärzähler (aus zwei SN 7493) in Ausgangsposition. Dadurch wird das AND-Gatter geöffnet, über welches die Uhrimpulse eintreten. Diese setzen den Rechner in Betrieb, wie oben gezeigt. Gleichzeitig werden sie vom Binärzähler gezählt. Wenn 32 Impulse eingetreten sind, wird das Gatter geschlossen. Der Rechenprozess ist dann beendet, das Resultat steht im Ausgangsregister (vier SN 74164) an.

Im Gesamtschema Fig. 5 sind noch einige Punkte bemerkenswert. Multiplikator und Ausgang werden von kleineren Registertypen gebildet[8]. Unten rechts ist zu sehen, dass der Uhrimpuls nach dem Gatter (SN 7408) über den Umweg zweier Negatoren in die Schaltung läuft. Dies deshalb, weil das Ausgangsregister auf negative Flanken anspricht, die gleichzeitig oder eher etwas früher anstehen müssen, als die übrigen positiven Flanken der Uhrimpulse. Bei der Numerierung der ICs und der Verbindungspunkte ist die erste Ziffer immer die Nummer der Steckeinheit. Bei den ICs folgen Buchstaben nach, die den Platz auf der Steckeinheit bezeichnen. Bei den Verbindungspunkten folgen Buchstaben oder ein- bis zweistellige Zahlen. Sie sind die Kontaktbezeichnungen der jeweiligen Steckeinheit. Damit schon die erste Rechnung nach Einschalten des Werkes ein richtiges Resultat liefert, wurde die Clear-Leitung der FF dennoch angeschlossen. Die FF werden so beim ersten Einlesen auf Null

---

[8] Aus Gründen der Lieferfrist, Lagerhaltung, etc.

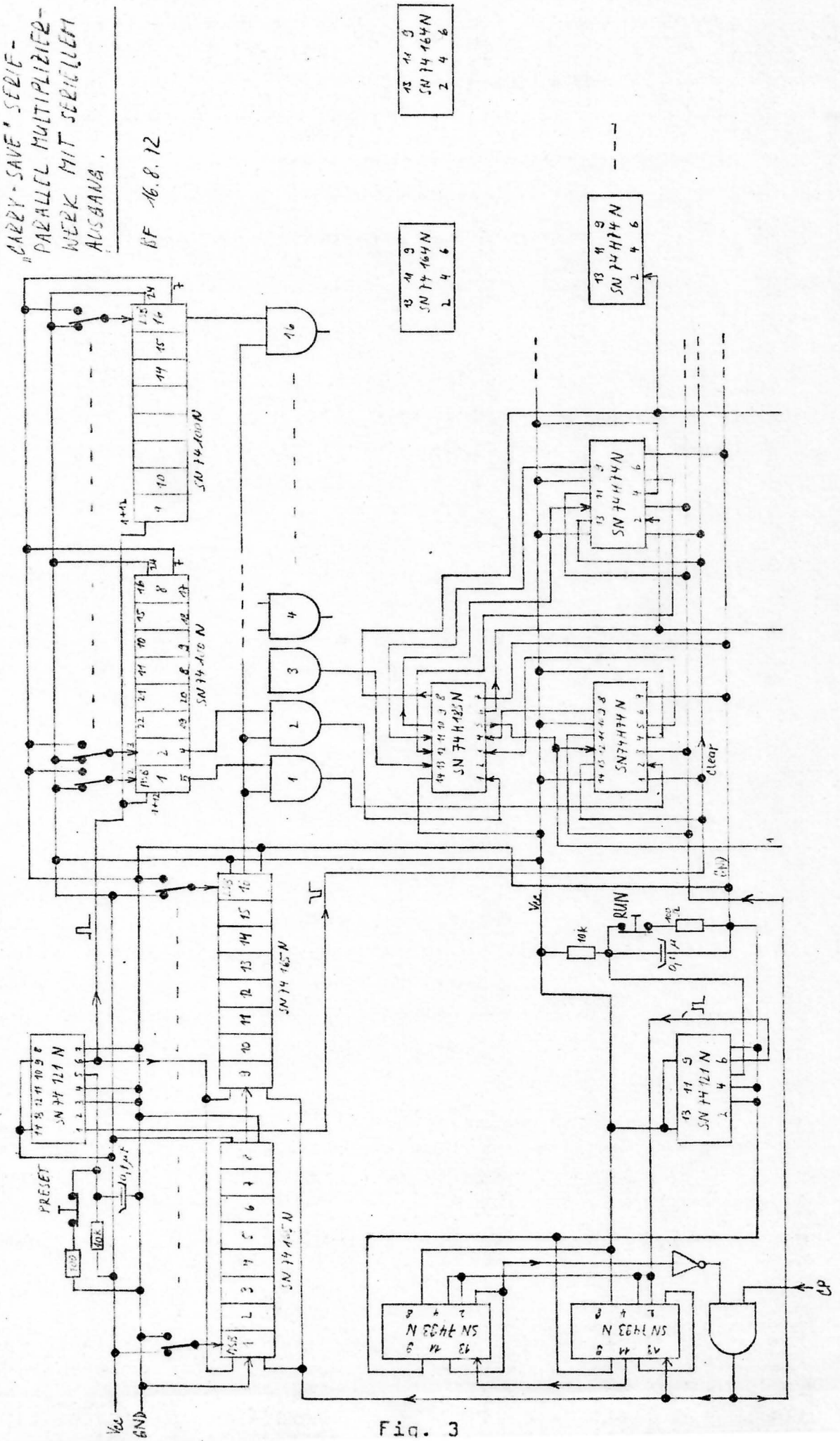
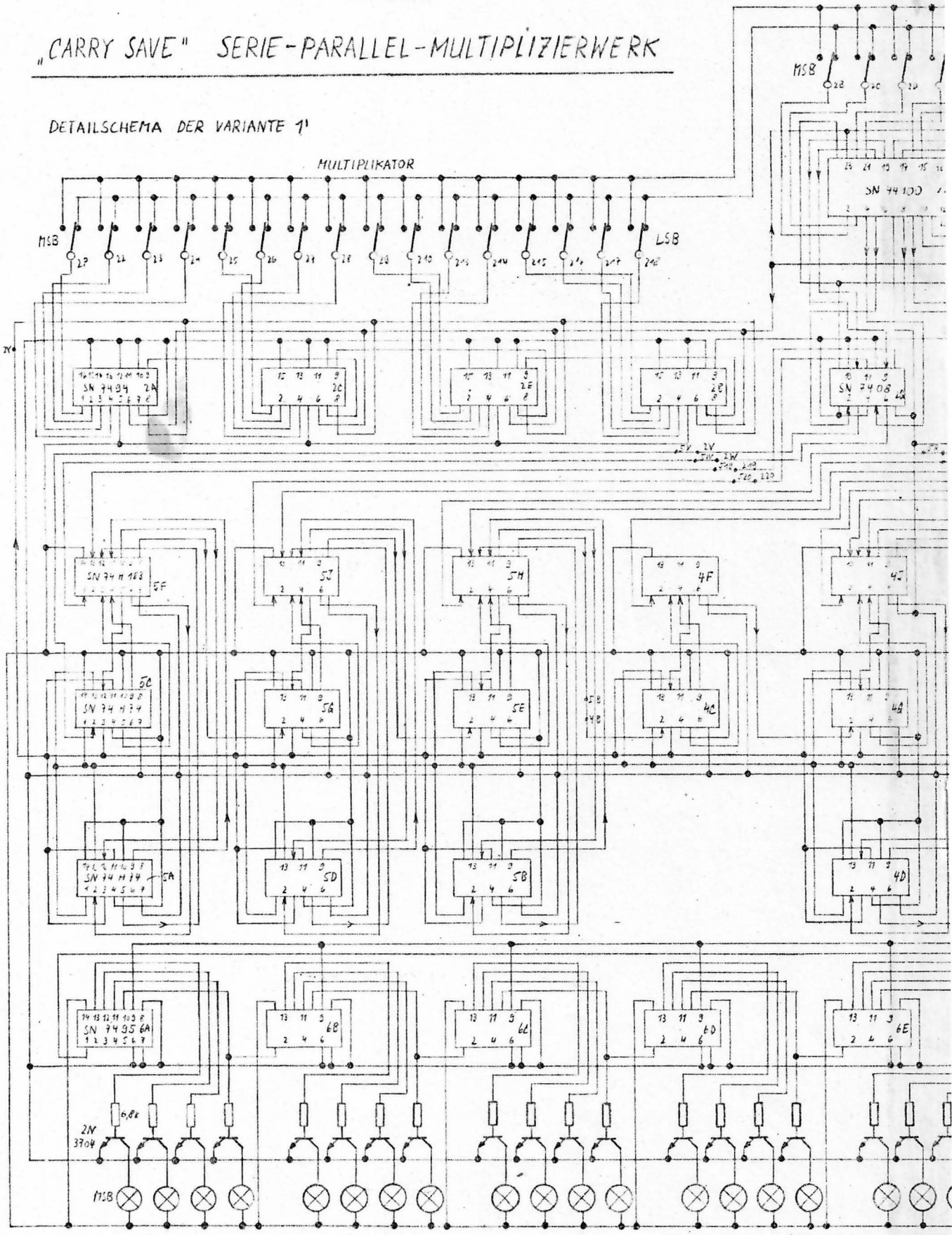
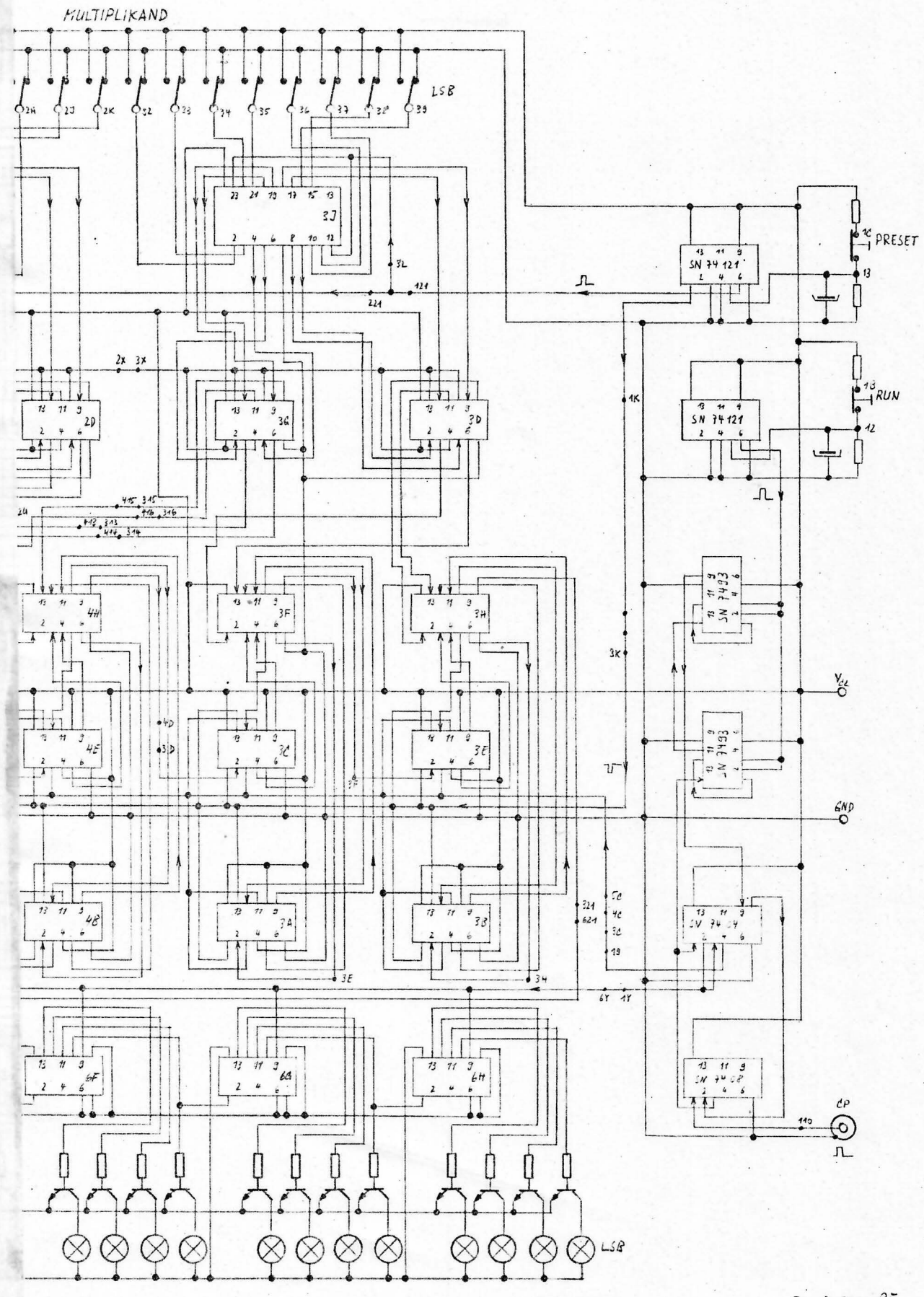

DETAILSCHEMA
„CARRY-SAVE" SERIE-
PARALLEL MULTIPLIZIER-
WERK MIT SERIELLEM
AUSGANG.
BF 16.8.12
SN 74 164 N
SN 74 H 74 N
SN 74 H 183 N
SN 74 H 74 N
SN 74 H 74 N
SN 74 H 183 N
SN 74 H 183 N
SN 74 121 N
SN 74 121 N
SN 74 165 N
SN 74 165 N
SN 74 H 00 N
PRESET
RUN
clear
CP
Vcc
GND
10k

Fig. 3

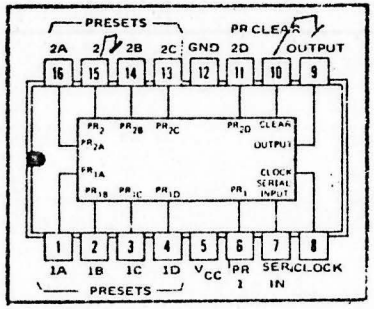
„CARRY SAVE" SERIE-PARALLEL-MULTIPLIZIERWERK

DETAILSCHEMA DER VARIANTE 1'

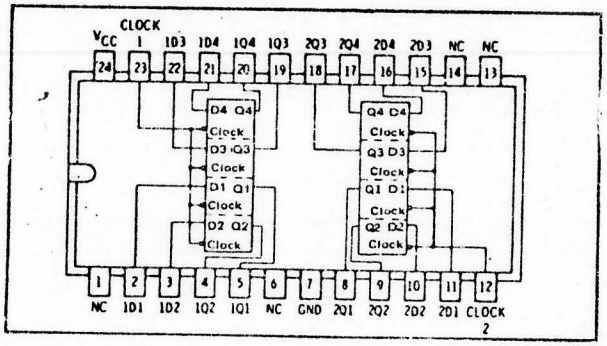




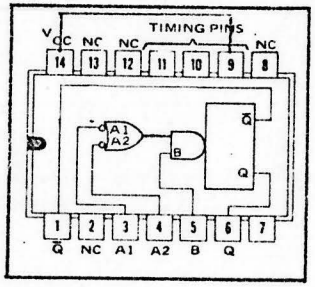
30.8.72 PF



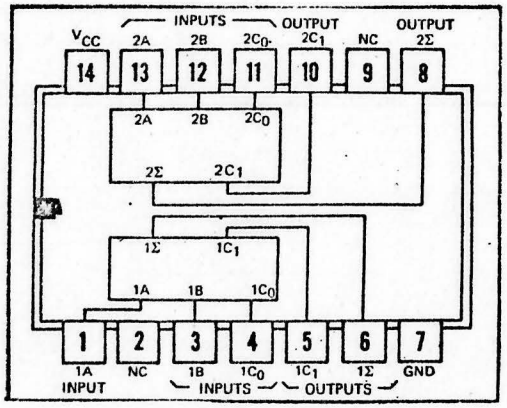
SN 7494



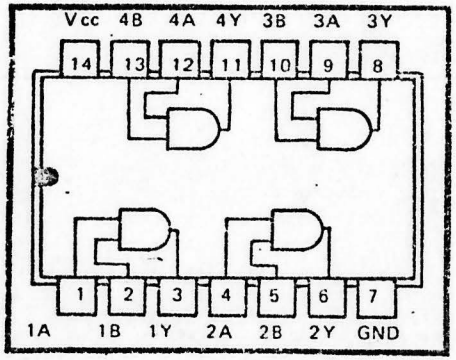
SN 74100



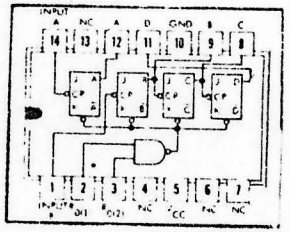
SN 74121



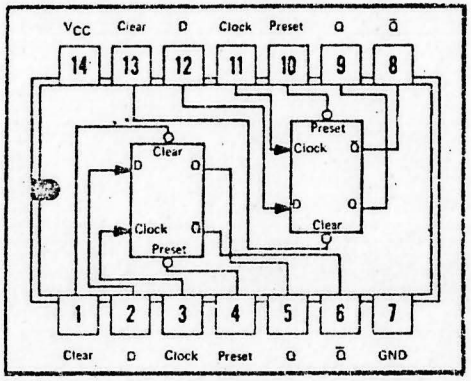
SN 74 H 183



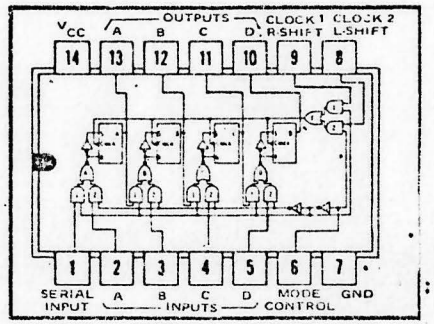
SN 7408



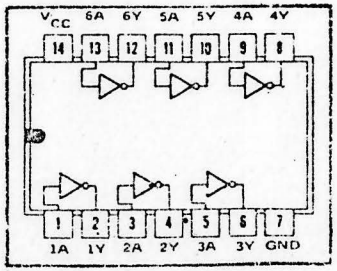
SN 7493



SN 74 H 74



SN 7495



SN 7404

Fig. 5

gestellt. Verzichtet man auf diese Finesse, so müssen die Clear-Eingänge der FF SN 74H74 an V_{cc} geschaltet werden.
Die Anzeige des Resultats erfolgt über Glühlampen.
Da die Clockfrequenz der Multiplikatorregister auf 10 MHz beschränkt ist, kann die theoretische Rechengeschwindigkeit nicht verifiziert werden. Wir beabsichtigen, dies in weiteren Versuchen zu tun, sobald schnellere Register verfügbar werden. Auch kann der Rechner später zwanglos zu einem BCD-Binär-Konverter ergänzt werden, indem man anstelle des Multiplikanden ein 4-Zeilen ROM einfügt, das durch eine Steuerlogik kontrolliert wird.

Fig. 6 schliesslich enthält die Anschlussschemata aller ICs, die im Multiplizierwerk verwendet werden.

ANHANG D: DAS NEUE EEG-ANALYSESYSTEM DER WISSENSCHAFTLICHEN DATENVERARBEITUNG GMBH, MUENCHEN

Vergleich eines industriellen
Produkts mit unserem Projekt.

Inhaltsübersicht

1. Einleitung

In diesem Bericht soll der neue INTIN-EEG-Prozessor von WDV unserem geplanten Analysesystem gegenübergestellt werden. Die WDV-Anlage ist unter allen marktgerechten EEG-Systemen dasjenige, welches unserer Zielsetzung am nächsten kommt. Abgesehen von Programmsystemen auf Grossrechenanlagen für medizinische Datenverarbeitung ist es auf die Durchschnittsanforderungen der EEG-Aerzte derart zugeschnitten, dass es, vorderhand ohne Konkurrenz, eine echte Marktlücke füllt. Die festverdrahteten Fourieranalysatoren, die besonders in jüngster Zeit zum Kauf angeboten werden, sind durchwegs Systeme, die nicht für medizinische Zwecke gebaut wurden. Sie haben einerseits Spezifikationen, die der Arzt nicht braucht (und deshalb auch nicht bezahlen möchte), andererseits fehlen solche, die für EEG-Analysen unumgänglich sind. Ausserdem sind alle diese Prozessoren auf Fourieranalyse und Korrelation beschränkt.

Es genügt deshalb, wenn wir WDV unseren Absichten gegenüberstellen. Dieser Bericht zeigt, dass unsere Zielsetzungen nur scheinbar mit WDV verwandt sind. Bei genauerem Hinsehen bestehen Unterschiede, die beide eigentlich kaum vergleichbar machen.

In folgender Tabelle seien die hauptsächlichsten Unterschiede zusammengefasst.

Kriterium	WDV	Inst. für Techn. Physik
zentrale Recheneinheit	Interdata M70 Minicomputer	Hardware FFT mit speziellen Zusätzen
Rechenstrategie	abgeschlossene Software-Programmsysteme	Transformationsalgorithmen in Hardware, Steuerung und Nachverarbeitung mit Mikrocomputer
Programmauswahl	8 Programme mit herkömmlichen Algorithmen für Einzel- und Summenpotentiale	FFT als Kerntransformation, Frequenzlupe nach Openheim Ausbau auf zusätzliche Algorithmen für Transientenanalyse
Speicherung	teurer Kernspeicher	billige Schieberegister
Ausbau- und Anpassfähigkeit	im Rahmen des M70 Minicomputers	hardwaremässig unbeschränkt
Spektralanalyse		
Zeitfenster	4 sec - Segmente aneinandergereiht ohne Mittelung des einzelnen Segmentes	sukzessive starke Ueberlappung von 4 sec-Segmenten und Mittelung
Abtastfrequenz	64 Hz	128 Hz
Frequenzachse	linear bis 30 Hz	linear bis 50 Hz und nichtlineare Verformung
Phaseninformation	kein Zugriff	komplexe Transformation
Programmspezifikationen, Fehlerangaben, etc.	sehr mangelhaft (Fehlerangaben fehlen völlig)	genau bestimmt

2. Beschreibung des WDV-Prozessors

WDV bringt gegenwärtig ein Computersystem für EEG-Analyse auf den Markt, das die Anforderungen der forschenden Aerzte im klinischen Bereich (z.B. EEG und Psychopharmaka) befriedigen dürfte. Zur Zeit ist das System noch nicht voll funktionstüchtig. Im Endausbaus jedoch verfügt der Benützer über eine Anzahl bewährter Algorithmen zur einfachen und raschen Quantifizierung von Einzel- und Summenpotentialen.

Folgende Gesichtspunkte waren bei der Entwicklung des Prozessors wegleitend:

1. Das Gerät soll on-line alle gängigen EEG-Analyseverfahren kontinuierlich auf mehreren Kanälen gleichzeitig anwenden können;
2. Es soll von ungelerntem Hilfspersonal benutzbar sein (z.B. Elimination von Artefakten),
3. Es soll flexibel und ausbaufähig sein, da mit fortschreitender Kenntnis der EEG-Genese sich auch die Analyseverfahren verändern werden.

Das Ergebnis dieser Bemühungen ist der INTIN-EEG-Prozessor, der aus einem Satz spezieller Programmsysteme besteht, die auf einem geeigneten Standard-Minicomputer (INTERDATA M70) mit der nötigen z.B. von WDV entwickleten Peripherie ablaufen.

Folgende acht Programmsysteme sind im Lieferumfang eingeschlossen:

1. Powerspektrum (Auto- und Kreuz-Spektrum)
2. Intervall-Amplituden-Analyse (Intervall-Histogramm, Amplituden-Intervall-Verteilung)
3. Averaging
4. Amplitudenhistogramm
5. Auto- und Kreuzkorrelation
6. Prä- und Poststimulushistogramm
7. Interspike-Intervall-Histogramm
8. Verbund-Intervall-Histogramm

An weiteren Programmen wird gearbeitet (z.B. Aktivität, Mobilität, Komplexität nach Hjorth und Mustererkennung).

Der Programmablauf wird durch vier Tasten gesteuert:

- START: Programm fortsetzen
- STOP: Programm anhalten
- CLEAR: zurück zur Anfangsstellung
- READ OUT: Ende Aufnahmeprogramm und Ausgabe

Für die Erstellung neuer Programmsysteme stehen Programmiersprachen (Fortran, Assembler) zur Verfügung. Die Minimalkonfiguration der Hardware besteht aus einem

- mikroprogrammierten Rechner INTERDATA M70 mit 16 K Kernspeichern
- Prozessinterface mit 16 Analog-Eingängen, 4 Analog-Ausgängen
- Teletype
- Speicherdisplay

P r e i s : ca. DM 130'000.--

Die Peripherie lässt sich wegen der modularen Struktur einfach erweitern.

Der zentrale Rechner INTERDATA M70 ist ein moderner Minicomputer mit Eigenschaften, die einen besonders raschen Programmablauf gewährleisten:

1. Mikroprogramme (16-Bit-ROM)
2. Sechzehn 16-Bit-Register (als Akkumulatoren, Indexregister, etc. verwendbar)
3. kurze Zykluszeit (< 1 Mikrosekunde)
4. direkte Adressierung im ganzen Kernspeicher (Speicher von 8192 - 65'536 Bytes)
5. flexibles I/O-System

Einen Einblick in die EEG-Software vermittelt z.B. eine kurze Beschreibung des Spektralanalyseprogramms, das in unserem Zusammenhang besonders interessant ist. Das Programm POSPEC nimmt eine Spektralanalyse von mehreren Kanälen (bis zu 8) EEG vor.

Es können Auto- und Kreuzspektren, sowie die dazugehörigen Kohärenzfunktionen in Echtzeit berechnet werden. Es wird die Fast-Fourier-Transformation nach Cooley und Tuckey verwendet. Die Dauer der Analyse ist in ganzen Vielfachen der Segmentlänge von 4 Sekunden wählbar, wobei die gewünschten Spektren von 0,25 - 30 Hz mit einer Auflösung von 0,25 Hz berechnet und aufsummiert werden. Die Abtastfrequenz beträgt 64 Hz. Die Resultate der Summationen werden auf dem Bildschirm laufend dargestellt.

Die Eingaben zur Steuerung des Programms erfolgen mit Hilfe zweier Dialoge über die Tastatur des Terminals:

- Dialog vom Wissenschaftler (Kanalnummer, Anzahl 4 sec-Segmente, Triggerung, Eichung, Rechteck-oder Hammingfenster)
- Dialog mit der Assistentin (Datum, Uhrzeit).

Die Ablaufsteuerung wird mittels vier Tasten vorgenommen. Ein Datensegment wird selbsttätig verworfen, wenn 10 % der Daten innerhalb des Segmentes die ADC-Grenze erreichen (Artefakte, Uebersteuerung). Die Spektren- und Kohärenzfunktionen werden zunächst auf einem Bildschirm dargestellt. Auch andere Ausgabeeinheiten (z.B. Plotter) sind wählbar. Ausserdem werden für fünf feste Frequenzbereiche (Gesamt, Delta, Theta, Alpha, Beta-Band) gewisse Parameter berechnet:

- die absolute Leistung je Band
- die dominante Frequenz je Band
- eine sogenannte Schärfe, mit der diese auftritt
- die 10 %, 50 %, 90 % - Grenzen der Leistung je Band
- die Schiefe des Spektrums.

3. Gegenüberstellung und Kritik

Der WDV-EEG-Prozessor ist unter allen marktgerechten EEG-Systemen wohl dasjenige, welches unserer Zielsetzung am nächsten kommt. Dabei bestehen aber Divergenzen, die beide eigentlich kaum vergleichbar machen.

WDV arbeiten mit reinen Software-Programmen auf einem Allzweck-Minicomputer. Eine möglichst hohe on-line Verarbeitungskapazität wurde offensichtlich angestrebt. Sie wird erreicht durch die oben aufgezählten Leistungsmerkmale des Zentralrechners und - im Spektralanalyseprogramm - durch weitgehende statistische Einschränkung, die in den Datenblättern kaum zum Ausdruck kommt. Die Idee der 4 sec-Segmente, die zeitlich aneinander gereiht und im Frequenzbereich gemittelt werden, entspricht - interessanterweise - grundsätzlich unserer Planungsabsicht. WDV verschweigt aber, dass ein Spektrum eines einzigen Segmentes statistisch keine Aussagekraft hat, falls nicht gemittelt wird (Auflösung, $\Delta f = 0{,}25$ Hz!).

Wegen des Leakage-Effekts ist das Rechteckfenster für so kurze Datensegmente nicht am Platz. Beim Hamming-Window geht aber Information verloren, falls keine Ueberlappung der Datenpakete vorgesehen ist. WDV reiht sie einfach aneinander; bei uns ist eine starke Ueberlappung vorgesehen, sodass man von einem"verschobenen Zeitfenster" sprechen kann. Dieses Vorgehen macht die Spektralanalyse möglicherweise auch für Transienten, jedenfalls aber für Instationarität sensibel.

64 Hz Abtastfrequenz von WDV ist ebenfalls eine rigorose Einschränkung. Wir planen 128 Hz, da EEG-Spektren bis etwa 50 Hz üblich sind und eine gute Anti-Aliasing-Filtrierung möglich sein soll.

Diese beiden Einschränkungen aufgehoben würden die erstaunliche on-line Kapazität von WDV zunicht machen. Unsere Lösung mit einem Hardware FFT hat eine so hohe Durchflussrate, dass selbst bei vielen EEG-Kanälen überhaupt keine Kompromisse nötig sind.

Bei WDV ist für Analyse und Zwischenspeicherung ein grosser und teurer Kernspeicher von Nöten (deshalb wurden am Burghölzli z.B. 24 k-Bytes gekauft). Wir verwenden preisgünstige dynamische Schieberegister für Puffer-Speicherung und Analyse, die nachfolgende Parameterberechnung könnte evtl. von einem Mikrocomputer bewältigt werden. Eine digitale Frequenzverformung fällt bei WDV vollends weg. Sie ist für den an Parameterextraktion interessierten EEG-Forscher aber wichtig, und beansprucht an Rechenzeit etwa dieselbe Grössenordnung wie FFT. Ihre hardwaremässige Darstellung ist (übrigens sehr elegant, vergleiche Openheim: Digital-Frequency-Warping) für unsere Zwecke unumgänglich.

WDV ist vorderhand sehr schlecht dokumentiert. Programmbeschreibungen existieren von 8 deren 2, die nichtssagend sind im Vergleich etwa zu Subroutinen-Beilagen des ETH Rechenzentrums. Rücktransformationen (evtl. logarithmisch zur Cepstrumgewinnung) sind nicht möglich. Es gibt überhaupt keinen Zugriff zur Phaseninformation. WDV ist gut und recht für Forscher, die sich nicht mit den Feinheiten der EEG-Analyse abgeben wollen, sondern altbekannte Algorithmen auf Routinenexperimenten anwenden.

Dies aber war und ist nicht unser Ziel. Wir wollen dem avancierten EEG-Analysten echt neue Methoden in die Hand geben, die ohne a priori Einschränkung in jeder Hinsicht ausbaufähig sind.