237

# Notice of Opposition to a European Patent

To the
European Patent Office

Tabulation marks

for EPO use only

| | | |
|---|---|---|
| **I. Patent opposed** | Opp. No. | OPPO (1) |
| **Patent No.** | 0 649 578 | |
| **Application No.** | 93914555.3 | |
| **Date of mention of the grant in the European Patent Bulletin (Art. 97(4), 99(1) EPC)** | 14.05.2003 | |

**Title of the invention:**

Digital filter having high accuracy and efficiency

**II. Proprietor of the Patent** first named in the patent specification: Lake Technology Limited

Opponent's or representative's reference (max. 15 spaces): U 03 021 DK — OREF

**III. Opponent** — OPPO (2)

| | | | |
|---|---|---|---|
| **Name** | TC Electronic A/S | | |
| **Address** | Sindalsvej 34<br>8240 Risskov | | |
| **State of residence or of principal place of business** | Denmark | | |
| **Telephone/Telex/Fax** | | | |
| **Multiple opponents** | □ further opponents see additional sheet | | |

**IV. Authorisation**

1. **Representative** (Name only one representative to whom notification is to be made) — OPPO (9)

| | |
|---|---|
| **Name**<br>**Address of place of business** | OLESEN, Kaj<br>Patentgruppen ApS<br>Arosgaarden<br>Aaboulevarden 31<br>8000 Aarhus C<br>Denmark |
| **Telephone/Telex/Fax** | +45 86 19 20 00 \| \| +45 86 19 91 91 |
| **Additional representative(s)** | ☒ (on additional sheet/see authorisation) — OPPO (5) |

(A) E620

2. **Employee(s)** of the opponent authorised for these opposition proceedings under Art. 133(3) EPC — Name(s):

**Authorisation(s)** To 1./2.

☒ not considered necessary

□ has/have been registered under No.

□ is/are enclosed

EPO Form 2300.1 04.93 (inc. ed. 12/97)

for EPO use only

## V. Opposition is filed against

— the patent as a whole ☒

— claim(s) No(s).

## VI. Grounds for opposition:

**Opposition is based on the following grounds:**

(a) the subject-matter of the European patent opposed is not patentable (Art. 100(a) EPC) because:

- it is not new (Art. 52(1); 54 EPC) ☒
- it does not involve an inventive step (Art. 52(1); 56 EPC) ☒
- patentability is excluded on other grounds, i. e. Art. □

(b) the patent opposed does not disclose the invention in a manner sufficiently clear and complete for it to be carried out by a person skilled in the art (Art. 100(b) EPC; see Art. 83 EPC). ☒

(c) the subject-matter of the patent opposed extends beyond the content of the application/ of the earlier application as filed (Art. 100(c) EPC, see Art. 123(2) EPC). ☒

## VII. Facts and arguments

(Rule 55(c) EPC)
presented in support of the opposition are submitted herewith on a separate sheet (annex 1) ☒

## VIII. Other requests:

Oral proceedings are requested for the event that the patent opposed is not to be revoked as requested on the written submission

for EPO use only

**IX. Evidence presented**

Enclosed = ☒

will be filed at a later date = □

A. **Publications:**

Publication date

1 A1 EP 0 649 578 (The opposed patent as published)

Particular relevance (page, column, line, fig.):
See Facts and Arguments

2 A2 WO 94/01933 A (The international application as filed)

Particular relevance (page, column, line, fig.):
See Facts and Arguments

3 A3 WO 88/03341 A (published on 5 May 1988)

Particular relevance (page, column, line, fig.):
See Facts and Arguments

4 A4 Digital Equalization Using Fourier Transform Techniques, 2694 (B-2) by Barry Kulp (published on 3 November 1988)

Particular relevance (page, column, line, fig.):
See Facts and Arguments

5 A5 EP 0 250 048 A (published on 23 December 1987)

Particular relevance (page, column, line, fig.):
See Facts and Arguments

6 A6 US 4 992 967 A (published on 12 February 1991)

Particular relevance (page, column, line, fig.):
See Facts and Arguments

7 A7 Multidelay Block Frequency Domain Adaptive Filter by Jia-Sien Soo and Khee K. Pang (published on 2 February 1990)

Particular relevance (page, column, line, fig.):
See Facts and Arguments

Continued on additional sheet ☒

B. **Other evidence**

A12 Testimony of prior disclosure by Knud Bank Christensen

Continued on additional sheet □

for EPO use only

## X. Payment of the opposition fee is made

☒ as indicated in the enclosed voucher for payment of fees and costs (EPO Form 1010)

☐

## XI. List of documents

| Enclosure No. | | No. of copies | |
|---|---|---|---|
| 0 | ☒ Form for notice of opposition | 2 | (min. 2) |
| 1 | ☒ Facts and arguments (see VII.) | 2 | (min. 2) |
| 2 | Copies of documents presented as evidence (see IX.) | | |
| 2a | ☒ — Publications | 2 | (min. 2 of each) |
| 2b | ☒ — Other documents | 2 | (min. 2 of each) |
| 3 | ☐ Signed authorisation(s) (see IV.) | | |
| 4 | ☒ Voucher for payment of fees and costs (see X.) | 1 | |
| 5 | ☐ Cheque | | |
| 6 | ☒ Additional sheet(s) | 4 | (min. 2 of each) |
| 7 | ☒ Other (please specify here): Form 1037 | 2 | |

## XII. Signature of opponent or representative

Place Aarhus

Date 16 February 2004

Kaj Olesen
European Patent Attorney

Please type name under signature. In the case of legal persons, the position which the person signing holds within the company should also be typed.

**Additional Sheet to EPO Form 2300**

Additional Representatives:

ELMEROS, Claus
SCHMIDT, Jens Jørgen

Aarhus, 16 February 2004

Kaj Olesen
Authorised Representative

**Additional Sheet to EPO Form 2300**

IX. Evidence presented – Additional publications:

8. A8 High-speed convolution and correlation by Thomas G. Stockham Jr. (published in 1966)

Particular relevance (page, column, line, fig.):
See Facts and Arguments

9. A9 Real Signals Fast Fourier Transform: Storage Capacity and Step Number Reduction by Means of an Odd Discrete Fourier Transform by J. L. Vernet (published in October 1971)

Particular relevance (page, column, line, fig.):
See Facts and Arguments

10. A10 Odd-Time Odd-Frequency Discrete Fourier Transform for Symmetric Real-Valued Series by G. Bonnerot and M. Bellanger (published in March 1976)

Particular relevance (page, column, line, fig.):
See Facts and Arguments

11. A11 Digital Processing of Signals – Theory and practice by Maurice Bellanger (published in 1984 and 1989 and reprinted in April 1990)

Particular relevance (page, column, line, fig.):
See Facts and Arguments

## GROUNDS FOR OPPOSITION

### Facts and Arguments

In support of the Notice of Opposition as set forth in the accompanying Form 2300, the following Grounds for Opposition are presented, in accordance with the indication of section VII of this Form.

The following annexes presented under section IX of the Notice of Opposition are referred to in this communication; the numbering will be adhered to the rest of this communication:

| | |
|---|---|
| Annex 1 (A1): | EP-B-0 649 578 - The opposed patent as published |
| Annex 2 (A2): | WO 94/01933 A - The International application as filed |
| Annex 3 (A3): | WO 88/03341 A (published on 5 May 1988) |
| Annex 4 (A4): | Digital Equalization Using Fourier Transform Techniques, 2694 (B-2) by Barry Kulp (published on 3 November 1988) |
| Annex 5 (A5): | EP 0 250 048 A (published on 23 December 1987) |
| Annex 6 (A6): | US 4 992 967 A (published on 12 February 1991) |
| Annex 7 (A7): | Multidelay Block Frequency Domain Adaptive Filter by Jia-Sien Soo and Khee K. Pang (published on 2 February 1990) |
| Annex 8 (A8): | High-speed convolution and correlation by Thomas G. Stockham Jr. (published in 1966) |
| Annex 9 (A9): | Real Signals Fast Fourier Transform: Storage Capacity and Step Number Reduction by Means of an Odd Discrete Fourier Transform by J. L. Vernet (published in October 1971) |
| Annex 10 (A10): | Odd-Time Odd-Frequency Discrete Fourier Transform for Symmetric Real-Valued Series by G. Bonnerot and M. Bellanger (published in March 1976) |
| Annex 11 (A11): | Digital Processing of Signals – Theory and practice by Maurice Bellanger (published in 1984 and 1989 and reprinted in April 1990) |
| Annex 12 (A12): | Testimony of prior disclosure by Knud Bank Christensen |

### Article 100(c) EPC; Article 123(2) EPC

Notwithstanding the other grounds of opposition, the patent is opposed on the ground that the subject-matter of the patent opposed extends beyond the content of the application as filed

#### A. Frequency domain transformation

In order to overcome prior art during the examination of the application, the applicant has introduced the phrasing "frequency-domain transformation".

The applied broad phrasing has no basis in the application as filed and, consequently, the subject-matter of the patent opposed extends beyond the content of the application as filed.

For this reason alone, the patent should be revoked.

In this context it should be noted that the applicant emphasises that the nature of this frequency-domain transformation is significant in order to distinguish the claimed invention over e.g. A3.

**B. Relationship between input sample length and overlap delay, P ≥ 2N-1**
During the examination of the application, the applicant has redrafted claim 1. In order to overcome prior art, the applicant has introduced a relationship between the delay length N and the input sample length P.

Nowhere in the application as filed may any disclosure be found indicating that P may be anything else than equalling 2N.

It is noted that the applicant apparently is aware of this fact as the next claim, - claim 2 -, has been used for the purpose of securing this position, namely that P equals 2N.

The applied broad phrasing of claim 1 has no basis in the application as filed and, consequently, the subject-matter of the patent opposed extends beyond the content of the application as filed.

For this reason alone, the patent should be revoked.

**C. The summing step of claim 1**
The step of

> *(iii) summing together said M frequency-domain filtered blocks to form a single frequency-domain output block;*

has no basis in the description and the accompanying drawings. In particular, it is noted that Fig. 8 has been applied as the basis for the amended claim 1 during the prosecution of the application. It is noted that fig. 8 of the application as filed clearly illustrates that the frequency domain filtered blocks must be multiplied.

The applied broad phrasing of claim 1 has no basis in the application as filed and, consequently, the subject-matter of the patent opposed extends beyond the content of the application as filed.

For this reason alone, the patent should be revoked.

**D. Claim 5 - filtering of input blocks**

Claim 5 of the opposed patent

> *A method as claimed in any previous claim, wherein said step of combining together frequency-domain input blocks comprises element by element multiplication of said frequency-domain input blocks.*

There is no basis for claiming this filtering of the input blocks in the application as filed and claim 5 has no basis in the application as filed and, consequently, the subject-matter of the patent opposed extends beyond the content of the application as filed.

For this reason alone, the patent should be revoked.

**E: Claim 11**

Claim 11 of the opposed patent states:

> *wherein said method is applied in parallel to a series of predetermined portions*

The application as filed gives no explanation of the phrasing "applied in parallel". Consequently, claim 11 has no basis in the application as filed and, consequently, the subject-matter of the patent opposed extends beyond the content of the application as filed.

For this reason alone, the patent should be revoked.

**Article 100(b) EPC; Article 83 EPC:**

Notwithstanding the above issue, the patent is also opposed on the ground that the invention is not disclosed in a manner sufficiently clear and complete for it to be carried out by a person skilled in the art.

**A. Substantially M (claim 1, page 9 line 35 of A1)**

The skilled person would, in the light of the description have difficulties in determining what substantially M means. No hint in the description may be found. A correction is even less obvious as the term "substantially" indicates that something has to be different from the exact "M".

It is therefore respectfully submitted for this reason alone that the opposed patent does not fulfil the provisions of Art 83 EPC in conjunction with Art. 100(b) and the patent should be revoked.

**B. Reference sign M (claim 1, page 9 lines 35, 47-50 and claim 10, page 10 line 27 of A1)**

The skilled person would, in the light of the description and drawings have difficulties in determining what the reference sign M in claim 1 and 10 refers to. The use of M in e.g. figures 4, 9 and 15 and the description page 5 lines 5-13 and page 5 lines 42-44 seem not to be consistent with the use of M in e.g. figures 6, 7, 8 and 14 and the description page 5 lines 16-18 and page 5 line 46. Several different uses of the reference sign M is thus apparently used among each other, leaving the skilled man no clue to the understanding of reference sign M used in claim 1 and claim 10 thereby leaving the skilled person with severe problems as regards the understanding of the invention. This practice is furthermore not in agreement with Rule 32(2)(i) EPC, 2nd paragraph.

It is therefore respectfully submitted for this reason alone that the opposed patent does not fulfil the provisions of Art 83 EPC in conjunction with Art. 100(b) and the patent should be revoked.

**C. Reference sign N (claim 1, page 9 lines 33 and 56, and page 10 line 1 of A1, and furthermore claims 2, 3, 4, 9 and 10 of A1)**

The skilled person would, in the light of the description and drawings have difficulties in determining what the reference sign N in claims 1, 2, 3, 4, 9 and 10 refers to. The use of N in e.g. figures 10, 13 and 16 and the description seem not at all to be consistent with the use of the reference sign N in the above-mentioned claims. Other reference signs, e.g. L and M, however in some drawings appears to show what is referred to as N in the claims. The different uses of the reference sign N thus completely confuses the issue, leaving the skilled man no clue to the understanding of reference sign N in claims 1, 2, 3, 4, 9 and 10. This practice is furthermore not in agreement with Rule 32(2)(i) EPC, 2$^{nd}$ paragraph.

It is therefore respectfully submitted for this reason alone that the opposed patent does not fulfil the provisions of Art 83 EPC in conjunction with Art. 100(b) and the patent should be revoked.

**D. Incomprehensible language of approved text (Claim 3, page 10 lines 5-8 of A1)**
A third party would have difficulties in determining the teaching of the description in the light of the wording of claim 3. Furthermore, the skilled person would, in light of the description and drawings have difficulties in exercising claim 3 due to uncertainty of the exact meaning.

It is therefore respectfully submitted for this reason alone that the opposed patent does not fulfil the provisions of Art 83 EPC in conjunction with Art. 100(b) and the patent should be revoked.

**E. Unclear wording: "in parallel to a series" (Claim 11, page 10 line 30 of A1)**
A third party would, in the light of the description and drawings have difficulties in determining the scope of the protection of claim 11, in particular the phrase: "... said method is applied in parallel to a series of predetermined portions ...". The meaning of the word "parallel" is not explained in the description, and may, by a skilled person, be interpreted in several different ways.

It is therefore respectfully submitted for this reason alone that the opposed patent does not fulfil the provisions of Art 83 EPC in conjunction with Art. 100(b) and the patent should be revoked.

**<u>Article 100(a) EPC; Articles 52(1) and (2) EPC:</u>**

Notwithstanding the above issues, the patent is also opposed on the ground that the subject-matter of the claims is not patentable in the sense to Article 52 EPC.

**Lack of Novelty and Inventive Step:**

**Claim 1:**

> *A method of finite impulse response (FIR) filtering an input signal using a predetermined portion of a desired time domain impulse response representing a filter characteristic and specified in terms of time-domain*

*impulse response values, so as to produce a filtered output signal, the method comprising the steps of:*

*(a) dividing said input signal into overlapping successive input blocks of P samples, with each successive input block being delayed by N samples relative to the previous input block, where P≥2N-1;*

*(b) creating each of substantially M frequency-domain coefficient blocks by:*
   *(i) dividing said predetermined portion of the desired time domain impulse response into a series of segments;*
   *(ii) computing a frequency-domain transformation of each segment to form a corresponding one of said M frequency-domain coefficient blocks; and*

*(c) for each of said input blocks:*
   *(i) computing a frequency-domain transform of said input block to form a corresponding frequency-domain input block;*
   *(ii) combining together the most recent M successive said frequency-domain input blocks with M frequency-domain coefficient blocks, to produce M frequency-domain filtered blocks;*
   *(iii) summing together said M frequency-domain filtered blocks to form a single frequency-domain output block;*
   *(iv) computing an inverse transform, which is the inverse of said frequency-domain transform, of said frequency-domain output block to form a time-domain output block;*
   *(v) discarding predetermined portions of said time-domain output block, to produce a new set of N output samples; and*
   *(vi) outputting said N output samples as a portion of said output signal.*

Novelty (A3)

A3 discloses the method of finite impulse response filtering an input signal (X(n) of fig. 1) using a predetermined portion of a desired time domain impulse response representing a filter characteristic (12, 17, fig. 1)

It is noted, contrary to the statements of the applicant of 5 October 1999, that the update control 17 is outputting frequency domain impulse response coefficients corresponding to a desired time domain impulse response. In other words, the established frequency domain impulse response blocks of the filter 12, fig. 1 are direct **predetermined** counterparts to desired corresponding time domain impulse response functions **and** they are established directly on the basis of a time domain signal e(n).

Moreover, it is noted that an initial state of the system of fig. 1 (A3), that is when e(n) equals zero, the initial settings of the FIR filter 12 necessarily have to represent the desired time domain impulse response. Therefore, the initial state of the system features all the steps of claim 1 of the opposed patent.

Consequently, every feature of claim 1 is known from A3 and lacks novelty.

Inventive step (A3+common knowledge)

The only difference between the invention as claimed and the further iterations (the initial iteration(s) excluded as explained above) performed by the disclosure of A3 is apparently the way the applied impulse response is established. The filtering processing in the filter 10, 11, 12, 15 and 16 of A3 is exactly the same of the filter applied in the claimed filtering method of the opposed patent and inherits the same benefits as described in the opposed patent, especially with respect to latency.

The man skilled in the art, faced with the problem of establishing desired (known!) M blocks would know that a time-domain counterpart exist on the basis on common knowledge. In particular such establishment of a correlation between a time domain impulse response would be obvious as A3 specifies the delays between the disclosed partial frequency domain impulse response.

Thus, it is respectfully submitted that claim 1 of the opposed patent A1 lacks inventive step.

Inventive step (A3+A8)
If, however the skilled person would be unable to apply such common knowledge for some reasons he would know from A8 page 231, first paragraph, that a time domain impulse response (i.e. the kernel s(j)) may be split into packets, each of which may be considered separately,

Faced with the problem of transforming a desired time-domain impulse response into a frequency domain response suitable for use in the filter of Fig. 1 of A3, the skilled man would know from A8 that such a transformation exist and that each packet may be dealt with separately.

Thus, it is respectfully submitted that claim 1 of the opposed patent lacks inventive step with respect to A3 in the light of A8.

In this context, it is noted that claim 1 of the opposed patent is quite relaxed to the wording of how the parts of the time impulse responses are brought into the frequency domain, namely by the wording "frequency domain transformation". In other words, the update control 17 of fig. 1 and the associated description of A3 definitely performs such "frequency domain transformation" insofar the basis of the update control is a time domain input.

Novelty (A4)
A4 generally discloses equalisation using Fourier transformation.
Initially, page 5, third paragraph specifies that authors will use FFT to convert both the input and impulse response from time domain representations into frequency domain representation, perform the fast convolution and than perform and inverse FFT operation in order to obtain the desired output signal in the time domain.

Page 19, lines 9-24 specifically discloses an optimisation of reducing the pipeline delay by segmenting the impulse response. Basically, this paragraph discloses the use of separate discrete components as disclosed in fig. 6 of the opposed patent.

The next two paragraphs of A4 disclose optimisations with respect to sharing of both FFT on the input signal and inverse FFT.

In other words, as the disclosure of A4, last part of 2nd paragraph specifies an example of 8 equally sized blocks and that the forward FFT on the input signal is only applied once, 3rd paragraph of page 19, all features of claim 1 is known from A4.

Consequently all features of the opposed claim 1 is known from A4 and lacks novelty.

Novelty (A5)
Page 3, line 46 to page 4, line 20 of A5 disclose every feature of claim 1 of the opposed patent. In particular it is noted that the weighting factor W(p;m) may be considered as points of a 2N point DFT performed on time domain weighting factor w(i;m), which represent values of the impulse response w(k) during block m.

Consequently, every feature of claim 1 is known from A5 and lacks novelty.

Novelty (A6)
Fig. 10 of A6 illustrates a filter structure corresponding to claim 1 of the opposed patent.

Consequently, every feature of claim 1 is known from A6 and lacks novelty.

Novelty (A7)
Fig. 1 of A7 illustrates a filter structure corresponding to claim 1 of the opposed patent, as it is noted that the initial impulse response is established on the basis of a zero-error on the output in the time domain.

Consequently, every feature of claim 1 is known from A7 and lacks novelty.

Novelty (A12)
A12 represents prior disclosure of a fast convolution filter comprising different sized fast convolution filters in parallel, where the output of those filter were added in order to obtain the desired input, all for the purpose of obtaining the desired low latency.

Consequently, every feature of claim 1 is known from A12 and lacks novelty.

**Claim 2:**

*A method as claimed in claim 1, wherein P is equal to 2N.*

Novelty (A3)
For the initial state:
Fig. 1 of A3 and the associated description page 13, lines 4-12 discloses that P equals 2N, namely in block 10 of fig. 1.

Consequently, every feature of claim 2 is known from A3 and lacks novelty.

Inventive step (A3)

For the states following the initial filterings, fig. 1 of A3 and the associated description page 13, lines 4-12 discloses that P equals 2N, namely in block 10 of fig. 1.

Consequently, it is submitted that claim 2 of the opposed patent lacks inventive step over A3.

Inventive step (A4+A3)
Starting from A4, faced with the problem of choosing a specific relationship between the applied overlap delay N and the period of the input signal P, the skilled person would look into A3, e.g. fig. 7 and learn that P may be chosen as 2 times N, that is P= 2N' and a delay of N' with the terms of A3.

Consequently, claim 2 lacks inventive step over A4 in the light of A3.

**Claim 3:**

> *A method as claimed in any one of the preceding claims wherein at least one of said frequency-domain transformation said frequency-domain transform comprises a 2N-point Real fast Fourier transform producing and N complex values in said frequency-domain coefficient block.*

Inventive step (A3/A4/A5/A6/A7/A12+A9/A10/A11)
The skilled person would, when starting from A3, A4, A5, A5, A7 or A12 faced with the problem of eliminating unnecessary operations for a symmetric real input series look into A9, A10 or A11 and know that such a Fourier transformation exists and may be applied for the purpose.

Consequently, it is submitted that claim 3 of the opposed patent lacks inventive step over A3, A4, A5, A5, A7 or A12 in combination with A9, A10 or A11.

Inventive step (A4+A9/A10/A11)
Starting from A4, faced with the problem of obtaining a more efficient Fourier transformation look into A9, A10 or A11 and use the claimed modified FFT

Consequently, claim 3 lacks inventive step over A4 in the light of A9, A10 or A11.

**Claim 4:**

> *A method as claimed in claim 3, wherein said inverse transform comprises an N complex value fast Fourier inverse transform producing 2N-point Real values in said output block.*

Inventive step (A3/A4/A5/A6/A7/A12+A9/A10/A11)
The skilled person would, when starting from A3, A4, A5, A5, A7 or A12 faced with the problem of eliminating unnecessary operations for a symmetric real input series look into A9, A10 or A11 and know that such a Fourier transformation exists and in particular that an inverse transformation exists and may be applied for the purpose.

Consequently, it is submitted that claim 2 of the opposed patent lacks inventive step over A3, A4, A5, A5, A7 or A12 in combination with A9, A10 or A11.

Inventive step (A4+A9/A10/A11)
Starting from A4, faced with the problem of obtaining a suitable inverse Fourier transformation corresponding to the Fourier transformation of claim 3 look into A9, A10 or A11 and find the corresponding inverse modified FFT.

Consequently, claim 4 lacks inventive step over A4 in the light of A9, A10 or A11.

**Claim 5:**

> *A method as claimed in any previous claim, wherein said step of combining together frequency-domain input blocks comprises element by element multiplication of said frequency-domain input blocks.*

This claim has no basis in the application as originally filed (A2).

**Claim 6:**

> *A method as claimed in any previous claim, wherein said summing together of frequency-domain filtered blocks comprises element by element addition of the blocks.*

Novelty (A3)
For the initial state fig. 7 of A3 discloses that the summing of the frequency domain filtered blocks may comprise element by element addition of the blocks, see 12-0 to 12-(2n'-1) of fig. 7.

Consequently, every feature of claim 6 is known from A3 and lacks novelty.

Inventive step (A3)
For the states following the initial filterings, fig.7 of A3 discloses that the summing of the frequency domain filtered blocks may comprise element by element addition of the blocks, see 12-0 to 12-(2n'-1) of fig. 7.

Consequently, it is submitted that claim 6 of the opposed patent lacks inventive step over A3.

Novelty (A4)
Page 19 paragraph 2 of A4 discloses that "future"/"current" segments may be summed up in the frequency domain.

Consequently, claim 6 lacks novelty.

**Claim 7:**

> *A method as claimed in any previous claim, wherein said combining and said summing are carried out in a single operation with the successive results of the combining operations being accumulated into the frequency-domain output block.*

Novelty (A3)
For the initial state fig. 7 of A3 discloses that the combining and the summing may be performed in one operation, see 12-0 to 12-(2n'-1) of fig. 7.

Consequently, every feature of claim 7 is known from A3 and lacks novelty.

Inventive step (A3)
For the states following the initial filtering, fig. 7 of A3 discloses that the combing and the summing may be performed in one operation, see 12-0 to 12-(2n'-1) of fig. 7.

Consequently, it is submitted that claim 7 of the opposed patent lacks inventive step over A3.

Inventive step (A4+common knowledge)
Starting from A4, faced with the problem of reducing the memory consumption of a filter, the skilled person would choose a trivial ladder-type design.

Consequently, claim 7 lacks inventive step over A4 in the light of general common knowledge.

**Claim 8:**

> *A method as claimed in any previous claim, wherein time-domain output block is of length 0.5 P.*

Novelty (A3)
For the initial state fig. 7 of A3 discloses that the length of the outputs blocks are 0.5 x P, that is N'.

Consequently, every feature of claim 8 is known from A3 and lacks novelty.

Inventive step (A3)
For the states following the initial filtering, fig. 7 of A3 discloses that the length of the outputs blocks are 0.5 x P, that is N'.

Consequently, it is submitted that claim 8 of the opposed patent lacks inventive step over A3.

Inventive step (A4+A3)
Starting from A4, faced with the problem of choosing the length of an output block, the skilled person would look into A3, e.g. fig. 7 and learn that the output block may be of the length N', that is P/2.

Consequently, claim 8 lacks inventive step over A4 in the light of A3.

**Claim 9:**

> *A method as claimed in any previous claim, wherein said discarding step (v) comprises discarding P-N samples.*

Novelty (A3)
For the initial state fig. 7 of A3 discloses a discarding step whereby P-N samples are discarded, namely N'.

Consequently, every feature of claim 9 is known from A3 and lacks novelty.

Inventive step (A3)
For the states following the initial filtering, fig. 7 of A3 discloses a discarding step whereby P-N samples are discarded, namely N'.

Consequently, it is submitted that claim 9 of the opposed patent lacks inventive step over A3.

Novelty (A4)
A4 is applied by means of fast convolution and consequently, the discarding step includes the step of discarding P-N samples.

Consequently, claim 9 lacks novelty.

**Claim 10:**

> *A method as claimed in any previous claim, wherein said predetermined portion of a desired impulse response comprises M times N time-domain values h(k) where $0 \leq k < NM$ and the m-th block ($0 \leq m < M$) of values is made up of the N sample points h(mN) to h(mN+N-1).*

Novelty (A3)
For the initial state, page 17 lines 1-8 of A3 discloses the use of equally sized impulse response blocks, namely consisting of N' samples.

Consequently, every feature of claim 10 is known from A3 and lacks novelty.

Inventive step (A3)
For the states following the initial filtering, page 17 lines 1-8 of A3 discloses the use of equally sized impulse response blocks, namely consisting of N' samples.

Consequently, it is submitted that claim 10 of the opposed patent lacks inventive step over A3.

Novelty (A4)

Page 19, 2nd paragraph, discloses that the impulse response is segmented into 8 equally sized predetermined portions.

Consequently, claim 10 lacks novelty.

**Claim 11:**

> *A method as claimed in claim 1 wherein said method is applied in parallel to a series of predetermined portions of an overall desired impulse response representing a filter characteristic and specified in terms of time-domain response values with the output of each parallel application of the method being combined so as to form an overall output which comprises a filtering of the input signal with the overall desired impulse response.*

Inventive step (A3)
Starting from A3, the skilled person would know from e.g. fig. 7 of A3 that a desired structure of a filter may be transformed into a fast convolution structure as illustrated in fig. 7 having all the features of claim 1 as earlier described.

Faced with the problem of splitting a desired filter into a parallel filter structure on the basis of a time domain impulse response values, the skilled person would look into A3, fig. 2 and 3 and learn that a desired impulse response in the time domain may be represented and regarded separately in parallel structures.

Consequently, it is submitted that claim 11 of the opposed patent lacks inventive step over A3.

Inventive step (A8+ A3)
Starting from A8 the skilled person would know that an impulse response (the already mentioned kernel s(j)) may be split into separate packets serially and dealt with separately in the frequency domain. Faced with the problem of splitting a desired filter into a parallel filter structure on the basis of a time domain impulse response values, he would look into A3, fig. 2 and 3 and learn that such structure may be designed to comprise of different parallel structures, which may be dealt with separately.

Consequently, it is submitted that claim 11 of the opposed patent lacks inventive step over A8 in the light of A3.

Inventive step (A4+A3)
Starting from A4, faced with the problem of choosing a suitable filter topology, the skilled person would look into fig. 3 of A3 and learn that a parallel representation exists in the time domain and that this representation may be represented in the frequency domain.

Consequently, claim 11 lacks inventive step over A4 in the light of A3.

**Claim 12:**

> *A method as claimed in claim 11 wherein said series of predetermined portions are of different lengths.*

Inventive step (A3)
Based on the arguments set forth in relation to claim 11, the skilled person would know that such a parallel filter structure may comprise predetermined portions of different lengths.

Consequently, it is submitted that claim 12 of the opposed patent lacks inventive step over A3,

Inventive step (A8+ A3)
Starting from A8 and based on the arguments set forth in relation to claim 11, the skilled person would know that such a parallel filter structure may comprise predetermined portions of different lengths.

Consequently, it is submitted that claim 12 of the opposed patent lacks inventive step over A8 in the light of A3.

Inventive step (A4+A3)
Starting from A4, faced with the problem of choosing a suitable filter topology, the skilled person would look into fig. 3 of A3 and learn that a parallel representation exists in the time domain and easily derive that the predetermined parallel may have different length.

Consequently, claim 12 lacks inventive step over A4 in the light of A3.

**Claim 13:**

> *A method as claimed in claim 12 wherein initial members of said series of predetermined portions are shorter then subsequent members of said series..*

Inventive step (A3)
Based on the arguments set forth in relation to claim 11, the skilled person would know that such a parallel filter structure may comprise predetermined portions of different lengths and that the provided parallel filters does not need to produce a response before the low-latency initial impulse response has been delivered.

This is in particular realised in A3 fig. 3, where a delay has been inserted into the second path in order to synchronise the provided impulse response outputs.

Consequently, it is submitted that claim 13 of the opposed patent lacks inventive step over A3.

Inventive step (A4+A3)
Starting from A4, faced with the problem of choosing a suitable filter topology, the skilled person would look into fig. 3 of A3 and learn that a parallel representation exists in the time domain and easily derive that the predetermined parallel may have different length, where the initial members have shorter length that the subsequent members..

Consequently, claim 13 lacks inventive step over A4 in the light of A3.

**Claim 14:**

> *A method as claimed in claim 11 wherein the latency of application of said method is varied so that each parallel application of the method produces an output for combination substantially simultaneously.*

Inventive step (A3)
Based on the arguments set forth in relation to claim 11, the skilled person would know that a combination of the different paths would advantageously be performed so that the output of each parallel string is substantially provided for combination at the same time.

Consequently, it is submitted that claim 14 of the opposed patent lacks inventive step over A3.

Inventive step (A4+A3)
Starting from A4, faced with the problem of choosing a suitable filter topology, the skilled person would look into fig. 3 of A3 and learn that a parallel representation exists in the time domain and easily derive that the output of the parallel filter structures should be synchronised in order to obtain the desired combination with the lowest latency possible.

Consequently, claim 14 lacks inventive step over A4 in the light of A3.

**Further prior public disclosure**

It is further submitted that the claims 1-14 of the opposed patent lack novelty and/or inventive step since the invention was made available to the public by demonstration and/or oral disclosure before the date of priority.

As stated in the testimony A12, a meeting took place at Bang & Olufsen, Struer, Denmark in the autumn of 1989, i.e. well in advance of the earliest priority date of the opposed patent, where the invention was formulated by Erik Sørensen amongst a group of persons, i.e., Peter Single, Henrik Fløe Mikkelsen and Knud Bank Christensen, of which none were bound by any secrecy obligations in this relation. Thus, this disclosure constitutes prior art in accordance with article 54(2) EPC, cf. e.g. decision T 11/99 of the Board of Appeal.

We reserve the right to produce further information, documentation and evidence concerning this prior public disclosure, if it is found necessary and/or expedient.

To further substantiate the circumstances and information disclosed at this meeting it is requested that witnesses are heard in accordance with Article 117 (1d), cf. Article 117 (3) EPC, in particular

Mr. Knud Bank Christensen
Research Engineer of TC Electronic A/S
Skovvej 2

DK-8550 Ryomgaard
Denmark

Mr. Peter Single

Mr. Erik Sørensen

and

Mr. Henrik Fløe Mikkelsen

**Additional Remarks**

On the basis of the above facts and arguments, the patent opposed should be revoked. In the event that the Opposition Division needs further evidence we would like to be notified thereof.

If, against the expectation of the opponent, the Opposition Division should be inclined to maintain the opposed patent in amended or unamended form oral proceedings under Article 116 EPC are hereby respectfully requested.

Århus, 16 February 2004

Kaj Olesen
European Patent Attorney

PATENTGRUPPEN

European Patent Office
D – 80298 München
Deutschland

PATENTGRUPPEN ApS
Arosgården
Aaboulevarden 31
DK-8000 Aarhus C

tel.: +45 86 19 20 00
fax: +45 86 19 91 91

PATENTGRUPPEN ApS
Toldbodgade 31
DK-1253 Copenhagen K

tel.: +45 33 11 34 00
fax: +45 33 11 34 35

www.patentgruppen.dk
patent@patentgruppen.dk

Fax: +49 89 2399-4465 – 25 pages including this page
Confirmation copy by mail

Aarhus, 16 February 2004

Your ref.: 0649578
Our ref.: U 03 021 DK JKM/GLR

**Opposition against European Patent Application No. 0 649 578 B1**
**Lake Technology Limited**

We hereby file opposition against European Patent No. 0 649 578 B1, cf. enclosed form 2300 and Grounds for Opposition as well as annex A1-A12, which were sent to you earlier today.

Yours truly,
**PATENTGRUPPEN**

Kaj Olesen

PATENTGRUPPEN

PATENTGRUPPEN ApS
Arosgården
Aaboulevarden 31
DK-8000 Aarhus C

tel.: +45 86 19 20 00
fax: +45 86 19 91 91

PATENTGRUPPEN ApS
Toldbodgade 31
DK-1253 Copenhagen K

tel.: +45 33 11 34 00
fax: +45 33 11 34 35

www.patentgruppen.dk
patent@patentgruppen.dk

European Patent Office
D – 80298 München
Deutschland

Fax: +49 89 2399-4465 – 214 pages including this page
Confirmation copy by ~~mail~~ PART I → pages 1 - 50

Aarhus, 16 February 2004

Your ref.: 0 649 578
Our ref.: U 03 021 DK JKM/GLR

**Opposition against European Patent No. 649 578 B1**
**Lake Technology Limited**

In relation to opposition filed today against European Patent No. 649 578 B1 (Digital Filter Having High Accuracy and Efficiency) wherein

TC Electronic A/S
Sindalsvej 34
DK-8240 Risskov
Denmark

is mentioned as opponent,

please find enclosed the following publications that will serve as evidence and which will be cited in "Facts and Arguments" of the opposition:

A1 EP 0 649 578 (The opposed patent as published)

A2 WO 94/01933 A (The international application as filed)

A3 WO 88/03341 A (published on 5 May 1988)

A4 Digital Equalization Using Fourier Transform Techniques, 2694 (B-2) by Barry Kulp (published on 3 November 1988)

A5 EP 0 250 048 A (published on 23 December 1987)

A6 US 4 992 967 A (published on 12 February 1991)

A7 Multidelay Block Frequency Domain Adaptive Filter by Jia-Sien Soo and Khee K. Pang (published on 2 February 1990)

A8 High-speed convolution and correlation by Thomas G. Stockham Jr. (published in 1966)

A9 Real Signals Fast Fourier Transform: Storage Capacity and Step Number Reduction by Means of an Odd Discrete Fourier Transform by J. L. Vernet (published in October 1971)

A10 Odd-Time Odd-Frequency Discrete Fourier Transform for Symmetric Real-Valued Series by G. Bonnerot and M. Bellanger (published in March 1976)

A11 Digital Processing of Signals – Theory and practice by Maurice Bellanger (published in 1984 and 1989 and reprinted in April 1990)

A12 Testimony of prior disclosure by Knud Bank Christensen

Form 2300 with enclosures will follow.

Yours truly,
**PATENTGRUPPEN**

Kaj Olesen

A I

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 649 578 B1

(12) EUROPEAN PATENT SPECIFICATION

(45) Date of publication and mention of the grant of the patent:
14.05.2003 Bulletin 2003/20

(21) Application number: 93914555.3

(22) Date of filing: 05.07.1993

(51) Int Cl.7: H03H 17/06, H03H 17/02, G06F 15/00

(86) International application number:
PCT/AU93/00330

(87) International publication number:
WO 94/001933 (20.01.1994 Gazette 1994/03)

# (54) DIGITAL FILTER HAVING HIGH ACCURACY AND EFFICIENCY
DIGITALES FILTER MIT HOHER GENAUIGKEIT UND EFFIZIENZ
FILTRE NUMERIQUE A HAUTE PRECISION ET A HAUT RENDEMENT

(84) Designated Contracting States:
BE DE DK FR GB IT NL SE

(30) Priority: 07.07.1992 AU PL338692
07.07.1992 AU PL338792
20.04.1993 AU PL835793

(43) Date of publication of application:
26.04.1995 Bulletin 1995/17

(60) Divisional application:
03001549.9 / 1 304 797

(73) Proprietor: LAKE TECHNOLOGY LIMITED
Maroubra, NSW 2035 (AU)

(72) Inventor: McGRATH, David, Stanley
Randwick, NSW 2031 (AU)

(74) Representative: Nicholls, Michael John
J.A. KEMP & CO.
14, South Square
Gray's Inn
London WC1R 5JJ (GB)

(56) References cited:
EP-A- 0 250 048
WO-A-88/03341
US-A- 4 992 967
EP-A- 0 448 758
US-A- 4 623 980

- Proceedings of the IEEE, Vol. 75, No. 9, Issued September 1987 (New York), R.C. AGARWAL, "Vectorized Mixed Radix Discrete Fourier Transform Algorithms", pages 1283-1292, whole document.

EP 0 649 578 B1

Note: Within nine months from the publication of the mention of the grant of the European patent, any person may give notice to the European Patent Office of opposition to the European patent granted. Notice of opposition shall be filed in a written reasoned statement. It shall not be deemed to have been filed until the opposition fee has been paid. (Art. 99(1) European Patent Convention).

Printed by Jouve, 75001 PARIS (FR)

## Description

### FIELD OF THE INVENTION

[0001] The present invention relates to the art of electronic signal processing, and more particularly but not exclusively, to an electronic filtering environment wherein relatively high accuracy and efficiency is desired and a relatively short flow-through delay (termed "latency") is desired.

### DESCRIPTION OF THE PRIOR ART

[0002] With reference to Fig. 1, electronic filters are utilised to modify the characteristics of an incoming electronic signal so as to provide an output signal which is modified in some defined fashion. In the case of Fig. 1 a "notch" filter is illustrated wherein, in the frequency domain, frequencies in the spectrum of the incoming signal S1 are attenuated in the $F_1$ to $F_2$ band so as to produce output signal S2.

[0003] Such filters can be implemented from entirely analog components although, in more recent times, there is a preference, in many circumstances, to implement the filter in a digital fashion. Digital implementation can be by means of dedicated digital circuitry or by means of computers (micro processors) programmed to operate as a filter.

[0004] Filters have many applications in the field of electronic modelling of real world conditions. For example, filters can be used to provide a model of the acoustic characteristics of rooms or halls. Filters are also used to model deficiencies in systems so as to apply appropriate correction factors for the purpose of removing (cancelling) imperfections in signals caused by the deficiencies.

[0005] Frequently it is desirable that such processing take place in "real time". Also, it is desirable that there is effectively no delay in filtering of a signal generated in a real/live environment so that the modelling/correcting steps performed by the filter are, to all intents and purposes, without any delay being perceptible to the end user.

[0006] To achieve this the delay introduced by the filter F while it performs its filtering function must be reduced to a negligible figure. That is, the time when signal S1 first presents to filter F and the time when the results of the filtering by filter F of the first incoming portion of signal S1 become available at the output of the filter S2 must be almost the same. The delay between these two events is hereinafter referred to as the "latency" of the filter system.

[0007] Where the filter F is implemented in a digital manner it may first be necessary to sample the incoming signal S1 (via an analog to digital converter) then perform the filtering function and then convert the digital signal back to an analog signal (by means of a digital to analog converter). The sampling process takes samples of the incoming signal at discrete time intervals $t_i$. The time between each sample is usually the same.

[0008] The sampling processing itself introduces finite delays into the system. Additionally, where the filter is implemented by one of the popular fast convolution techniques there is a delay introduced which, in very broad terms, is proportional to the accuracy (or length) of the filter.

[0009] Mathematically, the filtering operation (that is, the step of imposing the filter characteristic upon the incoming signal S1 so as to produce outgoing signal S2) is known as "convolution" in the time domain. The step of convolution in the time domain becomes a multiply operation in the frequency domain. That is, if the incoming signal S1 is first sampled, then Fourier transformed into the frequency domain, the frequency response of the filter F is vector multiplied with the Fourier transform of the signal S1. The signal is then inverse Fourier transformed to produce a sampled (convolved) output (which can be converted back to analog form if required).

[0010] Figure 2 shows the way a convolver (also known as a Finite Impulse Response (FIR) filter) has its impulse response $\{a_k\}$ measured (for a convolver operating on a treatment of sampled data). For a physical filter, $a_k$ is zero for all $k<0$. For a general input sequence $\{x_k\}$, the filter's output $\{Y_k\}$ is defined as:

$$Y_k=\sum_{i=0}^{\infty} a_i\, x_{k-i} \qquad (1)$$

[0011] A linear filter such as this has a measurable latency, d, defined as:

$a_d \neq 0$, and

$$a_k = 0 \text{ for all } k < d \tag{2}$$

[0012] In other words, $a_d$ is the first non-zero value in the output sequence. The latency d is never negative in a physical system. In a similar fashion, for a Finite Impulse Response Filter, we can determine which is the LAST non-zero value in the output sequence. This will give us the length of the impulse response. If we call the length $l$, then this means that $a_{d+l-1}$ is the last non-zero value in the output sequence (see Figure 3).

[0013] Typical schemes for implementing FIR filters fall into two categories:

1. Time domain filters that compute each output sample as each new input sample arrives, thus allowing latencies as low as d=0 or d=1. Typical filter lengths ($l$) are short.
2. Fast convolution filters that compute a number of output samples in a block. Typical filter lengths (l) can be very long. The lowest achievable latency is usually related to the filter length, $d \approx l + K$ or

$$K \approx l + d \tag{3}$$

where K is a measure of the efficiency of the particular algorithm used. A typical value of K, for the commonly used fast convolution algorithms such as illustrated in Figs. 4 and 5, is 0.5.

[0014] WO-88 03341 discloses an echo-canceller in which an input signal is divided into blocks and, on the decreased number of samples in each block, a fast Fourier transform and FIR type digital filtering are effected to decrease the processing delay while reducing the number of calculations. In particular, the echo-cancellor of WO-88/03341 implements a method of infinite impulse response (FIR) filtering an input signal to produce a filtered output signal which is an echo-cancelling signal. The method uses the error between an echo-signal derived from the input signal passed through an echo path and the filtered output signal to adaptively control a filter characteristic (specified in terms of time-domain impulse response values) which is an estimate of the impulse response of an echo path. The method comprises the steps of:

(a) dividing the input signal into overlapping successive input blocks of 2N samples, with each successive input block being delayed by N samples relative to the previous input block;

(b) updating each of k frequency-domain coefficient blocks using the error by:

(i) taking N values of the error;

(ii) adding N zeros to said N values of the error to form a block of 2N zero-padded values; and

(iii) computing a frequency-domain transformation of said block of 2N zero-padded values to form 2N updating error coefficients; and

(iv) using the updating error coefficients to update a corresponding one of said k frequency-domain coefficient blocks; and

(c) for each of said input blocks:

(i) computing a frequency-domain transform of said input block to form a corresponding frequency-domain input block;

(ii) combining together the most recent k successive said frequency-domain input blocks with k frequency-domain coefficient blocks, to produce k frequency-domain filtered blocks;

(iii) summing together said k frequency-domain filtered blocks to form a single frequency-domain output block;

(iv) computing an inverse transform, which is the inverse of said frequency-domain transform, of said frequency-domain output block to form a time-domain output block;

(v) discarding the first N samples of said time-domain output block, to produce a new set of N output samples; and

(vi) outputting said N output samples as a portion of said filtered signal.

[0015] It is an object of at least a preferred embodiment of the present invention to provide a method and apparatus for performing relatively long convolutions on digital sampled data so as to provide relatively higher efficiency for a given length than is ordinarily produced with other methods.

[0016] In this specification it is assumed that the filter characteristics can be modelled as approximately linear so that the principles of superposition can be applied.

[0017] Accordingly, according to the present invention there is provided a method of finite impulse response (FIR) filtering an input signal using a predetermined portion of a desired time domain impulse response representing a filter characteristic and specified in terms of time-domain impulse response values, so as to produce a filtered output signal, the method comprising the steps of:

(a) dividing said input signal into overlapping successive input blocks of P samples, with each successive input block being delayed by N samples relative to the previous input block, where P≥2N-1;

(b) creating each of substantially M frequency-domain coefficient blocks by:

(i) dividing said predetermined portion of the desired time domain impulse into a series of segments;

(ii) computing a frequency-domain transformation of each segment to form a corresponding one of said M frequency-domain coefficient blocks; and

(c) for each of said input blocks:

(i) computing a frequency-domain transform of said input block to form a corresponding frequency-domain input block;

(ii) combining together the most recent M successive said frequency-domain input blocks with M frequency-domain coefficient blocks, to produce M frequency-domain filtered blocks;

(iii) summing together said M frequency-domain filtered blocks to form a single frequency-domain output block;

(iv) computing an inverse transform, which is the inverse of said frequency-domain transform, of said frequency-domain output block to form a time-domain output block;

(v) discarding predetermined portions of said time-domain output block, to produce a new set of N output samples; and

(vi) outputting said N output samples as a portion of said output signal.

[0018] Embodiments of the invention will now be described with reference to the accompanying drawings wherein:

| | |
|---|---|
| Fig. 1 | is a generalised block diagram of a filter operation in the frequency domain, |
| Fig. 2 | defines the basic terminology used for a convolution filter, |
| Fig. 3 | defines the latency and length of the filter of Fig. 2, |
| Fig. 4 | illustrates in a diagrammatic flow chart form, a prior art method of processing sampled data by a Fast Fourier Transform approach, |
| Fig. 5 | further illustrates the approach of Fig. 4, |
| Fig. 6,7,8 | develop a method of filtering according to a generalised first embodiment of the invention whereby a relatively high efficiency factor K can be achieved as compared with the approach of Fig. 4, |
| Fig. 9 | is a diagram of an embodiment of the invention implementing the method of Fig. 8 where the number of sub-filters is 6, |
| Fig. 10 | illustrates in block diagram form a summed filter a part of which can be implemented advantageously with the filter of Fig. 8, |
| Fig. 11 | is a block diagram of the summed filter of Fig. 10 incorporating sub-filters some of which implement the method of Fig. 8, |
| Fig. 12 | illustrates the manner of processing of an input signal by an example of the filter arrangement of Fig. 10 which utilised five filter portions, |
| Fig. 13 | illustrates the manner of selection of the filter characteristics of the filter of Fig. 12, |
| Fig. 14 | is a bloc diagram of an alternative implementation of the summed filter of Fig. 8, |
| Fig. 15 | illustrates a typical flow of a (prior art) fast convolution algorithm implementation suitable for filters $F_2$ - $F_5$ of Fig. 12, |
| Fig. 16 | illustrates a DFT engine which forms the basis for an explanation of a fourier transform algorithm optimised to process real number strings, and |
| Fig. 17 | is a block diagram of a further embodiment of the invention wherein the summed filter of Fig. 10 is implemented utilizing a Modified Discrete Fourier Transform. |

## DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS OF THE INVENTION

### 1. High Efficiency Filter

[0019] Figure 4 illustrates the time-flow of a typical fast-convolution algorithm. This is an overlap-discard algorithm implemented using the Fast Fourier Transform (FFT). 2M words of input data that arrives during time segments a and b is processed fully during time segment c with a forward FFT, a vector multiply, and an inverse FFT. The resulting M words of output data are buffered, to be presented at the output during time segment d. The FFT and inverse FFT (IFFT) are *only used* to transform the data between the time-domain and frequency domain. The actual filter operation is executed in the vector multiply operation, which actually takes only a small fraction of the total compute time. So, the relevant parameters of the filter of Fig. 4 are:- Length = M,

Latency = 2M,

and, therefore K = 0.5

[0020] With reference to Figs. 6, 7 and 8, the rationale behind the method and apparatus according to at least one embodiment of the present invention is derived.

[0021] Fig. 6 illustrates a filter of length ML where the filter characteristics of each of the component filters F1, F2. .... $F_n$ are separate, discrete, component portions of the desired filter characteristic for the entire filter assembly. The delays L, 2L.... (M-1) L are imposed as illustrated so as to recreate, following addition, an output $y_k$ equivalent to that achieved by passing input samples $x_k$ through a filter having the filter characteristic from which the filter characteristic portions for filters F1, F2..... were derived. Fig. 7 is derived by implementing the filters F1, F2.... of Fig. 6 using the Fast Fourier Transform algorithm of Fig. 5.

[0022] With reference to Fig. 8, reorganisation of the filter of Fig. 7 allows the use of only one Fast Fourier Transform module 11 and one Inverse Past Fourier Transform module 12. It is implicit that the Fast Fourier Transform module is adapted to process a block of samples from input $x_k$ equal to twice the length of each of the filters Filter 1, Filter 2. Filter 3..... illustrated in Fig. 8.

[0023] As previously stated the filter characteristic (impulse response) of each consecutive filter F1, F2.... is taken from and corresponds to consecutive corresponding portions from the impulse response desired of the entire filter module.

[0024] The time delay L before each Fast Fourier transformed block of data is passed through the next filter is equal to half the sample length originally processed by the Fast Fourier Transform module.

[0025] Figure 9 shows the computation of one block of output data. In a similar style to Figure 4, but using the improved length/latency efficiency method derived in Figs. 6, 7 and 8. The method of Fig. 8 as used by Fig. 9 is summarised below.

[0026] During time segment h. the input data that arrived during time segments f and g is FFT'd and the resulting block of Frequency Domain Input Data is stored for future use. We then compute the next block of Frequency Output Data, which is inverse FFT'd and presented as output during time segment i. The old way of computing fast-convolution simply took the latest block of Frequency Domain Input Data, and multiplied it by a vector that represents the desired filter response, to get the new Frequency Domain Output Data. The improved length/latency efficiency method uses a number of previous Frequency Domain Input Data blocks to compute the new Frequency Domain Output Data block, as shown in Figure 9. In this example, the blocks of filter data are called Filter A, Filter B, ..., Filter F. In this example, the filter implemented is 6 times as long as the filter implemented in Figure 4, but with no greater latency. By comparison with Fig. 4, the relevant parameters of the filter of Fig. 9 are:- Length = 6M,

Latency = 2M,

and, therefore K = 3

[0027] Fig. 8 summarises the logic behind the implementation of the embodiment of Fig. 9.

[0028] Particularly, it will be noted that the progressive delays L, 2L, 3L, ...(M-1)L of Fig. 5 are achieved in Fig. 9 by the taking of delayed, overlapped groupings of consecutive samples a, b, c, d, ...

[0029] The above described filter arrangement can be used advantageously in a low-latency FIR filter arrangement such as illustrated in Fig. 10.

[0030] Figure 10 shows an architecture for implementing an FIR filter by adding together N filters. If each filter is characterised as: Filter $F_i$, latency $d_i$, length $l_i$, then generally the N filters are chosen so that their latencies are ordered in ascending order, and furthermore $d_{i+1}=d_i+l_i$. This means that the first non-zero value in the impulse response of filter $F_{i+1}$, comes immediately following the last non-zero value in the impulse response of filter $F_i$. Hence this summation of filters results in a single, longer filter with its impulse response being the sum of the impulse responses of the N component filters.

[0031] The important property of this filter is the length/latency efficiency, K, which is higher than any of the component filter efficiencies.

[0032] That is, the filter of Fig. 10 uses the technique of adding together several filters to form a new filter which is as long as the sum of the component filter lengths, and whose latency is as short as the latency of the lowest-latency component filter.

**[0033]** Fig. 11 shows an implementation of the low-latency filter 10 of Fig. 10 wherein there are three filter modules F1, F2, F3. The first module F1 is a low-latency (d=O) time domain filter whilst filters F2 and F3 are implemented according to the embodiment described in respect of Figs. 8 and 3.

## 2. A low-latency FIR filter

**[0034]** As previously described, Fig. 10 shows an architecture for implementing an FIR filter by adding together N filters. If each filter is characterised as: Filter $F_i$, latency $d_i$, length $l_i$, then generally the N filters are chosen so that their latencies are ordered in ascending order, and furthermore $d_{i+1}=d_i+l_i$. This means that the first non-zero value in the impulse response of filter $F_{i+1}$, comes immediately following the last non-zero value in the impulse response of filter $F_i$. Hence this summation of filters results in a single, longer filter with its impulse response being the sum of the impulse responses of the N component filters.

**[0035]** The important property of this filter is the length/latency efficiency, K, which is higher than any of the component filter efficiencies.

**[0036]** That is, the filter of Figs. 10 and 12 uses the technique of adding together several filters to form a new filter which is as long as the sum of the component filter lengths, and whose latency is as short as the latency of the lowest-latency component filter.

**[0037]** Particularly, the composite filter assembly of Fig. 12 utilises the technique of combining a first time-domain (low latency) filter with additional fast-convolution (longer latency) filters to maximise filter length while minimising latency. This technique is implemented by adding together N filters, $F_1$, $F_2$, ...$F_N$ where $F_1$ is a filter with very low latency, implemented with time-domain techniques, and the other filters, $F_i$, are each implemented with fast-convolution techniques. More specifically, the assembly adopts the technique whereby the N-1 fast-convolution filters, Fi, are composed of a sequence of filters, each with longer filter length than its predecessor, and hence each with longer latency, but still preserving the property that $d_{i+1}=d_i+1_i$. This ensures that the filter, F, which is made by summing together the N component filters, has an impulse response without any "holes" in it.

**[0038]** With particular reference to Fig. 12 the composite filter F comprises five filter portions F1, F2, F3, F4 and F5. The impulse response $a_k$ of the composite filter F is illustrated at the top of Fig. 12 and has a total sample length of 1024 samples.

**[0039]** Filter F1 has an impulse response comprising the first 64 samples of the impulse response $a_k$. That is, the filter has a length of 64 samples. The filter as implemented has a low latency filter (such as is referenced in Motorola document APR 7/D in respect of the DSP 56000 series of integrated Circuits). This filter has an effective latency of 0.

**[0040]** The subsequent filters F2, F3, F4, F5 are implemented using fast convolution digital techniques. Fig. 15 illustrates the basic algorithm for such techniques which comprises taking the fast Fourier transform of the incoming sampled data, multiplying the transformed data samples by the impulse response of the filter, converting the fast Fourier transformed data samples back to the time domain by use of an inverse fast Fourier transform and then outputting the data. An overlap/discard method is used whereby only a portion of the output data is utilised.

**[0041]** The length and latency of the additional filters F2, F3, F4, F5 is selected according to the rule illustrated diagrammatically in Fig. 13, whereby each filter portion has a latency equal to the sum of the length and the latency of the immediately preceding filter portion.

**[0042]** In this case the end result is a filter having a total length of 1024 samples and a latency of 0.

**[0043]** Fig. 14 illustrates a variation of the filter of Fig. 8 wherein delay is introduced after the filter algorithm is applied in the frequency domain.

## 3. Optimized Real String Handling

**[0044]** With reference to Fig. 16, a common method of frequency analysis is via the Discrete Fourier Transform (hereafter referred to as the DFT), which can be implemented efficiently in electronic apparatus using the Fast Fourier Transform algorithm (hereafter referred to as the FFT).

**[0045]** The DFT is formulated to operate efficiently when its input data and output data are both complex (having a real and imaginary component). When the data input to the DFT is real, the output data from the operation will contain some redundancy, indicating that some of the processing that led to this output data was unnecessary.

**[0046]** In this embodiment what is described is a new transform for operating on real numbers in the digital environment, that has many of the same applications as the DFT, but without the inefficiencies of the DFT for operation on real numbers. For the purposes of this document, the algorithm described herein will be named the Modified Discrete Fourier Transform (MDFT).

**[0047]** The DFT is computed according to the equation below:

$$X(n)=\sum_{k=0}^{N-1}x(k)e^{\frac{-2\pi jnk}{N}} \qquad (1)$$

[0048] If the input data x(k) is real (ie. it has no imaginary component), the output data X(n) has the following properties:

$$\begin{aligned}&X(0)\in\Re\\&X(\tfrac{N}{2})\in\Re \qquad (2)\\&X(n)=[X(N-n)]^{*}\ \mathit{for}\ 0<n<\tfrac{N}{2}\end{aligned}$$

[0049] Where the * operator is used to signify complex conjugation. This means that the imaginary part of X(O), the imaginary part of X(N/2) and all {X(n):H/2<n<N} are redundant. The process of extracting only the necessary information out of the DFT output is therefore not trivial.

[0050] An alternative transform is shown below:

$$Y(n)=\sum_{k=0}^{N-1}x(k)e^{\frac{-2\pi jk(n-\frac{1}{2})}{N}}. \qquad (3)$$

[0051] This is like a standard DFT except that the output vector Y(n) represents the signal's frequency components at different frequencies to the DFT. The output vector Y(n) has redundancies (just as the DFT output X(n) has), except that the redundant part of the data is more clearly extracted from Y(n) that from X(n). The redundancy in Y(n) that results when x(k) is real can be expressed as follows:

$$Y(n)=[Y(N\text{-}1\text{-}n)]^{*} \qquad (4)$$

[0052] This implies that the second half of this vector Y(n) is simply the complex conjugate of the first half, so that only the first half of the output vector is required to contain all of the information, when x(k) is real.

[0053] An alternative view of the above equation is that all of the odd elements of the vector are simply the complex conjugate of the even elements:

$$Y(1)=[Y(N\text{-}2)]^{*}\ Y(3)=[Y(N\text{-}4)]^{*}\ Y(N\text{-}3)=[Y(2)]^{*}$$

$$Y(N\text{-}1)=[Y(0)]^{*} \qquad (5)$$

[0054] This means that we only need to compute the even elements of Y(n) to obtain all of the required information from the modified DFT of the real signal x(k). We can name the array Z(p) the array that contains the even elements from Y(n), as follows:

$$Z(p)=Y(2p)\ \mathit{for}\ 0\le p<\frac{N}{2} \qquad (6)$$

Based on our previous equation for Y(n), we get:

$$Z(p) = Y(2p) = \sum_{k=0}^{N-1} x(k) e^{\frac{-2\pi k j(2p+\frac{1}{2})}{N}} \tag{7}$$

which after some manipulation becomes:

$$Z(p) = \sum_{k=0}^{\frac{N}{2}-1} [x(k) - jx(k+\frac{N}{2})] e^{\frac{-\pi jk}{N}} e^{\frac{-2\pi jpk}{(N/2)}} \tag{8}$$

If we create an N/2 length complex vector from the N length real vector:

$$x'(l) = [x(l) - jx(l+\frac{N}{2})] e^{\frac{-\pi jl}{N}} \text{ for } 0 \le l < N/2 \tag{9}$$

then we can see that:

$$Z(p) = DFT_{(N/2)}[x'(l)] \tag{10}$$

**[0055]** This means that we have computed the vector Z(p) by using a DFT of length N/2.

**[0056]** We say that Z(p)=MDFT[x(k)] (where MDFT indicates the Modified Discrete Fourier Transform operator). The procedure to follow for computing Z(p) is then as follows:

1. Take the input vector x(k) of length N, where each element of x(k) is real.
2. Create the vector x'(*l*), a complex vector of length N/2 by the method of equation (9) above.
3. Compute the N/2 point DFT of x'(*l*) to give the N/2 complex result vector Z(p).

**[0057]** The MDFT has many properties that make it useful in similar applications to the DFT. Firstly, it can be used to perform linear convolution in the same way as the DFT. Secondly, it has an inverse transform that looks very similar to the forward transform:

$$x'(k) = IDFT_{(N/2)}[Z(p)] \tag{11}$$

where IDFT indicates the N/2 point Inverse Discrete Fourier Transform.

**[0058]** The algorithm can be implemented in an electronic apparatus supplied with a set of N real numbers and producing N/2 complex output numbers, representing the MDFT of the input data. This apparatus uses digital arithmetic elements to perform each of the arithmetit operations as described in the preceding text.

**[0059]** Another embodiment of the present invention is a pair of apparatus, the first of which computes an MDFT as described in the previous paragraph, and the second of which computes an inverse MDFT, using the arithmetic procedures described previously in this document. 3y passing overlapped blocks of data from a continuous stream of input data through the MDFT computer, then multiplying the Z(p) coefficients by appropriate filter coefficients, then passing the resulting data through the inverse MDFT computer, and recombining segments of output data appropriately, a modified Fast Convolution processor can be built.

**[0060]** The above described a modification to the DFT that makes it more useful in a number of applications particularly but not limited to the real time filter applications previously described. All of these extensions to the DFT can also be applied to the FFT algorithm and other fast implementations of the DFT.

## Example 1

**[0061]** Fig. 17 illustrates an implementation of the summed filter of Fig. 11 wherein the Modified Discrete Fourier

Transform (MDFT) described immediately above is applied for the purposes of transforming the data stream into the frequency domain and the corresponding Inverse Modified Discrete Fourier Transform (IMDFT) is applied following application of the filter algorithm and prior to discard for conversion from the frequency domain.

**[0062]** In filter F2 of Fig. 17; the MDFT takes 64 real words of input and produces 32 complex words of output. The IMDFT takes 32 complex words of input and produces 64 real words of output.

**[0063]** In filter F3 of Fig. 17 the MDFT takes 256 real words of input and produces 128 complex words of output. The IMDFT takes 128 complex words of input and produces 256 real words of output.

**[0064]** The filter of Fig. 17 is implemented using a Motorola DSP 56001 processor incoporating software (bootable from ROM or from another host computer) to implement the algorithm. The delay elements are implemented using a bank of external memory chips comprising three MCM 6206 memory chips.

**[0065]** Data input and output between the analog and digital domains is effected by an ADC and DAC chip, the Crystal CS 4216, communicating via the synchronous serial communication port of the DSP 56001.

## INDUSTRIAL APPLICABILITY

**[0066]** Embodiments of the invention may be applied to digital filters implemented in software, hardware or a combination of both for applications such as audio filtering or electronic modelling of acoustic system characteristics. The method is broadly applicable in the field of signal processing and can be used to advantage, for example, in: adaptive filtering; audio reverberation processing; adaptive echo cancellation; spatial processing; virtual reality audio; correlation. radar; radar pulse compression; deconvolution; seismic analysis; telecommunications; pattern recognition; robotics; 3D acoustic modelling; audio post production (including oralisation, auto reverberant matching); audio equalisation: compression; sonar; ultrasonics; secure communication systems; digital audio broadcast, acoustic analysis: surveillance; noise cancellation; echo cancellation.

## Claims

1. A method of finite impulse response (FIR) filtering an input signal using a predetermined portion of a desired time domain impulse response representing a filter characteristic and specified in terms of time-domain impulse response values, so as to produce a filtered output signal, the method comprising the steps of:

   (a) dividing said input signal into overlapping successive input blocks of P samples, with each successive input block being delayed by N samples relative to the previous input block, where P≥2N-1;

   (b) creating each of substantially M frequency-domain coefficient blocks by:

      (i) dividing said predetermined portion of the desired time domain impulse into a series of segments;

      (ii) computing a frequency-domain transformation of each segment to form a corresponding one of said M frequency-domain coefficient blocks; and

   (c) for each of said input blocks:

      (i) computing a frequency-domain transform of said input block to form a corresponding frequency-domain input block;

      (ii) combining together the most recent M successive said frequency-domain input blocks with M frequency-domain coefficient blocks, to produce M frequency-domain filtered blocks;

      (iii) summing together said M frequency-domain filtered blocks to form a single frequency-domain output block;

      (iv) computing an inverse transform, which is the inverse of said frequency-domain transform, of said frequency-domain output block to form a time-domain output block;

      (v) discarding predetermined portions of said time-domain output block, to produce a new set of N output samples; and

(vi) outputting said N output samples as a portion of said output signal.

2. A method as claimed in claim 1, wherein P is equal to 2N.

3. A method as claimed in any one of the preceding claims wherein at least one of said frequency-domain transformation said frequency-domain transform comprises a 2N-point Real fast Fourier transform producing and N complex values in said frequency-domain coefficient block.

4. A method as claimed in claim 3, wherein said inverse transform comprises an N complex value fast Fourier inverse transform producing 2N-point Real values in said output block.

5. A method as claimed in any previous claim, wherein said step of combining together frequency-domain input blocks comprises element by element multiplication of said frequency-domain input blocks.

6. A method as claimed in any previous claim, wherein said summing together of frequency-domain filtered blocks comprises element by element addition of the blocks.

7. A method as claimed in any previous claim, wherein said combining and said summing are carried out in a single operation with the successive results of the combining operations being accumulated into the frequency-domain output block.

8. A method as claimed in any previous claim, wherein said time-domain output block is of length 0.5 P.

9. A method as claimed in any previous claim, wherein said discarding step (v) comprises discarding P-N samples.

10. A method as claimed in any previous claim, wherein said predetermined portion of a desired impulse response comprises M times N time-domain values h(k) where $0 \leq k < NM$ and the m-th block ($0 \leq m < M$) of values is made up of the N sample points h(mN) to h(mN+N-1).

11. A method as claimed in claim 1 wherein said method is applied in parallel to a series of predetermined portions of an overall desired impulse response representing a filter characteristic and specified in terms of time-domain response values with the outputs of each parallel application of the method being combined so as to form an overall output which comprises a filtering of the input signal with the overall desired impulse response.

12. A method as claimed in claim 11 wherein said series of predetermined portions are of different lengths.

13. A method as claimed in claim 12 wherein initial members of said series of predetermined portions are shorter then subsequent members of said series.

14. A method as claimed in claim 11 wherein the latency of application of said method is varied so that each parallel application of the method produces an output for combination substantially simultaneously.

**Patentansprüche**

1. Verfahren zum Filtern eines Eingangssignals mit finiter Impulsantwort (FIR) mittels eines vorbestimmten Teils einer gewünschten Zeitdomänen-Impulsantwort, die eine Filtercharakteristik darstellt und hinsichtlich Zeitdomänen-Impulsantwortwerten spezifiziert ist, um ein gefiltertes Ausgangssignal zu erzeugen, wobei das Verfahren folgende Schritte umfasst:

   (a) Aufteilen des Eingangssignals in überlappende aufeinanderfolgende Eingangsblöcke von P Abtastungen, wobei jeder aufeinanderfolgende Eingangsblock um N Abtastungen bezüglich des vorhergehenden Eingangsblocks verzögert wird, wobei $P \geq 2N-1$ ist;
   (b) Erzeugen von im wesentlichen M Frequenzdomänen-Koeffizientenblöcken durch:

      (i) Aufteilen des vorbestimmten Teils des gewünschten Zeitdomänen-Impulses in eine Reihe von Segmenten;
      (ii) Berechnen einer Frequenzdomänen-Transformation von jedem Segment, um einen entsprechenden

Block der M Frequenzdomänen-Koeffizientenblöcke zu bilden; und

(c) für jeden Eingangsblock:

(i) Berechnen einer Frequenzdomänen-Transformierten des Eingangsblocks, um einen entsprechenden Frequenzdomänen-Eingangsblock zu bilden;
(ii) Zusammenfassen des letzten M aufeinanderfolgenden Frequenzdomänen-Eingangsblöcke mit M Frequenzdomänen-Koeffizientenblöcken, um M Frequenzdomänengefilterte Blöcke zu erzeugen;
(iii) Aufsummieren der M Frequenzdomänen-gefilterten Blöcke, um einen einzigen Frequenzdomänen-Ausgangsblock zu bilden;
(iv) Berechnen einer Rücktransformierten, die die Umkehrung der Frequenzdomänen-Transformierten des Frequenzdomänen-Ausgangsblocks ist, um einen Zeitdomänen-Ausgangsblock zu bilden;
(v) Verwerfen vorbestimmter Teile des Zeitdomänen-Ausgangsblocks, um einen neuen Satz von N Ausgangsabtastungen zu bilden; und
(vi) Ausgeben der N Ausgangsabtastungen als einen Teil des Ausgangssignals.

2. Verfahren gemäß Anspruch 1, bei dem P gleich 2N ist.

3. Verfahren gemäß einem der vorhergehenden Ansprüche, bei dem mindestens eine der Frequenzdomänen-Transformation und der Frequenzdomänen-Transformierten eine Fast-Fourier-Transformierte mit N reellen Punkten umfasst, die N komplexe Werte in dem Frequenzdomänen-Koeffizientenblock erzeugt.

4. Verfahren gemäß Anspruch 3, bei dem die Rücktransformierte eine Fast-Fourier-Rücktransformierte mit N komplexen Werte umfasst, die 2N Punkte reeller Werte in dem Ausgangsblock erzeugt.

5. Verfahren gemäß einem der vorhergehenden Ansprüche, bei dem der Schritt des Zusammenfassens der Frequenzdomänen-Eingangsblöcke eine elementweise Multiplikation der Frequenzdomänen-Eingangsblöcke umfasst.

6. Verfahren gemäß einem der vorhergehenden Ansprüche, bei dem das Aufsummieren der Frequenzdomänen-gefilterten Blöcke eine elementweise Addition der Blöcke umfasst.

7. Verfahren gemäß einem der vorhergehenden Ansprüche, bei dem das Zusammenfassen und das Aufsummieren in einer einzigen Operation durchgeführt werden, wobei die aufeinander folgenden Ergebnisse der Zusammenfassungsoperationen in dem Frequenzdomänen-Ausgangsblock akkumuliert werden.

8. Verfahren gemäß einem der vorhergehenden Ansprüche, bei dem der Zeitdomänen-Ausgangsblock eine Länge von 0,5 P aufweist.

9. Verfahren gemäß einem der vorhergehenden Ansprüche, bei dem der Verwerfungsschritt (v) ein Verwerfen von P-N Abtastungen umfasst.

10. Verfahren gemäß einem der vorhergehenden Ansprüche, bei dem der vorbestimmte Teil einer gewünschten Impulsantwort M mal N Zeitdomänenwerte h(k) umfasst, wobei 0<k<NM und der m-te Block (0<m<M) von Werten aus den N Abtastpunkten h(mN) bis h(mN+N-1) zusammengesetzt ist.

11. Verfahren gemäß Anspruch 1, bei dem das Verfahren parallel auf eine Reihe vorbestimmter Teile einer gewünschten Gesamtimpulsantwort angewendet wird, die eine Filtercharakteristik darstellt und hinsichtlich Zeitdomänen-Antwortwerten spezifiziert ist, wobei die Ausgaben jeder parallelen Anwendung des Verfahrens zusammengefasst werden, um eine Gesamtausgabe zu bilden, die ein Filtern des Eingangssignals mit der gewünschten Gesamtimpulsantwort umfasst.

12. Verfahren gemäß Anspruch 11, bei dem die Reihe von vorbestimmten Teilen unterschiedliche Längen haben.

13. Verfahren gemäß Anspruch 12, bei dem die Anfangselemente der Reihe von vorbestimmten Teilen kürzer als nachfolgende Elemente der Reihe sind.

14. Verfahren gemäß Anspruch 11, bei dem die Latenzzeit der Anwendung des Verfahrens geändert wird, so dass

jede parallele Anwendung des Verfahrens eine Ausgabe zur im wesentlichen gleichzeitigen Zusammenfassung erzeugt.

## Revendications

1. Procédé de réponse impulsionnelle finie (RIF) filtrant un signal d'entrée utilisant une partie prédéterminée d'une réponse impulsionnelle temporelle désirée représentant une caractéristique de filtrage et spécifiée en fonction de valeurs de réponse impulsionnelle temporelle, afin de produire un signal de sortie filtré, le procédé comprenant les étapes consistant à :

   (a) diviser ledit signal d'entrée en blocs d'entrée successifs superposés de P échantillons, avec chaque bloc d'entrée successif retardé par N échantillons par rapport au bloc d'entrée précédent, où $P \geq 2N-1$ ;
   (b) créer chacun des blocs de coefficient substantiellement dans le domaine de fréquence M :

      (i) en divisant ladite partie prédéterminée de l'impulsion temporelle désirée en une série de segments ;
      (ii) en calculant une transformation dans le domaine de fréquence de chaque segment pour former un bloc correspondant desdits M blocs de coefficient dans le domaine de fréquence ; et

   c) pour chacun desdits blocs d'entrée :

      (i) en calculant une transformée dans le domaine de fréquence dudit bloc d'entrée pour former un bloc d'entrée dans le domaine de fréquence correspondant ;
      (ii) en combinant ensemble les M blocs d'entrée successifs dans ledit domaine de fréquence avec M blocs de coefficient dans le domaine de fréquence, pour produire M blocs filtrés dans le domaine de fréquence ;
      (iii) additionner ensemble lesdits M blocs filtrés dans le domaine de fréquence pour former un seul bloc de sortie dans le domaine de fréquence ;
      (iv) en calculant une transformée inverse, qui est l'inverse de ladite transformée dans le domaine de fréquence, dudit bloc de sortie dans le domaine de fréquence pour former un bloc de sortie dans le domaine temporel ;
      (v) en rejetant des parties prédéterminées dudit bloc de sortie dans le domaine temporel, pour produire un nouvel ensemble de N échantillons de sortie ; et
      (vi) en délivrant lesdits N échantillons de sortie comme une partie dudit signal de sortie.

2. Procédé selon la revendication 1, dans lequel P est égal à 2N.

3. Procédé selon l'une quelconque des revendications précédentes, dans lequel au moins une de ladite transformation dans le domaine de fréquence et de ladite transformée dans le domaine de fréquence comprend une transformée de Fourier rapide réelle de 2N points produisant N valeurs complexes dans ledit bloc de coefficients dans le domaine de fréquence.

4. Procédé selon la revendication 3, dans lequel ladite transformée inverse comprend une transformée inverse de Fourier rapide de N valeurs complexes produisant des valeurs réelles de 2N points dans ledit bloc de sortie.

5. Procédé selon l'une quelconque des revendications précédentes, dans lequel ladite étape consistant à combiner ensemble les blocs d'entrée dans le domaine de fréquence comprend une multiplication élément par élément desdits blocs d'entrée dans le domaine de fréquence.

6. Procédé selon l'une quelconque des revendications précédentes, dans lequel ladite sommation des blocs filtrés dans le domaine de fréquence comprend l'addition élément par élément des blocs.

7. Procédé selon l'une quelconque des revendications précédentes, dans lequel ladite combinaison et ladite sommation sont exécutées dans une simple opération avec les résultats successifs des opérations de combinaison étant cumulées dans le bloc de sortie dans le domaine de fréquence.

8. Procédé selon l'une quelconque des revendications précédentes, dans lequel ledit bloc de sortie dans le domaine temporel a une longueur de 0,5 P.

9. Procédé selon l'une quelconque des revendications précédentes, dans lequel ladite étape de suppression (v) comprend la suppression de P-N échantillons.

10. Procédé selon l'une quelconque des revendications précédentes, dans lequel ladite partie prédéterminée d'une réponse impulsionnelle désirée comprend M fois N valeurs dans le domaine temporel h(k) où $0 \leq k < NM$ et le m$^{ième}$ bloc ($0 \leq m < M$) de valeurs est constitué de N points d'échantillonnage h(mN) à h(mN + N - 1).

11. Procédé selon la revendication 1, dans lequel ledit procédé est appliqué en parallèle à une série de parties prédéterminées d'une réponse impulsionnelle globale désirée représentant une caractéristique de filtre et spécifiée en fonction de valeurs de réponse dans le domaine temporel avec les sorties de chaque application parallèle du procédé étant combinées afin de former une sortie globale qui comprend un filtrage du signal d'entrée avec la réponse impulsionnelle globale désirée.

12. Procédé selon la revendication 11, dans lequel lesdites séries de parties prédéterminées sont de différentes longueurs.

13. Procédé selon la revendication 12, dans lequel les éléments initiaux desdites séries de parties prédéterminées sont plus courts que les éléments suivants desdites séries.

14. Procédé selon la revendication 11, dans lequel le temps d'attente d'application dudit procédé est modifié afin que chaque application parallèle du procédé produise une sortie pour combinaison substantiellement simultanée.

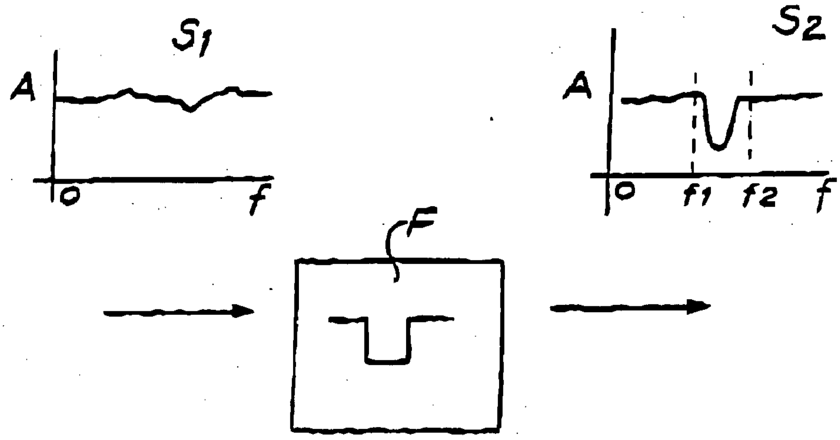


FIG. 1

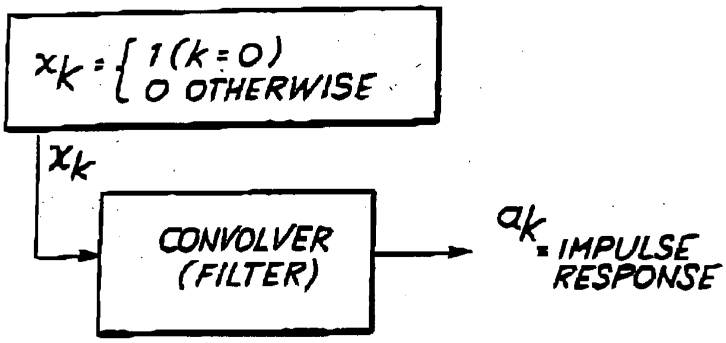

$x_k = \begin{cases} 1 (k=0) \\ 0 \text{ OTHERWISE} \end{cases}$
$x_k$
CONVOLVER (FILTER)
$a_k$ = IMPULSE RESPONSE

FIG. 2

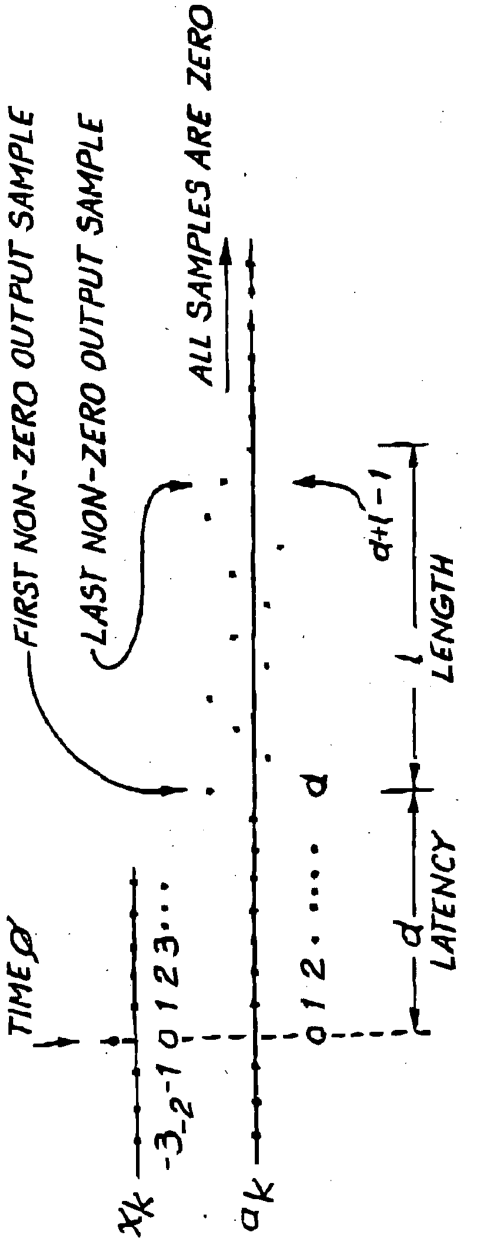

TIME Ø
$x_k$
-3 -2 -1 0 1 2 3 ...
$a_k$
0 1 2 .... d
FIRST NON-ZERO OUTPUT SAMPLE
LAST NON-ZERO OUTPUT SAMPLE
ALL SAMPLES ARE ZERO
d
LATENCY
l
LENGTH
d+l-1

FIG. 3

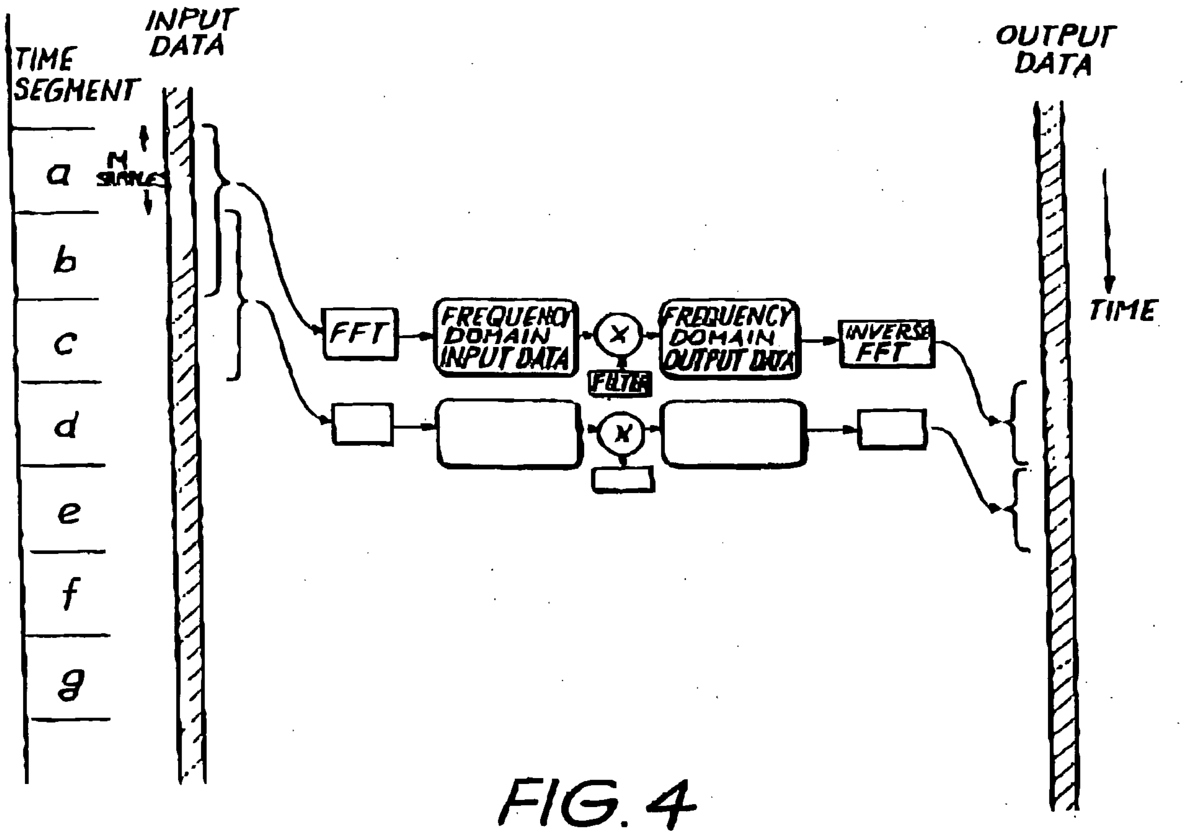

TIME SEGMENT
INPUT DATA
a
b
c
d
e
f
g
M SAMPLES
FFT
FREQUENCY DOMAIN INPUT DATA
X
FILTER
FREQUENCY DOMAIN OUTPUT DATA
INVERSE FFT
X
OUTPUT DATA
TIME

FIG. 4

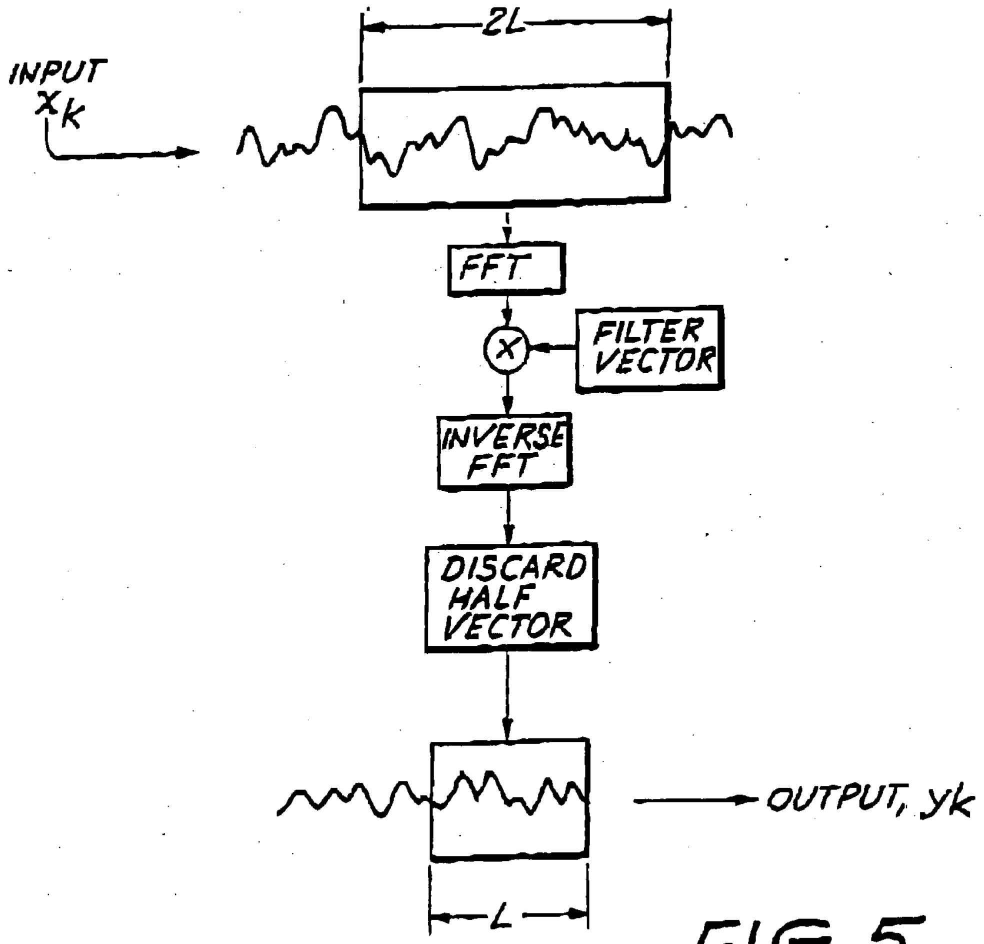

2L
INPUT
$x_k$
FFT
FILTER VECTOR
INVERSE FFT
DISCARD HALF VECTOR
OUTPUT, yk
L

FIG. 5

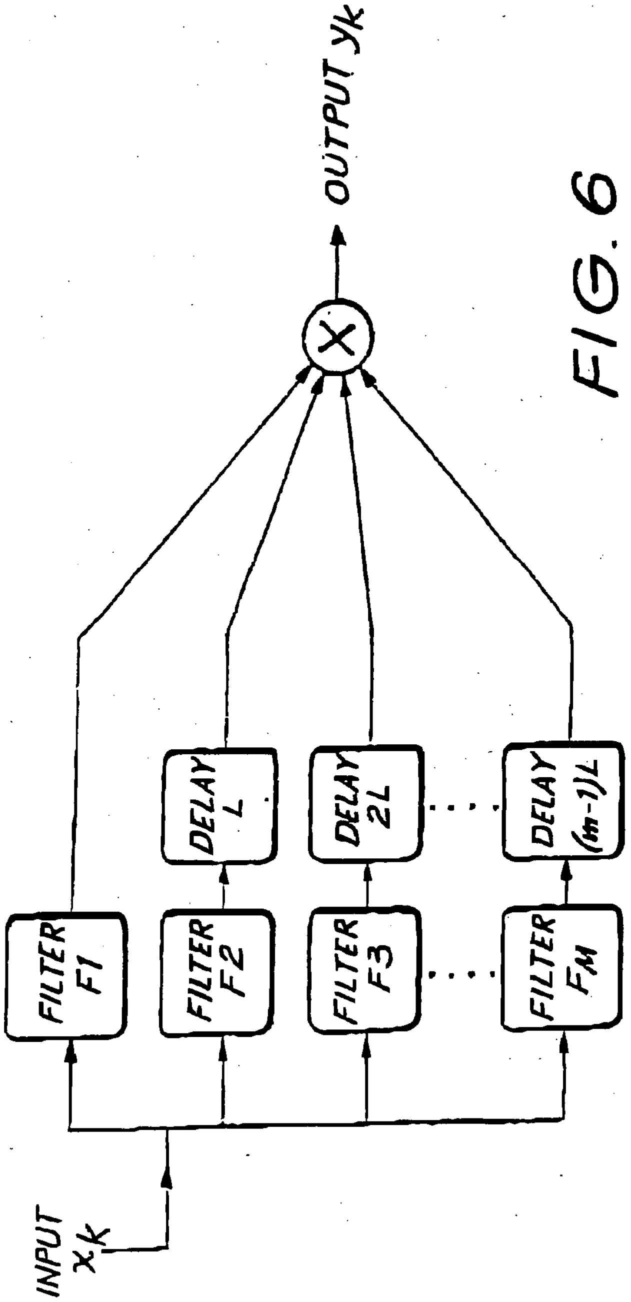

OUTPUT $y_k$
FILTER F1
FILTER F2
FILTER F3
FILTER $F_M$
DELAY L
DELAY 2L
DELAY (m-1)L
INPUT $x_k$

FIG. 6

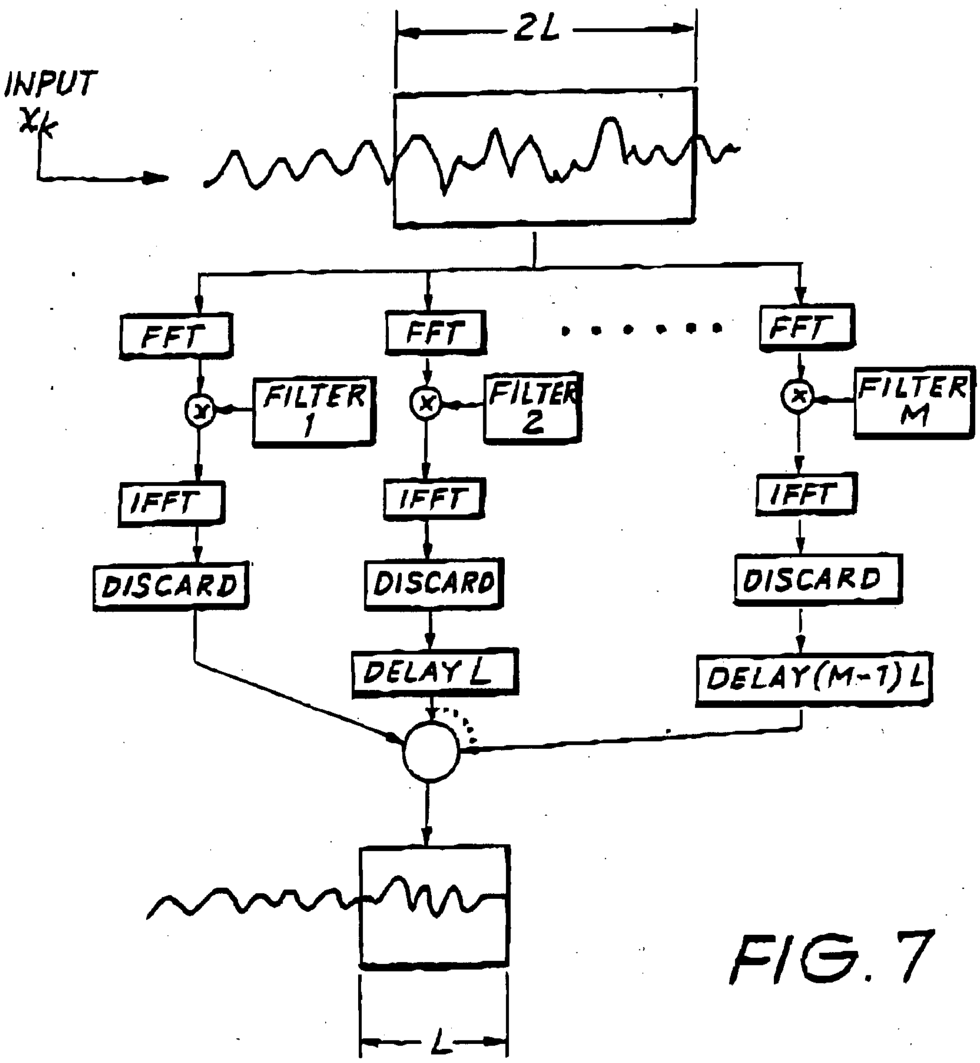

2L
INPUT
$x_k$
FFT
FFT
FFT
FILTER 1
FILTER 2
FILTER M
IFFT
IFFT
IFFT
DISCARD
DISCARD
DISCARD
DELAY L
DELAY (M-1) L
L

FIG. 7

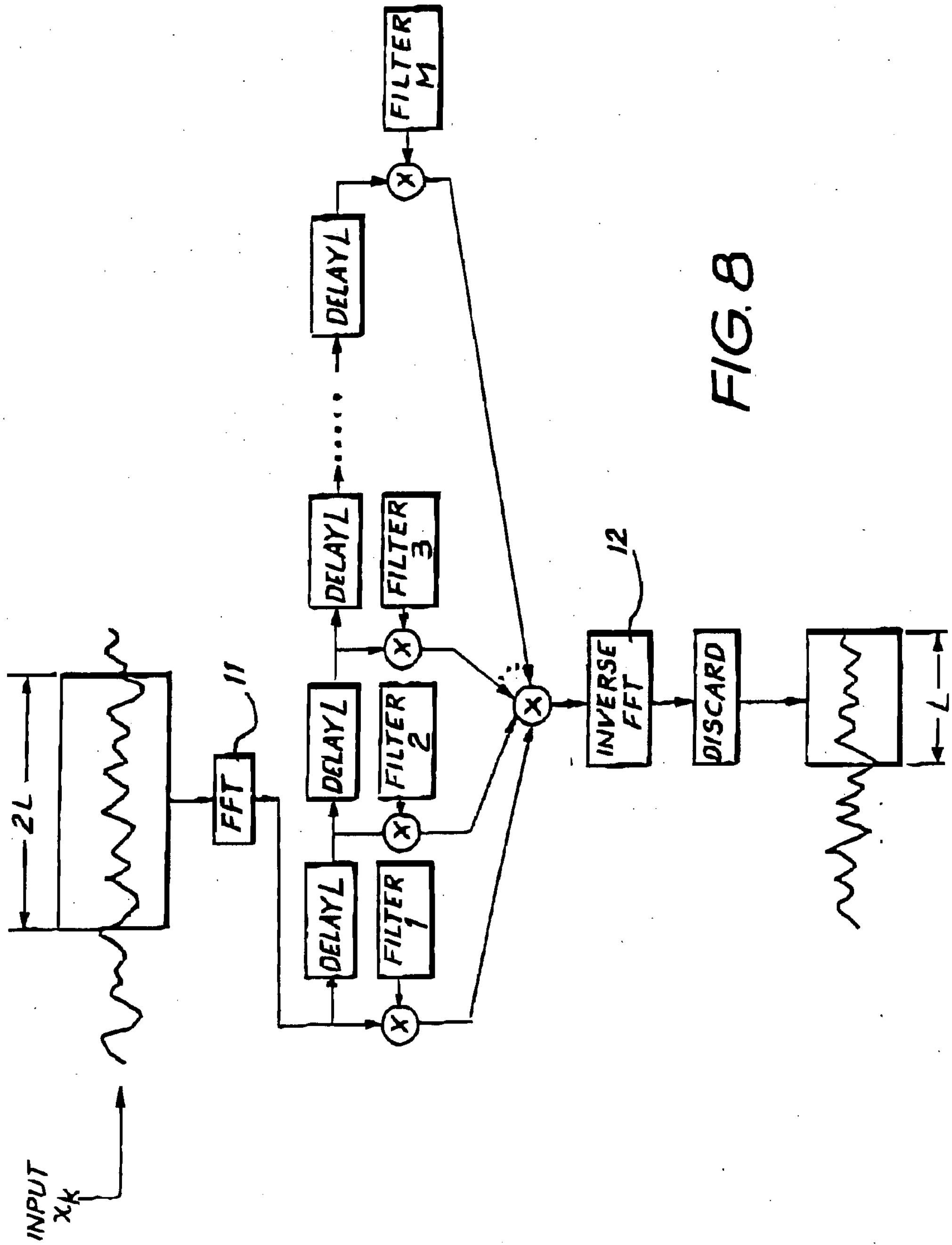

INPUT
$x_k$
2L
FFT
11
DELAY L
FILTER 1
DELAY L
FILTER 2
DELAY L
FILTER 3
DELAY L
FILTER M
12
INVERSE FFT
DISCARD
L

FIG. 8

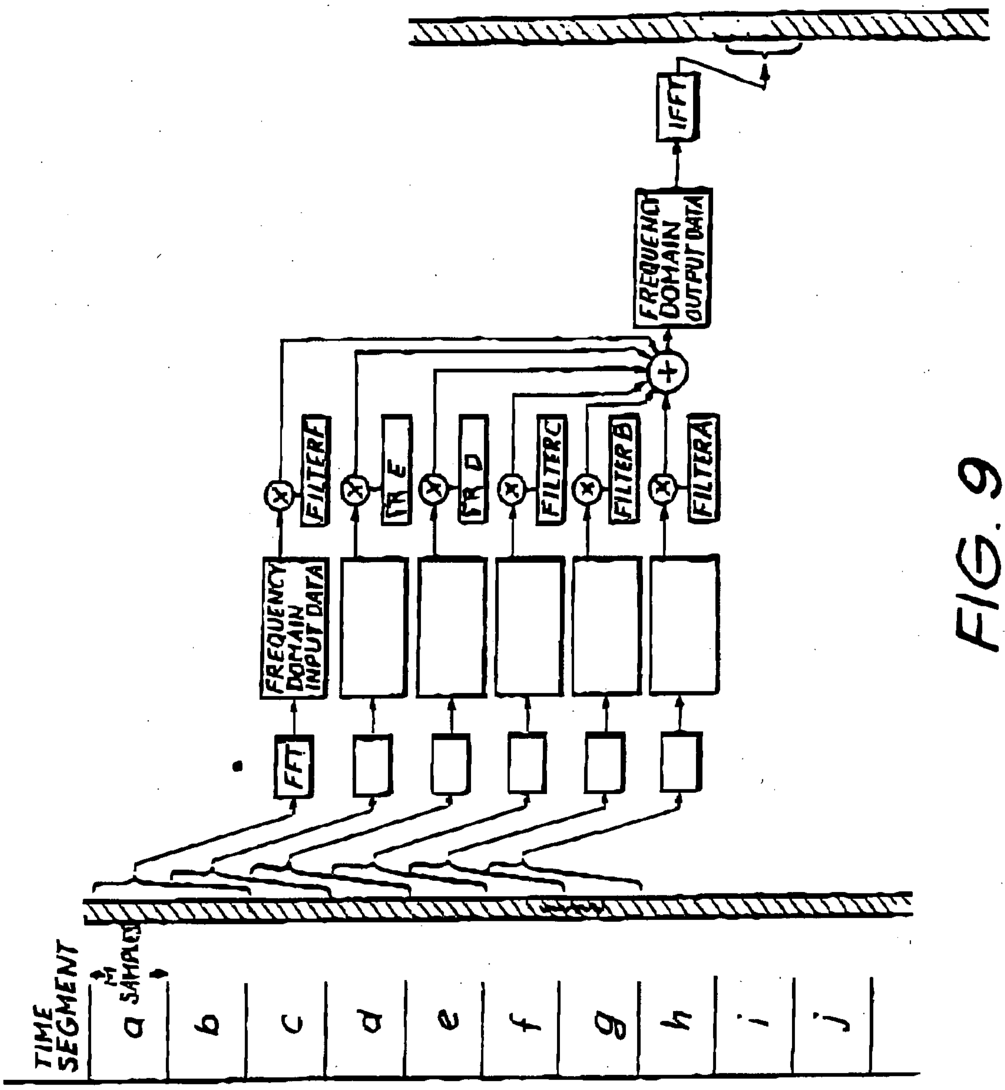

TIME SEGMENT
M SAMPLES
a
b
c
d
e
f
g
h
i
j
FFT
FREQUENCY DOMAIN INPUT DATA
FILTER F
FR E
FR D
FILTER C
FILTER B
FILTER A
FREQUENCY DOMAIN OUTPUT DATA
IFFT

FIG. 9

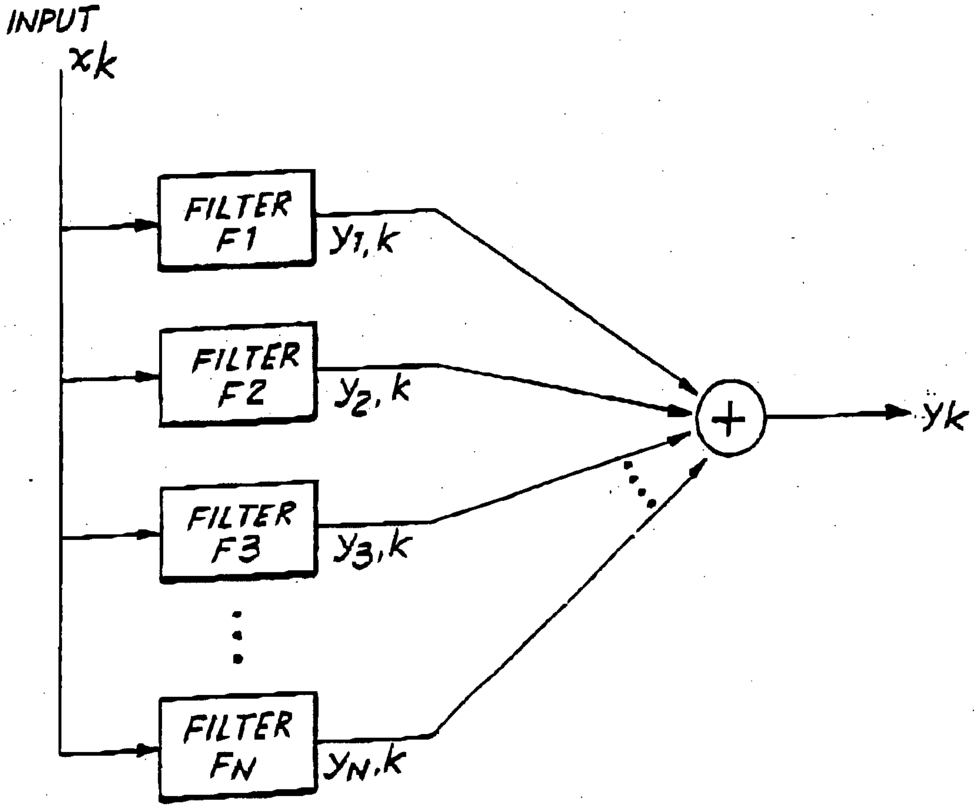

INPUT
$x_k$
FILTER F1
$y_{1,k}$
FILTER F2
$y_{2,k}$
FILTER F3
$y_{3,k}$
FILTER FN
$y_{N,k}$
+
$y_k$

FIG. 10

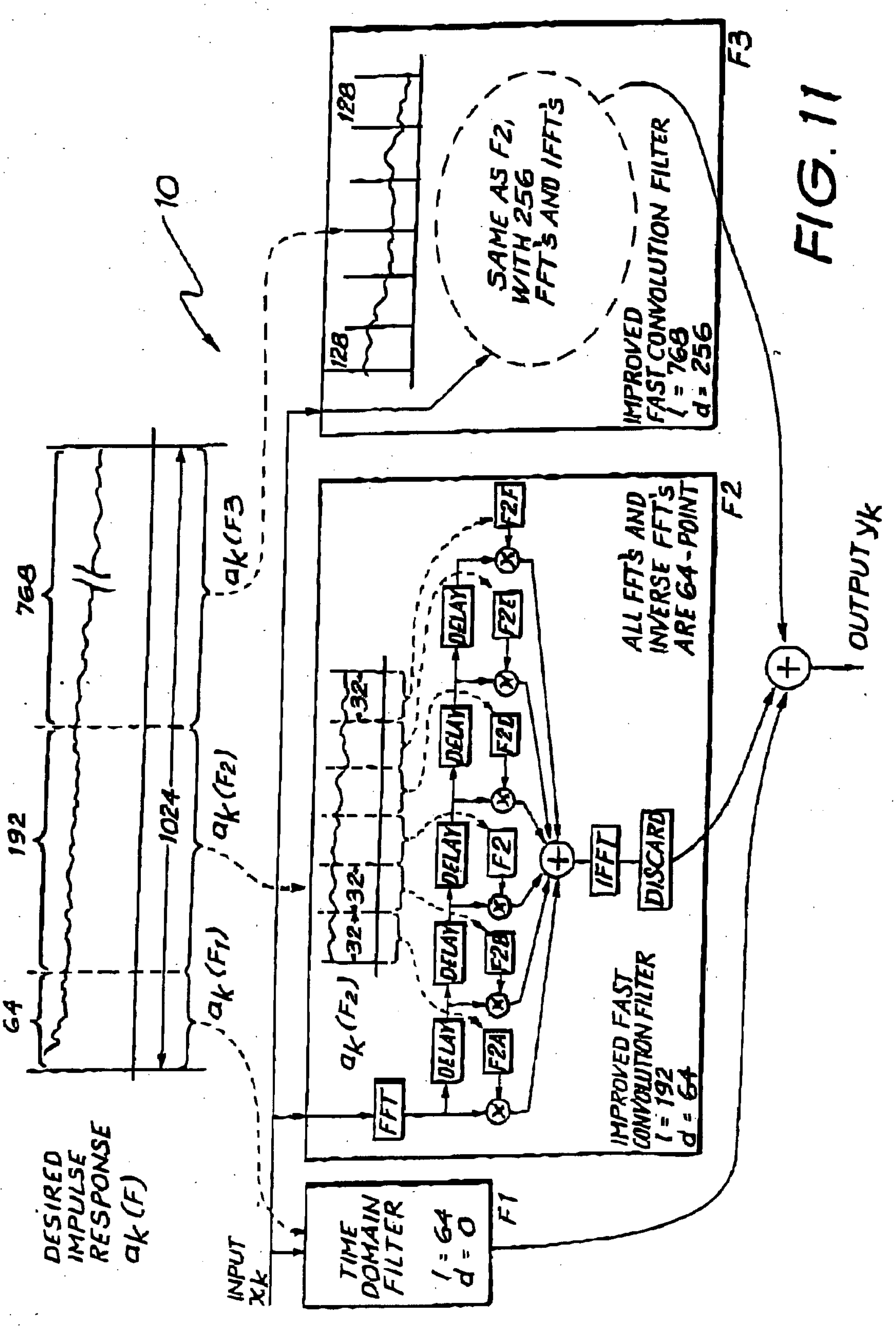

10
DESIRED IMPULSE RESPONSE $a_k(F)$
64
192
768
1024
$a_k(F_1)$
$a_k(F_2)$
$a_k(F_3)$
INPUT $x_k$
TIME DOMAIN FILTER
$l = 64$
$d = 0$
F1
FFT
DELAY
F2A
F2B
F2
F2D
F2E
F2F
$a_k(F_2)$
32
IFFT
DISCARD
IMPROVED FAST CONVOLUTION FILTER
$l = 192$
$d = 64$
ALL FFT'S AND INVERSE FFT'S ARE 64-POINT
F2
128
SAME AS F2, WITH 256 FFT'S AND IFFT'S
IMPROVED FAST CONVOLUTION FILTER
$l = 768$
$d = 256$
F3
OUTPUT $y_k$

FIG. 11

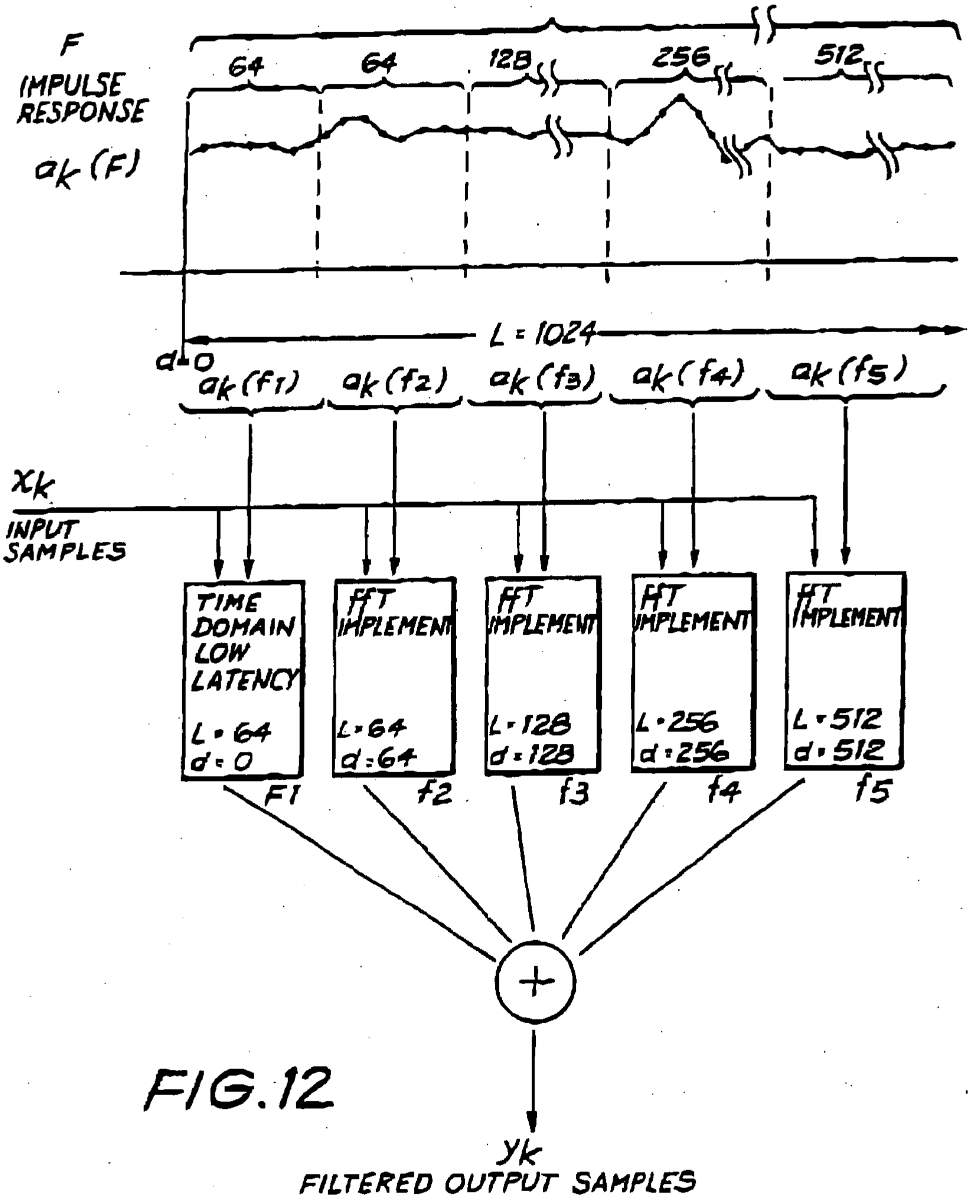
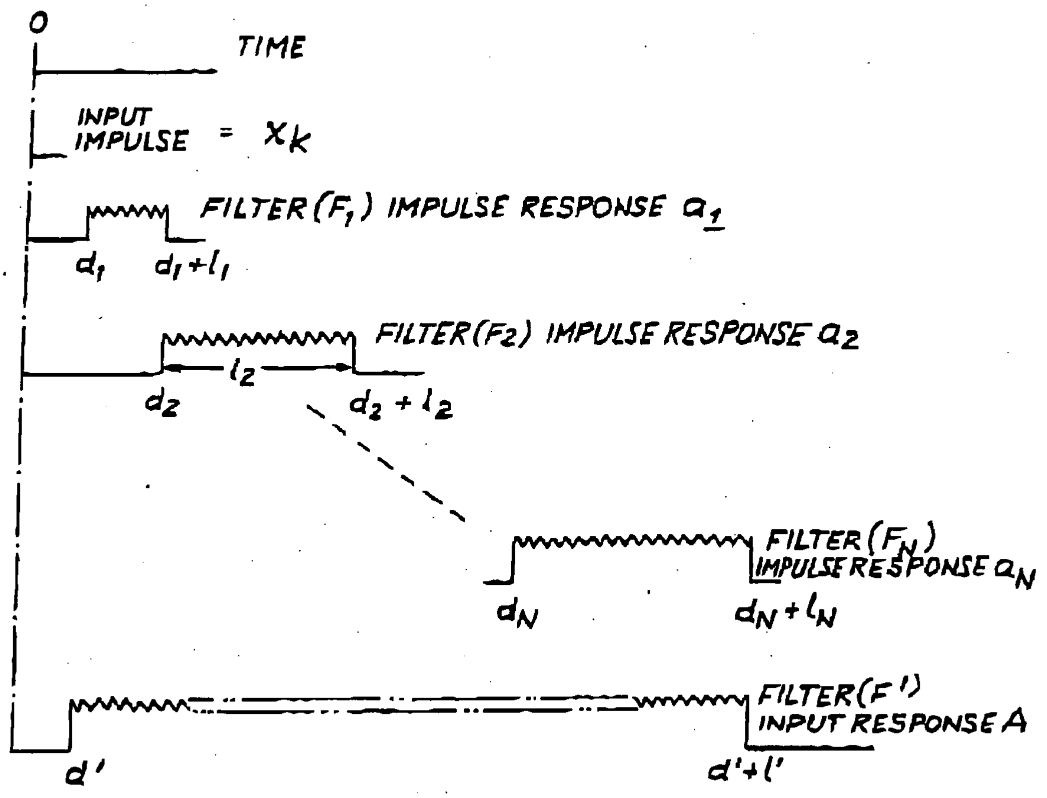

F
IMPULSE RESPONSE
$a_k(F)$
64
64
128
256
512
L = 1024
d = 0
$a_k(f_1)$
$a_k(f_2)$
$a_k(f_3)$
$a_k(f_4)$
$a_k(f_5)$
$x_k$
INPUT SAMPLES
TIME DOMAIN LOW LATENCY
L = 64
d = 0
F1
FFT IMPLEMENT
L = 64
d = 64
f2
FFT IMPLEMENT
L = 128
d = 128
f3
FFT IMPLEMENT
L = 256
d = 256
f4
FFT IMPLEMENT
L = 512
d = 512
f5
+
$y_k$
FILTERED OUTPUT SAMPLES

FIG. 12


0
TIME
INPUT IMPULSE = $x_k$
FILTER ($F_1$) IMPULSE RESPONSE $a_1$
$d_1$
$d_1+l_1$
FILTER ($F_2$) IMPULSE RESPONSE $a_2$
$l_2$
$d_2$
$d_2+l_2$
FILTER ($F_N$) IMPULSE RESPONSE $a_N$
$d_N$
$d_N+l_N$
FILTER ($F'$) INPUT RESPONSE $A$
$d'$
$d'+l'$

THE FILTER $F'$ IS MADE BY SUMMING TOGETHER THE OUTPUTS OF FILTERS $F_1, F_2, \ldots\ldots, F_N$.

EACH FILTER $F_i$ DEFINES OUTPUTS $y_{i,k}$ THE OUTPUT OF FILTER $F_i$ AT TIME SAMPLE $k$, AS BEING

$$y_{i,k} = \sum_{n=d_i}^{d_i+l_i-1} a_{i,n}\, x_{k-n}$$

IF WE SET $y'_k = \sum_{n=1}^{N} y_{i,k}$

THEN WE HAVE $y'_k = \sum_{n=d'}^{d'+l'-1} A_n\, x_{k-n}$

FIG. 13

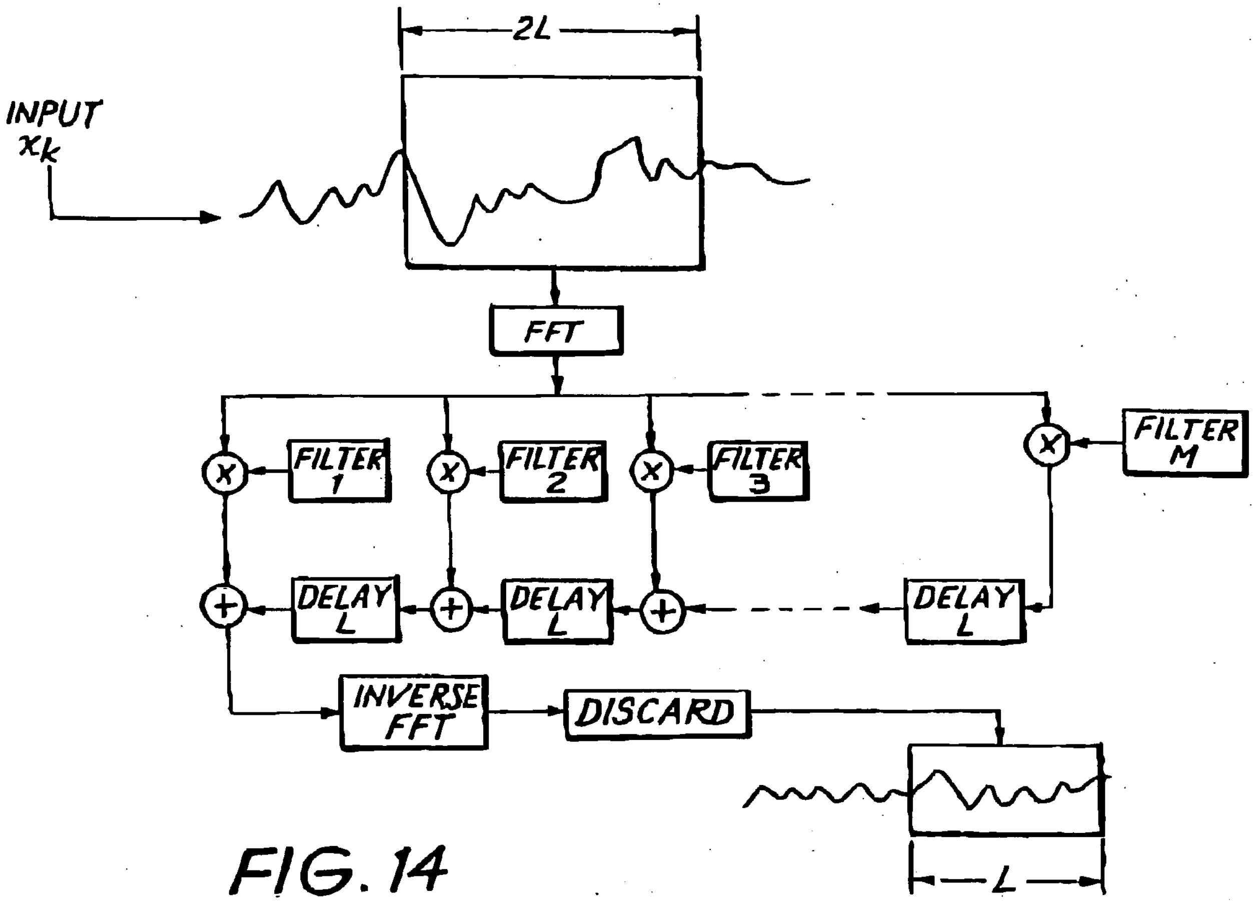

2L
INPUT
$x_k$
FFT
FILTER 1
FILTER 2
FILTER 3
FILTER M
X
DELAY L
DELAY L
DELAY L
+
INVERSE FFT
DISCARD
L

FIG. 14

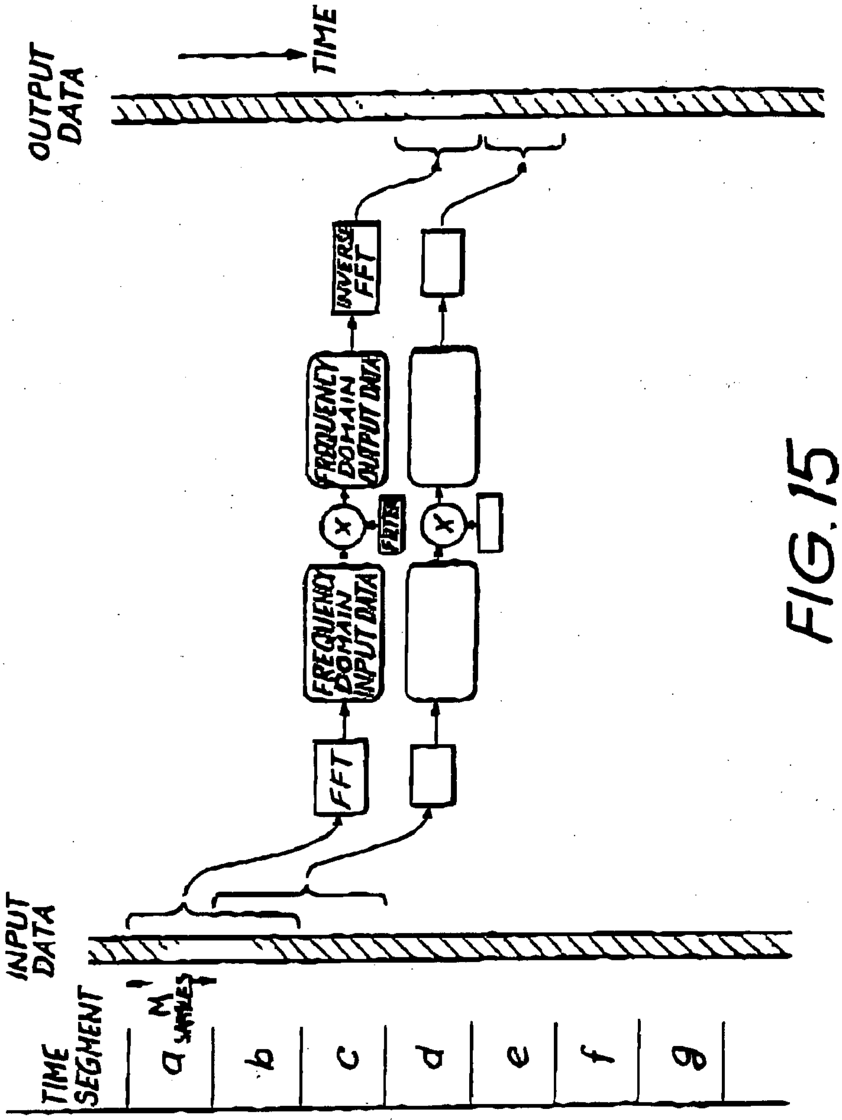

OUTPUT DATA
TIME
INVERSE FFT
FREQUENCY DOMAIN OUTPUT DATA
x
FILTER
x
FREQUENCY DOMAIN INPUT DATA
FFT
INPUT DATA
TIME SEGMENT
M SAMPLES
a
b
c
d
e
f
g

FIG. 15

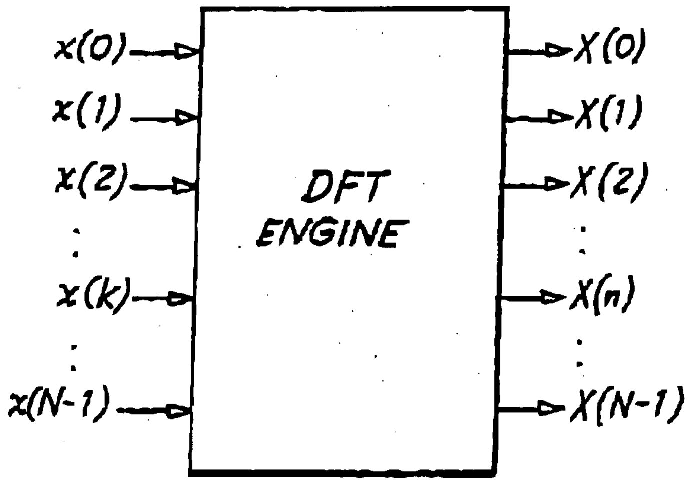


FIG. 16

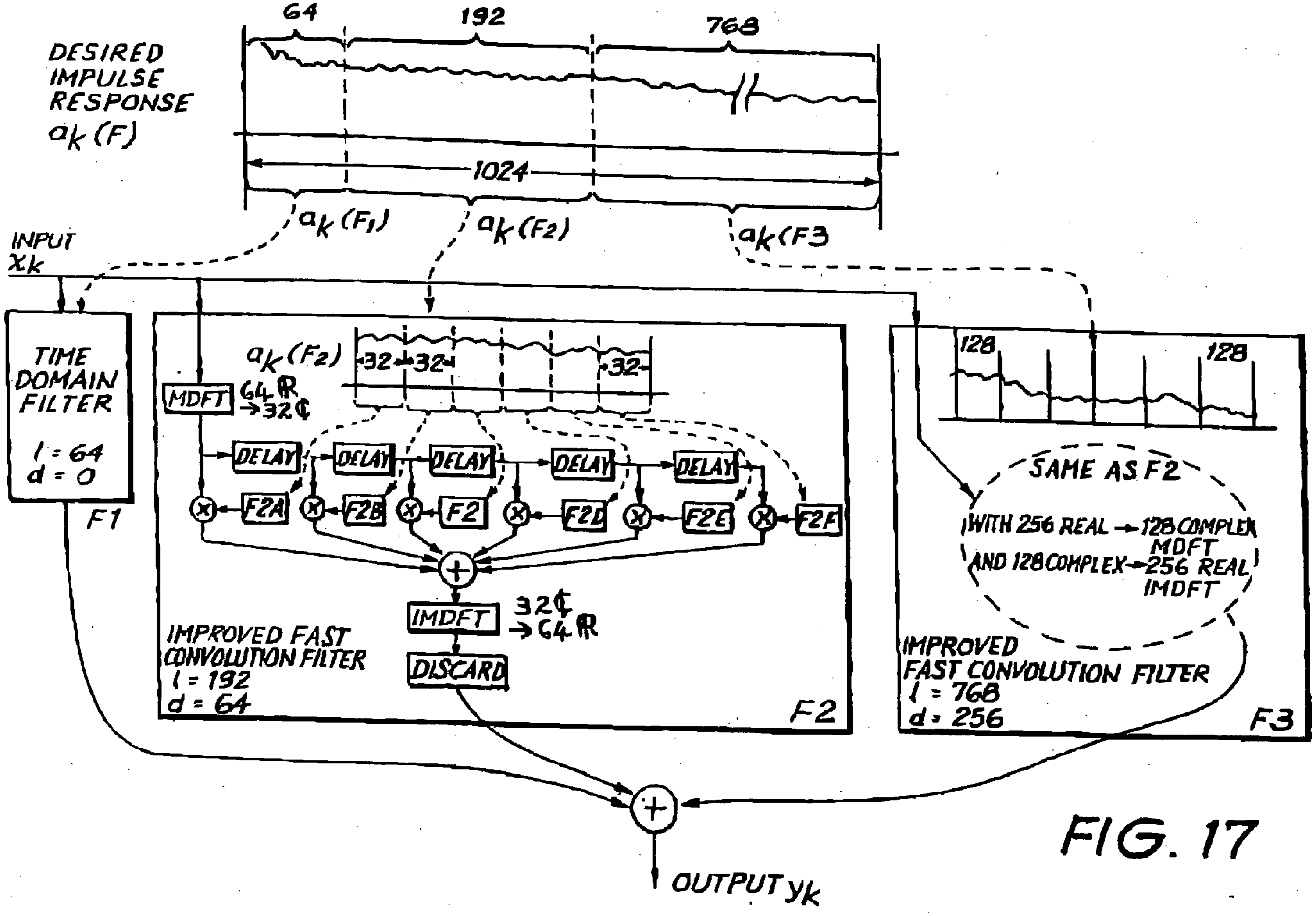

DESIRED IMPULSE RESPONSE $a_k(F)$
64
192
768
1024
$a_k(F_1)$
$a_k(F_2)$
$a_k(F3$
INPUT $x_k$
TIME DOMAIN FILTER
$l = 64$
$d = 0$
F1
$a_k(F_2)$
32
32
32
MDFT
64 ℝ → 32 ℂ
DELAY
DELAY
DELAY
DELAY
DELAY
F2A
F2B
F2
F2D
F2E
F2F
IMDFT
32 ℂ → 64 ℝ
DISCARD
IMPROVED FAST CONVOLUTION FILTER
$l = 192$
$d = 64$
F2
128
128
SAME AS F2
WITH 256 REAL → 128 COMPLEX MDFT
AND 128 COMPLEX → 256 REAL IMDFT
IMPROVED FAST CONVOLUTION FILTER
$l = 768$
$d = 256$
F3
OUTPUT $y_k$

FIG. 17

A2

**PCT** WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

# INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

**(51) International Patent Classification $^5$ :**
**H03H 17/06, 17/02, G06F 15/332**

**A1**

**(11) International Publication Number:** **WO 94/01933**

**(43) International Publication Date:** 20 January 1994 (20.01.94)

**(21) International Application Number:** PCT/AU93/00330

**(22) International Filing Date:** 5 July 1993 (05.07.93)

**(30) Priority data:**

| | | |
|---|---|---|
| PL 3386 | 7 July 1992 (07.07.92) | AU |
| PL 3387 | 7 July 1992 (07.07.92) | AU |
| PL 8357 | 20 April 1993 (20.04.93) | AU |

**(71) Applicant** *(for all designated States except US)*: LAKE DSP PTY. LIMITED [AU/AU]; 1/166 Maroubra Road, Maroubra, NSW 2035 (AU).

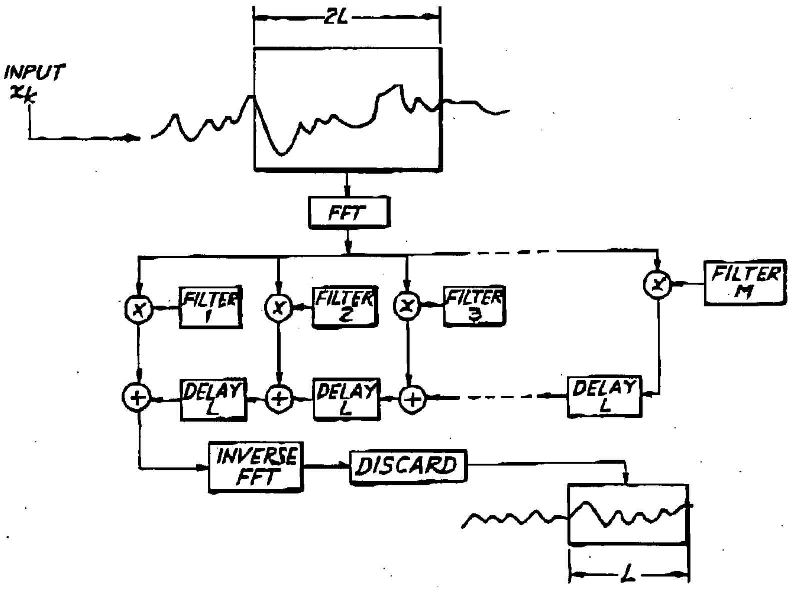
**(72) Inventor; and**
**(75) Inventor/Applicant** *(for US only)* : McGRATH, David, Stanley [AU/AU]; 16 Caerleon Crescent, Randwick, NSW 2031 (AU).

**(74) Agent:** MAXWELL, Peter, Francis; Peter Maxwell & Associates, Blaxland House, 5 Ross Street, North Parramatta, NSW 2151 (AU).

**(81) Designated States:** AT, AU, BB, BG, BR, BY, CA, CH, CZ, DE, DK, ES, FI, GB, HU, JP, KP, KR, KZ, LK, LU, MG, MN, MW, NL, NO, NZ, PL, PT, RO, RU, SD, SE, SK, UA, US, VN, European patent (AT, BE, CH, DE, DK, ES, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG).

**Published**
*With international search report.*

**(54) Title:** DIGITAL FILTER HAVING HIGH ACCURACY AND EFFICIENCY



**(57) Abstract**

Apparatus for and methods of operation of digital filters and certain other electronic digital signal processing devices are provided to improve the accuracy and efficiency of filtering. Particularly, the apparatus and method includes a digital filter with a long impulse response and lower latency, built by operating a numbr of small component filters (F1, F2, F3) in parallel and combining their outputs by addition, with each component filter operating with a different delay such that the net operation of the ensemble of said component filters is the same as a single filter with a longer impulse response, and the latency of the ensemble is equal to the shortest latency of the component filters. At least one group of the component filters is implemented using a Discrete Fourier Transform technique. A Fourier transform processor adapted to efficiently transform strings of real data is also described.

***FOR THE PURPOSES OF INFORMATION ONLY***

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

| | | | | | |
|---|---|---|---|---|---|
| AT | Austria | FR | France | MR | Mauritania |
| AU | Australia | GA | Gabon | MW | Malawi |
| BB | Barbados | GB | United Kingdom | NE | Niger |
| BE | Belgium | GN | Guinea | NL | Netherlands |
| BF | Burkina Faso | GR | Greece | NO | Norway |
| BG | Bulgaria | HU | Hungary | NZ | New Zealand |
| BJ | Benin | IE | Ireland | PL | Poland |
| BR | Brazil | IT | Italy | PT | Portugal |
| BY | Belarus | JP | Japan | RO | Romania |
| CA | Canada | KP | Democratic People's Republic of Korea | RU | Russian Federation |
| CF | Central African Republic | | | SD | Sudan |
| CG | Congo | KR | Republic of Korea | SE | Sweden |
| CH | Switzerland | KZ | Kazakhstan | SI | Slovenia |
| CI | Côte d'Ivoire | LI | Liechtenstein | SK | Slovak Republic |
| CM | Cameroon | LK | Sri Lanka | SN | Senegal |
| CN | China | LU | Luxembourg | TD | Chad |
| CS | Czechoslovakia | LV | Latvia | TG | Togo |
| CZ | Czech Republic | MC | Monaco | UA | Ukraine |
| DE | Germany | MG | Madagascar | US | United States of America |
| DK | Denmark | ML | Mali | UZ | Uzbekistan |
| ES | Spain | MN | Mongolia | VN | Viet Nam |
| FI | Finland | | | | |

# DIGITAL FILTER HAVING HIGH ACCURACY AND EFFICIENCY

## FIELD OF THE INVENTION

The present invention relates to the art of electronic signal processing, and more particularly but not exclusively, to an electronic filtering environment wherein relatively high accuracy and efficiency is desired and a relatively short flow-through delay (termed "latency") is desired.

## DESCRIPTION OF THE PRIOR ART

With reference to Fig. 1, electronic filters are utilised to modify the characteristics of an incoming electronic signal so as to provide an output signal which is modified in some defined fashion. In the case of Fig. 1 a "notch" filter is illustrated wherein, in the frequency domain, frequencies in the spectrum of the incoming signal S1 are attenuated in the $F_1$ to $F_2$ band so as to produce output signal S2.

Such filters can be implemented from entirely analog components although, in more recent times, there is a preference, in many circumstances, to implement the filter in a digital fashion. Digital implementation can be by means of dedicated digital circuitry or by means of computers (micro processors) programmed to operate as a filter.

Filters have many applications in the field of electronic modelling of real world conditions. For example, filters can be used to provide a model of the acoustic characteristics of rooms or halls. Filters are also used to model deficiencies in systems so as to apply appropriate correction factors for the purpose of removing (cancelling) imperfections in signals caused by the deficiencies.

Frequently it is desirable that such processing take place in "real time". Also, it is desirable that there is effectively no delay in filtering of a signal generated in a real/live environment so that the modelling/correcting steps performed by the filter are, to all intents and purposes, without any delay being perceptible to the end user.

To achieve this the delay introduced by the filter F while it performs its filtering function must be reduced to a negligible figure. That is, the time when signal S1 first presents to filter F and the time when the results of the filtering by filter F of the first incoming portion of signal S1 become available at the output of the filter S2 must be almost the same. The delay between these two events is hereinafter referred to as the "latency" of the filter system.

Where the filter F is implemented in a digital manner it may first be necessary to sample the incoming signal S1 (via an analog to digital converter) then perform the filtering function and then convert the digital signal back to an analog signal (by means of a digital to analog converter). The sampling process takes samples of the incoming signal at discrete time intervals $t_i$. The time between each sample is usually the same.

The sampling processing itself introduces finite delays into the system. Additionally, where the filter is implemented by one of the popular fast convolution techniques there is a delay introduced which, in very broad terms, is proportional to the accuracy (or length) of the filter.

Mathematically, the filtering operation (that is, the step of imposing the filter characteristic upon the incoming signal S1 so as to produce outgoing signal S2) is known as "convolution" in the time domain. The step of convolution in the time domain becomes a multiply operation in the frequency domain. That is, if the incoming signal S1 is first sampled, then Fourier transformed into the frequency domain, the frequency response of the filter F is vector multiplied with the Fourier transform of the signal S1. The signal is then inverse Fourier transformed to produce a sampled (convolved) output (which can be converted back to analog form if required).

Figure 2 shows the way a convolver (also known as a Finite Impulse Response (FIR) filter) has its impulse response $\{a_k\}$ measured (for a convolver operating on a treatment of sampled data). For a physical filter, $a_k$ is zero for all $k<0$. For a general input sequence $\{x_k\}$, the filter's output $\{Y_k\}$ is defined as:

$$Y_k=\sum_{i=0}^{\infty} a_i\, x_{k-i} \qquad (1)$$

A linear filter such as this has a measurable latency, d, defined as:-

$$\begin{gathered} a_d \neq 0,\ and \\ a_k = 0\ for\ all\ k < d \end{gathered} \qquad (2)$$

In other words, $a_d$ is the first non-zero value in the output sequence. The latency d is never negative in a physical system. In a similar fashion, for a Finite Impulse Response Filter, we can determine which is the LAST non-zero value in the output sequence. This will give us the length of the impulse response. If we call the length $l$, then this means that $a_{d+l-1}$ is the last non-zero value in the output sequence (see Figure 3).

Typical schemes for implementing FIR filters fall into two categories:-

1. Time domain filters that compute each output sample as each new input sample arrives, thus allowing latencies as low as d=0 or d=1. Typical filter lengths ($l$) are short.
2. Fast convolution filters that compute a number of output samples in a block. Typical filter lengths ($l$) can be very long. The lowest achievable latency is usually related to the filter length, $d \approx l \div K$ or

$$K \approx l \div d \qquad (3)$$

where K is a measure of the efficiency of the particular algorithm used. A typical value of K, for the commonly used fast convolution algorithms such as illustrated in Figs. 4 and 5, is 0.5.

## BRIEF DESCRIPTION OF THE INVENTION

It is an object of at least a preferred embodiment of the present invention to provide a method and apparatus for performing relatively long convolutions on digital sampled data so as to provide relatively higher efficiency for a given length than is ordinarily produced with other methods.

In this specification it is assumed that the filter characteristics can be modelled as approximately linear so that the principles of superposition can be applied.

Accordingly, in one broad form of the invention there is provided a filter with a long impulse response and low latency, built by operating a number of smaller component filters in parallel and combining their outputs by addition, with each component filter operating with a different delay such that the net operation of the ensemble of said component filters is the same as a single filter with a longer impulse response, and the latency of the ensemble is equal to the shortest latency of the said component filters.

Preferably the component filters are implemented in different ways, with some filters adapted to provide low latency, and other filters adapted to provide longer filter lengths, such that the ensemble filter provides both low latency and long impulse response characteristics.

Preferably one or more of the component filters is implemented as a time-domain finite impulse response filter (built with multiply and add operations) and the remainder are implemented using a fast convolution method, such that the time-domain filter(s) provides the lowest latency portion of the ensemble impulse response, and the fast-convolution filter(s) provide the longer filter components.

Preferably the fast-convolution filters are built using the Discrete Fourier Transform or the Fast Fourier Transform.

In an alternative preferred form the fast-convolution filters are built using the Modified Discrete Fourier Transform as described in this specification.

Preferably a group or groups of more than one of the component filters are implemented using fast convolution techniques with the component filters in each group using a transform of the same length, and wherein only one transform operation is performed for each group, so that the component filters in each group share the transformed output from the respective transform operation.

In an alternative preferred form a group or groups of more than one of the component filters are implemented using fast convolution techniques with the component filters in each group using a transform of the same length, and wherein only one inverse transform operation is performed for each group, so that the output from the component filters in each group is summed before being passed to the respective inverse transform operation.

In a further alternative preferred form a group or groups of more than one of the component filters are implemented using fast convolution techniques with the component filters in each group using a transform of the same length, and wherein only one transform operation is performed for each group, so that the component filters in each group share the transformed output from the respective transform operation; and wherein a group or groups of more than one of the component filters are implemented using fast convolution techniques with the component filters in each group using a transform of the same length, and wherein only one inverse transform operation is performed for each group, so that the output from the component filters in each group is summed before being passed to the respective inverse transform operation.

In a further broad form of the invention there is provided a method for building a digital filter with a long impulse response and low latency, built by operating a number of smaller component filters in parallel and combining their outputs by addition, with each component filter operating with a different delay such that the net operation of the ensemble of said component filters is the same as a single filter with a longer impulse response, and the latency of the ensemble is equal to the shortest latency of the said component filters.

Preferably the component filters are implemented in different ways, with some filters adapted to provide low latency, and other filters adapted to provide longer filter lengths, such that the ensemble filter provides both low latency and long impulse response characteristics.

Preferably one or more of the component filters is implemented as a time-domain finite impulse response filter (built with multiply and add operations) and the remainder are implemented using a fast convolution method, such that the time-domain filter(s) provides the lowest latency portion of the ensemble impulse response, and the fast-convolution filter(s) provide the longer filter components.

Preferably the fast-convolution filters are built using the Discrete Fourier Transform or the Fast Fourier Transform.

Preferably the fast-convolution filters are built using the Modified Discrete Fourier Transform as described in this specification.

Preferably a group or groups of more than one of the component filters are implemented using fast convolution techniques with the component filters in each group using a transform of the same length, and wherein only one transform operation is performed for each group, so that the component filters in each group share the transformed output from the respective transform operation.

In an alternative preferred form a group or groups of more than one of the component filters are implemented using fast convolution techniques with the component filters in each group using a transform of the same length, and wherein only one inverse transform operation is performed for each group, so that the output from the component filters in each group is summed before being passed to the respective inverse transform operation.

In a further alternative preferred form a group or groups of more than one of the component filters are implemented using fast convolution techniques with the component filters in each group using a transform of the same length, and wherein only one transform operation is performed for each group, so that the component filters in each group share the transformed output from the respective transform operation; wherein a group or groups of more than one of the component filters are implemented using fast convolution techniques with the component filters in each group using a transform of the same length, and wherein only one inverse transform operation is performed for each group, so that the output from the component filters in each group is summed before being passed to the respective inverse transform operation.

In yet a further broad form of the invention there is provided a digital filter for filtering overlapped groupings of consecutive samples a, b, c, d...., said filter comprising a transform processor of length m, and N filter processors, each of length k, an adder adapted to sum the outputs as they are fed in parallel from said N filter processors, an inverse transform processor of length m; said consecutive samples a, b, c, d, ..... being fed in blocks of length m, each of said blocks being passed through said transform processor and then through each of said N filter processors with a delay of zero before being passed through a first filter processor of said N filter processors, a delay of k before being passed through a second filter processor and so on up to a delay of (N-1)k before being passed through the Nth filter processor; whereby a filter of effective length Nk is effected with a latency corresponding to that of a conventional filter of length k.

In a further broad form of the invention there is provided a method of implementing a filter with a relatively high length/latency efficiency K; said method comprising transforming progressive, consecutive and overlapping portions of input data into the frequency domain, to produce corresponding transformed data, storing said corresponding transformed data, so as to effect passing said portions of transformed data through a transform processor consecutively; feeding a first transformed portion through a first filter function processor; then feeding said first transformed portion through an N-1th filter function processor whilst an N-1th transformed portion is being fed through said first filter function processor; and so on so that a continuously moving N consecutive blocks of input data are transformed and passed through N filter function processors; adding the output from said N filter function processors; inverse transforming said output; performing a discard operation as necessary whereby consecutive portions of filtered output data are produced from said progressive, consecutive and overlapping portions of input data.

In a further broad form of the invention there is provided a method of implementing a filter with a relatively high length/latency efficiency K; said method comprising applying a mathematical transform to progressive, consecutive and overlapping portions of input data so as to produce corresponding transformed data; performing a mathematical operation on individual ones (i.e. two or more) of said transformed data; superposing (by addition) the data resulting from said mathematical operations so as to produce resultant data; applying an inverse mathematical transform to said resultant data so as to produce filtered output data.

Preferably each said filter function processor is processing a filter whose Impulse Response is half the length of the transform processor.

Preferably said transform processor is a Fast Fourier Transform Processor and said inverse transform processor is an inverse Fast Fourier Transform Processor and said filter function processor is effected by a multiply operation of transformed input data with an impulse response corresponding to a selected portion of a desired impulse response for the entire filter.

Preferably said method is optimized whereby an approximately equal amount of processing time is spent on Fourier transformation as on filter function (multiply) operation.

In a further broad form of the invention there is provided a method of implementing a filter with a relatively high length/latency efficiency K; said method comprising transforming progressive, consecutive and overlapping portions of input data into the frequency domain, performing a mathematical operation on individual ones of said transformed signals, superposing (by addition) the consecutive signals resulting from said mathematical operations, inverse transforming the resultant signal from the frequency domain back to the time domain and outputting said signal.

Preferably said transform is a Fast Fourier Transform and said inverse transform is an Inverse Fast Fourier Transform.

Preferably said mathematical operation is a vector multiply of the Fourier transformed input signal segment with the frequency response of the desired (time domain) filter characteristic.

Preferably the underlying operation is an overlap and discard operation on successive, overlapping portions of input data.

Alternately the underlying operation is an overlap and add operation on successive, overlapping portions of input dated.

In a particular preferred form the Fourier transform of any given overlapping block of input samples is taken only once and reused as required.

Preferably, the inverse Fourier transform of any given summed grouping of filtered data is performed only once.

In a particular preferred form the above described method is utilised to implement at least some of the filter modules of a composite electrical filter; said composite electrical filter comprising a plurality of filter modules arranged to receive in parallel an incoming input signal for filtering; the output from each of said filter modules being summed to produce a composite filtered output signal; each of said filter modules adapted to have an impulse response that is a selected portion of the impulse response of the composite filter.

It is preferable to minimise the overlap of the selected portions of impulse response, or to make them not overlap at all.

Preferably the length of each of said filter modules is different with the characteristic of the shorter length filter modules adapted to process first (or earlier) portions of said impulse response and longer length filter modules adapted to process following (or later) portions of said impulse response.

Preferably said longer length filter modules are adapted to filter progressively longer portions of said impulse response.

In a particular preferred form, the shortest module of said plurality of filter modules is a time-domain (low latency) filter whilst additional ones of said filter modules are longer fast-convolution (longer latency) filters implemented using the Fast Fourier Transform method described above or other traditional fast convolution methods.

It is an object of at least a further preferred embodiment of the present invention to provide a method and apparatus for performing relatively long convolutions on digital sampled data so as to provide relatively lower latency than is ordinarily incurred with other methods.

It is assumed that the filter characteristics can be modelled as approximately linear so that the principles of superposition can be applied.

Accordingly, in yet a further broad form of the invention, there is provided a composite electrical filter comprising a plurality of filter modules arranged to receive in parallel an incoming input signal for filtering; the output from each of said filter modules being summed to produce a composite filtered output signal; each of said filter modules adapted to have an impulse response that is a selected portion of the impulse response of the composite filter.

It is preferable to minimise the overlap of the selected portions of impulse response, or to make them not overlap at all.

Preferably the length of each of said filter modules is different with the characteristic of the shorter length filter modules adapted to process first (or earlier) portions of said impulse response and longer length filter modules adapted to process following (or later) portions of said impulse response.

Preferably said longer length filter modules are adapted to filter progressively longer portions of said impulse response.

In a particular preferred form, the shortest module of said plurality of filter modules is a time-domain (low latency) filter whilst additional ones of said filter modules are longer fast-convolution (longer latency) filters.

Preferably only the shortest of said filter modules is a time domain filter.

In a further particular preferred form, where the number of said filter modules is N comprising filters $F_1$, $F_2 \ldots F_N$ filter module $F_1$ is a filter with very low latency implemented with time domain techniques whilst all other filter modules $F_i$ are implemented with fast convolution techniques and these fast convolution filters

$F_i$ are composed of a sequence of filters each with longer filter length than its predecessor and hence each with longer latency, but still preserving the property that $d_{i+1}=d_i+l_i$ whereby it is ensured that the composite filter F output formed by summing together the N component filter outputs has an impulse response without any "holes" in it.

In a further broad form of the invention there is provided a method of filtering sampled data so as to achieve a relatively long length but short latency filtering of said data, said method comprising passing said data in parallel through a plurality of sub-filters and summing the output samples from all of said sub-filters to produce filtered sampled data; and wherein the Impulse Response of each of said sub-filters is a selected portion of the desired Impulse Response of the filter characteristic required to produce said filtered sampled data from said sampled data and wherein each said selected portion is selected for each of said sub-filters such that the output from all of said sub-filters, when summed, behaves as if filtered through a filter having said desired Impulse Response.

In a particular implementation of the invention, there is provided a method and filter incorporating a fourier transform processor adapted to efficiently transform strings of real numbers; said processor operating according to the following method:

1. Take the input vector $x(k)$ of length N, where each element of $x(k)$ is real.

2. Create the vector x'($l$), a complex vector of length N/2 by application of the following:

$$x'(l) = [x(l) - jx(l+\frac{N}{2})]e^{\frac{-\pi jl}{N}} \text{ for } 0 \leq l < N/2 \qquad (9)$$

3. Compute the N/2 point DFT of x'($l$) to give the N/2 complex result vector Z(p):

$$Z(p)=DFT_{(N/2)}[x'(l)] \qquad (10)$$

Embodiments of the invention will now be described with reference to the accompanying drawings wherein:-

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 is a generalised block diagram of a filter operation in the frequency domain,

Fig. 2 defines the basic terminology used for a convolution filter,

Fig. 3 defines the latency and length of the filter of Fig. 2,

Fig. 4 illustrates in a diagrammatic flow chart form, a prior art method of processing sampled data by a Fast Fourier Transform approach,

Fig. 5 further illustrates the approach of Fig. 4,

Fig. 6,7,8 develop a method of filtering according to a generalised first embodiment of the invention whereby a relatively high efficiency factor K can be achieved as compared with the approach of Fig. 4, Fig. 9 is a diagram of an embodiment of the invention implementing the method of Fig. 8 where the number of sub-filters is 6,

Fig. 10 illustrates in block diagram form a summed filter a part of which can be implemented advantageously with the filter of Fig. 8,

Fig. 11 is a block diagram of the summed filter of Fig. 10 incorporating sub-filters some of which implement the method of Fig. 8,

Fig. 12 illustrates the manner of processing of an input signal by an example of the filter arrangement of Fig. 10 which utilised five filter portions,

Fig. 13 illustrates the manner of selection of the filter characteristics of the filter of Fig. 12,

Fig. 14 is a bloc diagram of an alternative implementation of the summed filter of Fig. 8,

Fig. 15 illustrates a typical flow of a (prior art) fast convolution algorithm implementation suitable for filters $F_2$ - $F_5$ of Fig. 12,

Fig. 16 illustrates a DFT engine which forms the basis for an explanation of a fourier transform algorithm optimised to process real number strings, and

Fig. 17 is a block diagram of a further embodiment of the invention wherein the summed filter of Fig. 10 is implemented utilizing a Modified Discrete Fourier Transform.

## DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS OF THE INVENTION

### 1. High Efficiency Filter

Figure 4 illustrates the time-flow of a typical fast-convolution algorithm. This is an overlap-discard algorithm implemented using the Fast Fourier Transform (FFT). 2M words of input data that arrives during time segments a and b is processed fully during time segment c with a forward FFT, a vector multiply, and an inverse FFT. The resulting M words of output data are buffered, to be presented at the output during time segment d. The FFT and inverse FFT(IFFT) are only used to transform the data between the time-domain and frequency domain. The actual filter operation is executed in the vector multiply operation, which actually takes only a small fraction of the total compute time. So, the relevant parameters of the filter of Fig. 4 are:-

Length = M,<br>
Latency = 2M,<br>
and, therefore K = 0.5

With reference to Figs. 6, 7 and 8, the rationale behind the method and apparatus according to at least one embodiment of the present invention is derived.

Fig. 6 illustrates a filter of length ML where the filter characteristics of each of the component filters F1, F2, .... $F_n$ are separate, discrete, component portions of the desired filter characteristic for the entire filter assembly. The delays L, 2L.... (M-1) L are imposed as illustrated so as to recreate, following addition, an output $y_k$ equivalent to that achieved by passing input samples $x_k$ through a filter having the filter characteristic from which the filter characteristic portions for filters F1, F2..... were derived. Fig. 7 is derived by implementing the filters F1, F2.... of Fig. 6 using the Fast Fourier Transform algorithm of Fig. 5.

With reference to Fig. 8, reorganisation of the filter of Fig. 7 allows the use of only one Fast Fourier Transform module 11 and one inverse Fast Fourier Transform module 12. It is implicit that the Fast Fourier Transform module is adapted to process a block of samples from input $x_k$ equal to twice the length of each of the filters Filter 1, Filter 2, Filter 3...... illustrated in Fig. 8.

As previously stated the filter characteristic (impulse response) of each consecutive filter F1, F2.... is taken from and corresponds to consecutive corresponding portions from the impulse response desired of the entire filter module.

The time delay L before each Fast Fourier transformed block of data is passed through the next filter is equal to half the sample length originally processed by the Fast Fourier Transform module.

Figure 9 shows the computation of one block of output data, in a similar style to Figure 4, but using the improved length/latency efficiency method derived in Figs. 6, 7 and 8.

The method of Fig. 8 as used by Fig. 9 is summarised below.

During time segment h, the input data that arrived during time segments f and g is FFT'd and the resulting block of Frequency Domain Input Data is stored for future use. We then compute the next block of Frequency Output Data, which is inverse FFT'd and presented as output during time segment i. The old way of computing fast-convolution simply took the latest block of Frequency Domain Input Data, and multiplied it by a vector that represents the desired filter response, to get the new Frequency Domain Output Data. The improved length/latency efficiency method uses a number of previous Frequency Domain Input Data blocks to compute the new Frequency Domain Output Data block, as shown in Figure 9. In this example, the blocks of filter data are called Filter A, Filter B, ..., Filter F. In this example, the filter implemented is 6 times as long as the filter implemented in Figure 4, but with no greater latency. By comparison with Fig. 4, the relevant parameters of the filter of Fig. 9 are:-

Length = 6M,

Latency = 2M,

and, therefore K = 3

Fig. 8 summarises the logic behind the implementation of the embodiment of Fig. 9.

Particularly, it will be noted that the progressive delays L, 2L, 3L, ...(M-1)L of Fig. 8 are achieved in Fig. 9 by the taking of delayed, overlapped groupings of consecutive samples a, b, c, d, ...

The above described filter arrangement can be used advantageously in a low-latency FIR filter arrangement such as illustrated in Fig. 10.

Figure 10 shows an architecture for implementing an FIR filter by adding together N filters. If each filter is characterised as: Filter $F_i$, latency $d_i$, length $l_i$, then generally the N filters are chosen so that their latencies are ordered in ascending order, and furthermore $d_{i+1}=d_i+l_i$. This means that the first non-zero value in the impulse response of filter $F_{i+1}$, comes immediately following the last non-zero value in the impulse response of filter $F_i$. Hence this summation of filters results in a single, longer filter with its impulse response being the sum of the impulse responses of the N component filters.

The important property of this filter is the length/latency efficiency, K, which is higher than any of the component filter efficiencies.

That is, the filter of Fig. 10 uses the technique of adding together several filters to form a new filter which is as long as the sum of the component filter lengths, and whose latency is as short as the latency of the lowest-latency component filter.

Fig. 11 shows an implementation of the low-latency filter 10 of Fig. 10 wherein there are three filter modules F1, F2, F3. The first module F1 is a low-latency (d=0) time domain filter whilst filters F2 and F3 are implemented according to the embodiment described in respect of Figs. 8 and 9.

SUBSTITUTE SHEET

2. A low-latency FIR filter

As previously described. Fig. 10 shows an architecture for implementing an FIR filter by adding together N filters. If each filter is characterised as: Filter $F_i$, latency $d_i$, length $l_i$, then generally the N filters are chosen so that their latencies are ordered in ascending order, and furthermore $d_{i+1}=d_i+l_i$. This means that the first non-zero value in the impulse response of filter $F_{i+1}$, comes immediately following the last non-zero value in the impulse response of filter $F_i$. Hence this summation of filters results in a single, longer filter with its impulse response being the sum of the impulse responses of the N component filters.

The important property of this filter is the length/latency efficiency, K, which is higher than any of the component filter efficiencies.

That is, the filter of Figs. 10 and 12 uses the technique of adding together several filters to form a new filter which is as long as the sum of the component filter lengths, and whose latency is as short as the latency of the lowest-latency component filter.

Particularly, the composite filter assembly of Fig. 12 utilises the technique of combining a first time-domain (low latency) filter with additional fast-convolution (longer latency) filters to maximise filter length while minimising latency. This technique is implemented by adding together N filters, $F_1$, $F_2$, ...$F_N$ where $F_1$ is a filter with very low latency, implemented with time-domain techniques, and the other filters, $F_i$, are each implemented with fast-convolution techniques. More specifically, the assembly adopts the technique whereby the N-1 fast-convolution filters, Fi, are composed of a sequence of filters, each with longer filter length than its predecessor, and hence each with longer latency, but still preserving the property that $d_{i+1}=d_i+l_i$. This ensures that the filter, F, which is made by summing together the N component filters, has an impulse response without any "holes" in it.

With particular reference to Fig. 12 the composite filter F comprises five filter portions F1, F2, F3, F4 and F5. The impulse response $a_k$ of the composite filter F is illustrated at the top of Fig. 12 and has a total sample length of 1024 samples.

Filter F1 has an impulse response comprising the first 64 samples of the impulse response $a_k$. That is, the filter has a length of 64 samples. The filter as implemented has a low latency filter (such as is referenced in Motorola document APR 7/D in respect of the DSP 56000 series of Integrated Circuits). This filter has an effective latency of 0.

The subsequent filters F2, F3, F4, F5 are implemented using fast convolution digital techniques. Fig. 15 illustrates the basic algorithm for such techniques which comprises taking the fast Fourier transform of the incoming sampled data, multiplying the transformed data samples by the impulse response of the filter, converting the fast Fourier transformed data samples back to the time domain by use of an inverse fast Fourier transform and then outputting the data. An overlap/discard method is used whereby only a portion of the output data is utilised.

The length and latency of the additional filters F2, F3, F4, F5 is selected according to the rule illustrated diagrammatically in Fig. 13, whereby each filter portion has a latency equal to the sum of the length and the latency of the immediately preceding filter portion.

In this case the end result is a filter having a total length of 1024 samples and a latency of 0.

Fig. 14 illustrates a variation of the filter of Fig. 8 wherein delay is introduced after the filter algorithm is applied in the frequency domain.

3. Optimized Real String Handling

With reference to Fig. 16, a common method of frequency analysis is via the Discrete Fourier Transform (hereafter referred to as the DFT), which can be implemented efficiently in electronic apparatus using the Fast Fourier Transform algorithm (hereafter referred to as the FFT).

The DFT is formulated to operate efficiently when its input data and output data are both complex (having a real and imaginary component). When the data input to the DFT is real, the output data from the operation will contain some redundancy, indicating that some of the processing that led to this output data was unnecessary.

In this embodiment what is described is a new transform for operating on real numbers in the digital environment, that has many of the same applications as the DFT, but without the inefficiencies of the DFT for operation on real numbers. For the purposes of this document, the algorithm described herein will be named the Modified Discrete Fourier Transform (MDFT).

The DFT is computed according to the equation below:

$$X(n)=\sum_{k=0}^{N-1}x(k)e^{\frac{-2\pi jnk}{N}} \qquad (1)$$

If the input data x(k) is real (ie. it has no imaginary component), the output data X(n) has the following properties:

$$\begin{aligned} &X(0)\in\Re \\ &X(\tfrac{N}{2})\in\Re \qquad (2)\\ &X(n) = [X(N-n)]^{*} \; for \; 0<n<\tfrac{N}{2} \end{aligned}$$

Where the * operator is used to signify complex conjugation. This means that the imaginary part of X(0), the imaginary part of X(N/2) and all {X(n):N/2<n<N} are redundant. The process of extracting only the necessary information out of the DFT output is therefore not trivial.

An alternative transform is shown below:

$$Y(n) = \sum_{k=0}^{N-1}x(k)e^{\frac{-2\pi jk(n+\frac{1}{2})}{N}} \qquad (3)$$

This is like a standard DFT except that the output vector Y(n) represents the signal's frequency components at different frequencies to the DFT. The output vector Y(n) has redundancies (just as the DFT output X(n) has), except that the redundant part of the data is more clearly extracted from Y(n) that from X(n). The redundancy in Y(n) that results when x(k) is real can be expressed as follows:

$$Y(n) = [Y(N-1-n)]^{*} \qquad (4)$$

This implies that the second half of this vector Y(n) is simply the complex conjugate of the first half, so that only the first half of the output vector is required to contain all of the information, when x(k) is real.

An alternative view of the above equation is that all of the odd elements of the vector are simply the complex conjugate of the even elements:

$$\begin{aligned} Y(1) &= [Y(N-2)]^{*} \\ Y(3) &= [Y(N-4)]^{*} \\ &\cdots \\ Y(N-3) &= [Y(2)]^{*} \\ Y(N-1) &= [Y(0)]^{*} \end{aligned} \qquad (5)$$

This means that we only need to compute the even elements of Y(n) to obtain all of the required information from the modified DFT of the real signal x(k). We can name the array Z(p) the array that contains the even elements from Y(n), as follows:

$$Z(p) = Y(2p) \ for \ 0 \le p < \frac{N}{2} \qquad (6)$$

Based on our previous equation for Y(n), we get:

$$Z(p) = Y(2p) = \sum_{k=0}^{N-1} x(k) e^{\frac{-2\pi kj(2p+\frac{1}{2})}{N}} \qquad (7)$$

which after some manipulation becomes:

$$Z(p) = \sum_{k=0}^{\frac{N}{2}-1} [x(k) - jx(k+\frac{N}{2})] e^{\frac{-\pi jk}{N}} e^{\frac{-2\pi jpk}{(N/2)}} \qquad (8)$$

if we create an N/2 length complex vector from the N length real vector:

$$x'(l) = [x(l) - jx(l + \frac{N}{2})]e^{\frac{-\pi jl}{N}} \quad for\ 0 \leq l < N/2 \qquad (9)$$

then we can see that:

$$Z(p) = DFT_{(N/2)}[x'(l)] \qquad (10)$$

This means that we have computed the vector Z(p) by using a DFT of length N/2.

We say that Z(p)=MDFT[x(k)] (where MDFT indicates the Modified Discrete Fourier Transform operator). The procedure to follow for computing Z(p) is then as follows:

1. Take the input vector x(k) of length N, where each element of x(k) is real.
2. Create the vector x'(*l*), a complex vector of length N/2 by the method of equation (9) above.
3. Compute the N/2 point DFT of x'(*l*) to give the N/2 complex result vector Z(p).

The MDFT has many properties that make it useful in similar applications to the DFT. Firstly, it can be used to perform linear convolution in the same way as the DFT. Secondly, it has an inverse transform that looks very similar to the forward transform:

$$x'(k) = IDFT_{(N/2)}[Z(p)] \qquad (11)$$

where IDFT indicates the N/2 point Inverse Discrete Fourier

Transform.

The algorithm can be implemented in an electronic apparatus supplied with a set of N real numbers and producing N/2 complex output numbers, representing the MDFT of the input data. This apparatus uses digital arithmetic elements to perform each of the arithmetic operations as described in the preceding text.

Another embodiment of the present invention is a pair of apparatus, the first of which computes an MDFT as described in the previous paragraph, and the second of which computes an inverse MDFT, using the arithmetic procedures described previously in this document. By passing overlapped blocks of data from a continuous stream of input data through the MDFT computer, then multiplying the Z(p) coefficients by appropriate filter coefficients, then passing the resulting data through the Inverse MDFT computer, and recombining segments of output data appropriately, a modified Fast Convolution processor can be built.

The above described a modification to the DFT that makes it more useful in a number of applications particularly but not limited to the real time filter applications previously described. All of these extensions to the DFT can also be applied to the FFT algorithm and other fast implementations of the DFT.

## Example 1

Fig. 17 illustrates an implementation of the summed filter of Fig. 11 wherein the Modified Discrete Fourier Transform (MDFT) described immediately above is applied for the purposes of transforming the data stream into the frequency domain and the corresponding Inverse Modified Discrete Fourier Transform (IMDFT) is applied following application of the filter algorithm and prior to discard for conversion from the frequency domain.

In filter F2 of Fig. 17, the MDFT takes 64 real words of input and produces 32 complex words of output. The IMDFT takes 32 complex words of input and produces 64 real words of output.

In filter F3 of Fig. 17 the MDFT takes 256 real words of input and produces 128 complex words of output. The IMDFT takes 128 complex words of input and produces 256 real words of output.

The filter of Fig. 17 is implemented using a Motorola DSP 56001 processor incoporating software (bootable from ROM or from another host computer) to implement the algorithm. The delay elements are implemented using a bank of external memory chips comprising three MCM 6206 memory chips.

Data input and output between the analog and digital domains is effected by an ADC and DAC chip, the Crystal CS 4216, communicating via the syncrhonous serial communication port of the DSP 56001.

INDUSTRIAL APPLICABILITY

Embodiments of the invention may be applied to digital filters implemented in software, hardware or a combination of both for applications such as audio filtering or electronic modelling of acoustic system characteristics. The method is broadly applicable in the field of signal processing and can

SUBSTITUTE SHEET

be used to advantage, for example, in: adaptive filtering; audio reverberation processing; adaptive echo cancellation; spatial processing; virtual reality audio; correlation, radar; radar pulse compression; deconvolution; seismic analysis; telecommunications; pattern recognition; robotics; 3D acoustic modelling; audio post production (including oralisation, auto reverberant matching); audio equalisation; compression; sonar; ultrasonics; secure communication systems; digital audio broadcast, acoustic analysis; surveillance; noise cancellation; echo cancellation.

The above describes only some embodiments of the present invention and modifications obvious to those skilled in the art can be made thereto without departing from the scope and spirit of the present invention.

CLAIMS

1. A digital filter with a long impulse response and low latency, built by operating a number of smaller component filters in parallel and combining their outputs by addition, with each component filter operating with a different delay such that the net operation of the ensemble of said component filters is the same as a single filter with a longer impulse response, and the latency of the ensemble is equal to the shortest latency of the said component filters.

2. The filter of claim 1 wherein the component filters are implemented in different ways, with some filters adapted to provide low latency, and other filters adapted to provide longer filter lengths, such that the ensemble filter provides both low latency and long impulse response characteristics.

3. The filter of claim 1 wherein one or more of the component filters is implemented as a time-domain finite impulse response filter (built with multiply and add operations) and the remainder are implemented using a fast convolution method, such that the time-domain filter(s) provides the lowest latency portion of the ensemble impulse response, and the fast-convolution filter(s) provide the longer filter components.

4. The filter of claim 3 wherein the fast-convolution filters are built using the Discrete Fourier Transform or the Fast Fourier Transform.

5. The filter of claim 3 wherein the fast-convolution filters are built using the Modified Discrete Fourier Transform as claimed in claim 33.

SUBSTITUTE SHEET

6. The filter of claim 1 wherein a group or groups of more than one of the component filters are implemented using fast convolution techniques with the component filters in each group using a transform of the same length, and wherein only one transform operation is performed for each group, so that the component filters in each group share the transformed output from the respective transform operation.

7. The filter of claim 1 wherein a group or groups of more than one of the component filters are implemented using fast convolution techniques with the component filters in each group using a transform of the same length, and wherein only one inverse transform operation is performed for each group, so that the output from the component filters in each group is summed before being passed to the respective inverse transform operation.

8. The filter of claim 1 wherein a group or groups of more than one of the component filters are implemented using fast convolution techniques with the component filters in each group using a transform of the same length, and wherein only one transform operation is performed for each group, so that the component filters in each group share the transformed output from the respective transform operation; and wherein a group or groups of more than one of the component filters are implemented using fast convolution techniques with the component filters in each group using a transform of the same length, and wherein only one inverse transform operation is performed for each group, so that the output from the

component filters in each group is summed before being passed to the respective inverse transform operation.

9. A method for building a digital filter with a long impulse response and low latency, built by operating a number of smaller component filters in parallel and combining their outputs by addition, with each component filter operating with a different delay such that the net operation of the ensemble of said component filters is the same as a single filter with a longer impulse response, and the latency of the ensemble is equal to the shortest latency of the said component filters.

10. The method of claim 9 wherein the component filters are implemented in different ways, with some filters adapted to provide low latency, and other filters adapted to provide longer filter lengths, such that the ensemble filter provides both low latency and long impulse response characteristics.

11. The method of claim 9 wherein one or more of the component filters is implemented as a time-domain finite impulse response filter (built with multiply and add operations) and the remainder are implemented using a fast convolution method, such that the time-domain filter(s) provides the lowest latency portion of the ensemble impulse response, and the fast-convolution filter(s) provide the longer filter components.

12. The method of claim 11 wherein the fast-convolution filters are built using the Discrete Fourier Transform or the Fast Fourier Transform.

SUBSTITUTE SHEET

13. The method of claim 11 wherein the fast-convolution filters are built using the Modified Discrete Fourier Transform as claimed in claim 33.

14. The method of claim 9 wherein a group or groups of more than one of the component filters are implemented using fast convolution techniques with the component filters in each group using a transform of the same length, and wherein only one transform operation is performed for each group, so that the component filters in each group share the transformed output from the respective transform operation.

15. The method of claim 9 wherein a group or groups of more than one of the component filters are implemented using fast convolution techniques with the component filters in each group using a transform of the same length, and wherein only one inverse transform operation is performed for each group, so that the output from the component filters in each group is summed before being passed to the respective inverse transform operation.

16. The method of claim 9 wherein a group or groups of more than one of the component filters are implemented using fast convolution techniques with the component filters in each group using a transform of the same length, and wherein only one transform operation is performed for each group, so that the component filters in each group share the transformed output from the respective transform operation; wherein a group or groups of more than one of the component filters are implemented using fast convolution techniques with the component filters in each group using a transform of the same

length, and wherein only one inverse transform operation is performed for each group, so that the output from the component filters in each group is summed before being passed to the respective inverse transform operation.

17. A digital filter for filtering delayed, overlapped groupings of consecutive samples a, b, c, d..... said filter comprising a transform processor of length m, and N filter processors, each of length k, an adder adapted to sum the outputs as they are fed in parallel from said N filter processors, an inverse transform processor of length m: said consecutive samples a, b, c, d, ..... being fed in blocks of length m, each of said blocks being passed through said transform processor and then through each of said N filter processors with a delay of zero before being passed through a first filter processor of said N filter processors, a delay of k before being passed through a second filter processor and so on up to a delay of (N-1)k before being passed through the Nth filter processor; whereby a filter of effective length Nk is effected with a latency corresponding to that of a conventional filter of length k.

18. The filter of claim 17 wherein said transform processor is a form of Fourier Transform Processor and said inverse transform processor is a form of inverse Fourier Transform Processor and said filter function processor is effected by a multiply operation of transformed input data with frequency response corresponding to a selected portion of a desired impulse response for the entire filter.

19. A method of implementing a filter with a relatively high length/latency efficiency K; said method comprising transforming progressive, consecutive and overlapping portions of input data into the frequency domain, performing a mathematical operation on individual ones of said transformed signals, superposing (by addition) the consecutive signals resulting from said mathematical operations, inverse transforming the resultant signal from the frequency domain back to the time domain and outputting said signal.

20. A method of implementing a filter with a relatively high length/latency efficiency K; said method comprising applying a mathematical transform to progressive, consecutive and overlapping portions of input data so as to produce corresponding transformed data; performing a mathematical operation on individual ones (i.e. two or more) of said transformed data; superposing (by addition) the data resulting from said mathematical operations so as to produce resultant data; applying an inverse mathematical transform to said resultant data so as to produce filtered output data.

21. The method of claim 20 wherein said transform is a form of Fourier Transform and said inverse transform is a form of inverse Fourier Transform and said mathematical operation is a vector multiply of the Fourier transformed input signal segment with the frequency responses of segments of the desired (time domain) filter characteristic.

22. The method of claim 21 wherein the underlying operation is an overlap operation on successive, overlapping portions

of input data and a discard or add operation on succession overlapping blocks of output data (commonly referred to as an overlap/save or overlap/add method).

23. The method of claim 20 wherein said method is utilised to implement at least some of the filter modules of a composite electrical filter; said composite electrical filter comprising a plurality of filter modules arranged to receive in parallel an incoming input signal for filtering; the output from each of said filter modules being summed to produce a composite filtered output signal; each of said filter modules adapted to have an impulse response that is a selected portion of the impulse response of the composite filter.

24. The method of claim 23 including the step of minimising the overlap of the selected portions of impulse response, or to make them not overlap at all.

25. The method of claim 24 wherein the length of each of said filter modules is different with the characteristic of the shorter length filter modules adapted to process first (or earlier) portions of said impulse response and longer length filter modules adapted to process following (or later) portions of said impulse response.

26. The method of claim 25 wherein the shortest module of said plurality of filter modules is a time-domain (low latency) filter whilst additional ones of said filter modules are longer fast-convolution (longer latency) filters.

27. A composite electrical filter comprising a plurality of filter modules arranged to receive in parallel an incoming

SUBSTITUTE SHEET

input signal for filtering; the output from each of said filter modules being summed to produce a composite filtered output signal; each of said filter modules adapted to have an impulse response that is a selected portion of the impulse response of the composite filter.

28. The filter of claim 27 adapted to minimise the overlap of the selected portions of impulse response, or to make them not overlap at all.

29. The filter of claim 28 wherein the length of each of said filter modules is different with the characteristic of the shorter length filter modules adapted to process first (or earlier) portions of said impulse response and longer length filter modules adapted to process following (or later) portions of said impulse response.

30. The filter of claim 29 wherein the shortest module of said plurality of filter modules is a time-domain (low latency) filter whilst additional ones of said filter modules are longer fast-convolution (longer latency) filters.

31. The filter of claim 27 where the number of said filter modules is N comprising filters $F_1$, $F_2 \ldots F_N$ and wherein filter module $F_1$ is a filter with very low latency implemented with time domain techniques whilst all other filter modules $F_i$ are implemented with fast convolution techniques and these fast convolution filters $F_i$ are composed of a sequence of filters each with longer filter length than its predecessor and hence each with longer latency, but still preserving the property that $d_{i+1}=d_i+l_i$ whereby it is ensured that the composite

filter F output formed by summing together the N component filter outputs has an impulse response without any "holes" in it.

32. A method of filtering sampled data so as to achieve a relatively long length but short latency filtering of said data, said method comprising passing said data in parallel through a plurality of sub-filters and summing the output samples from all of said sub-filters to produce filtered sampled data; and wherein the Impulse Response of each of said sub-filters is a selected portion of the desired Impulse Response of the filter characteristic required to produce said filtered sampled data from said sampled data and wherein each said selected portion is selected for each of said sub-filters such that the output from all of said sub-filters, when summed, behaves as if filtered through a filter having said desired Impulse Response.

33. A fourier transform processor adapted to efficiently transform strings of real numbers; said processor operating according to the following method:

1. Take the input vector x(k) of length N, where each element of x(k) is real.

2. Create the vector x'($l$), a complex vector of length N/2 by application of the following:

$$x'(l) = [x(l) - jx(l+\frac{N}{2})]e^{\frac{-\pi jl}{N}} \quad for\ 0 \leq l < N/2 \qquad (9)$$

3. Compute the N/2 point DFT of x'($l$) to give the N/2 complex result vector Z(p):

$$Z(p) = DFT_{(N/2)}[x'(l)] \qquad (10)$$

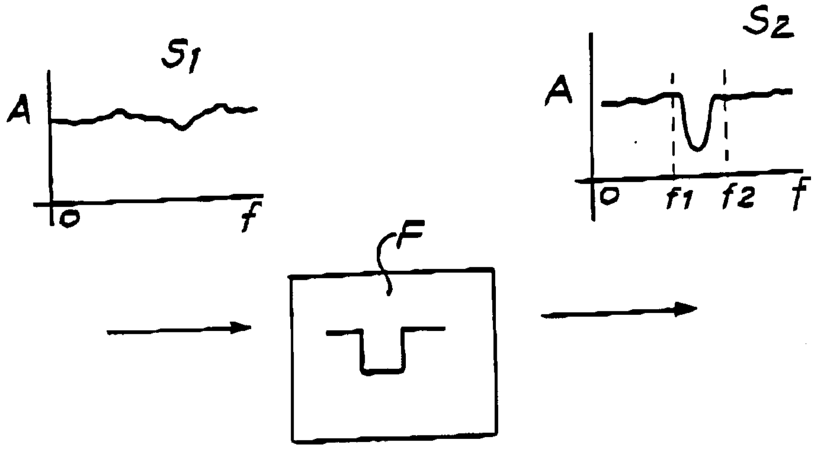


FIG. 1

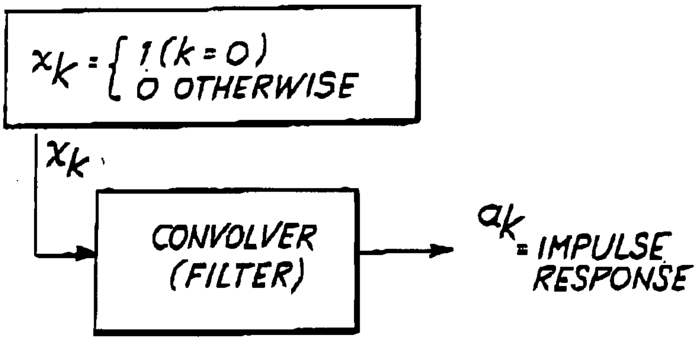

$x_k = \begin{cases} 1 (k=0) \\ 0 \text{ OTHERWISE} \end{cases}$
$x_k$
CONVOLVER (FILTER)
$a_k$ = IMPULSE RESPONSE

FIG. 2

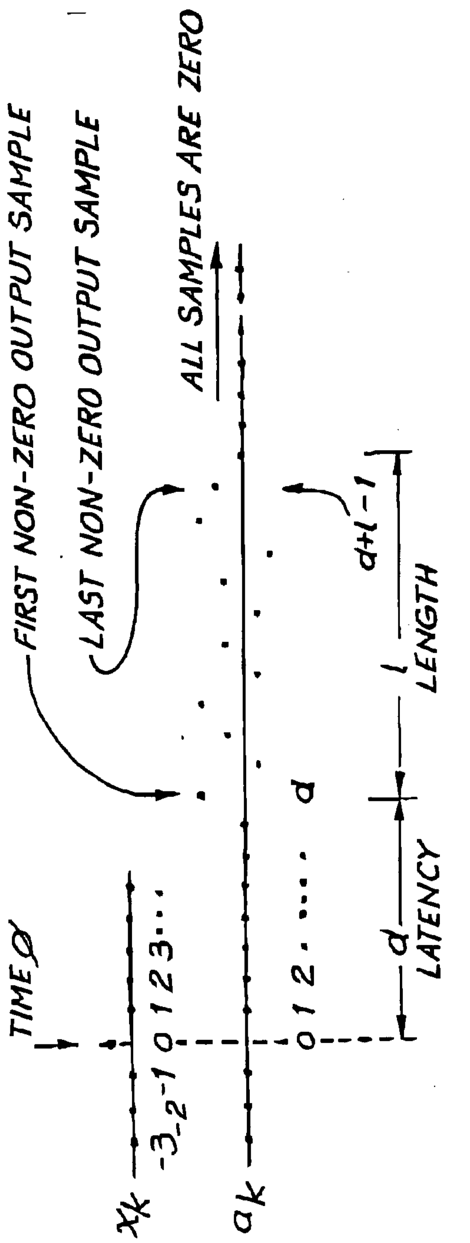


FIG. 3

SUBSTITUTE SHEET

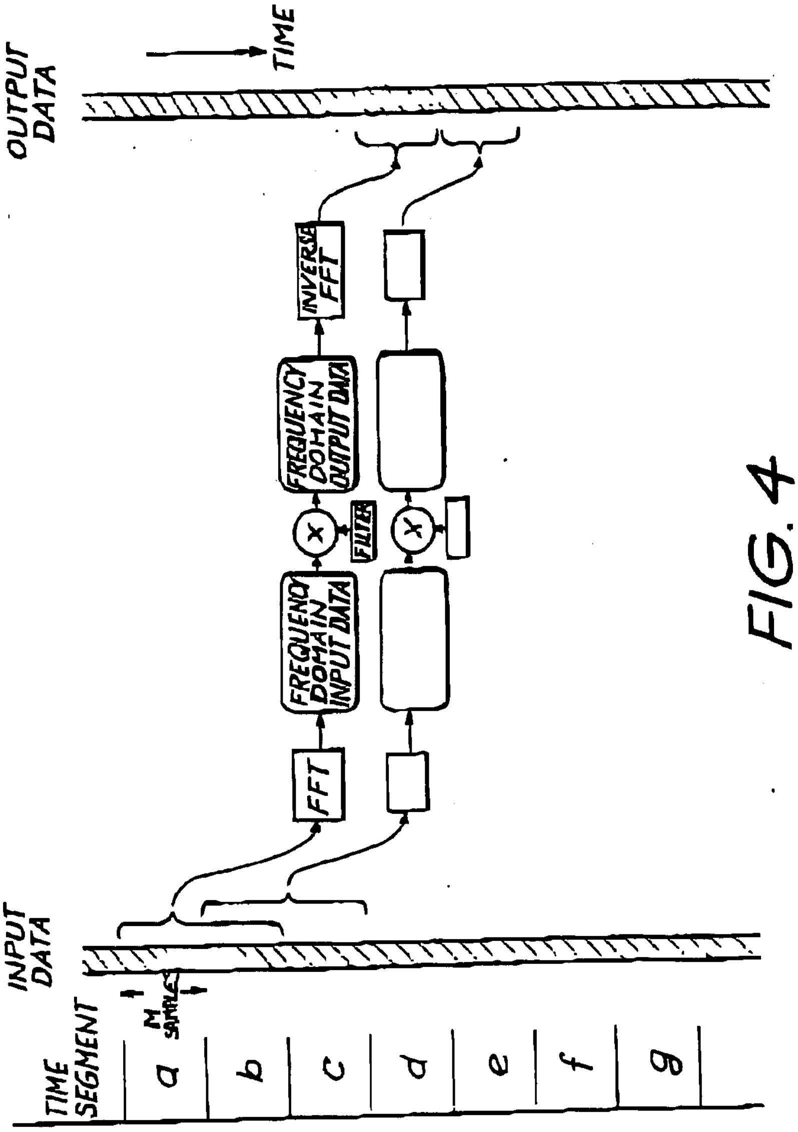


FIG. 4

SUBSTITUTE SHEET

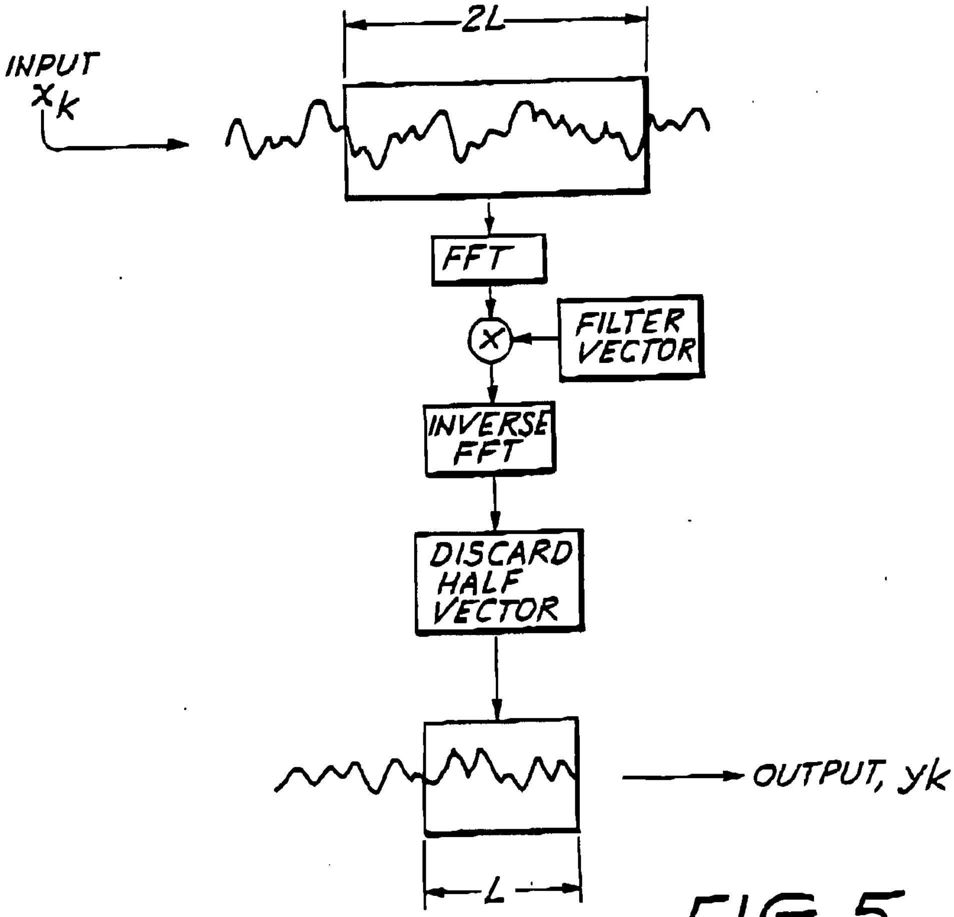

2L
INPUT
$x_k$
FFT
FILTER VECTOR
INVERSE FFT
DISCARD HALF VECTOR
OUTPUT, yk
L

FIG. 5

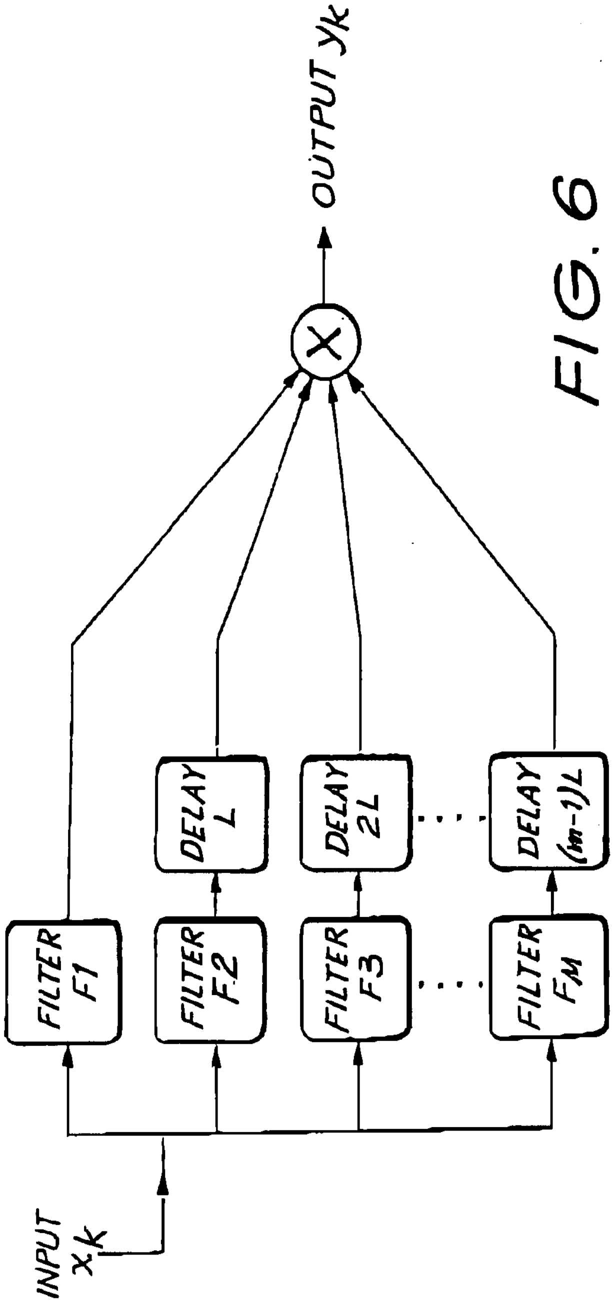

OUTPUT $y_k$
×
FILTER F1
FILTER F2
FILTER F3
FILTER FM
DELAY L
DELAY 2L
DELAY (m-1)L
INPUT $x_k$

FIG. 6

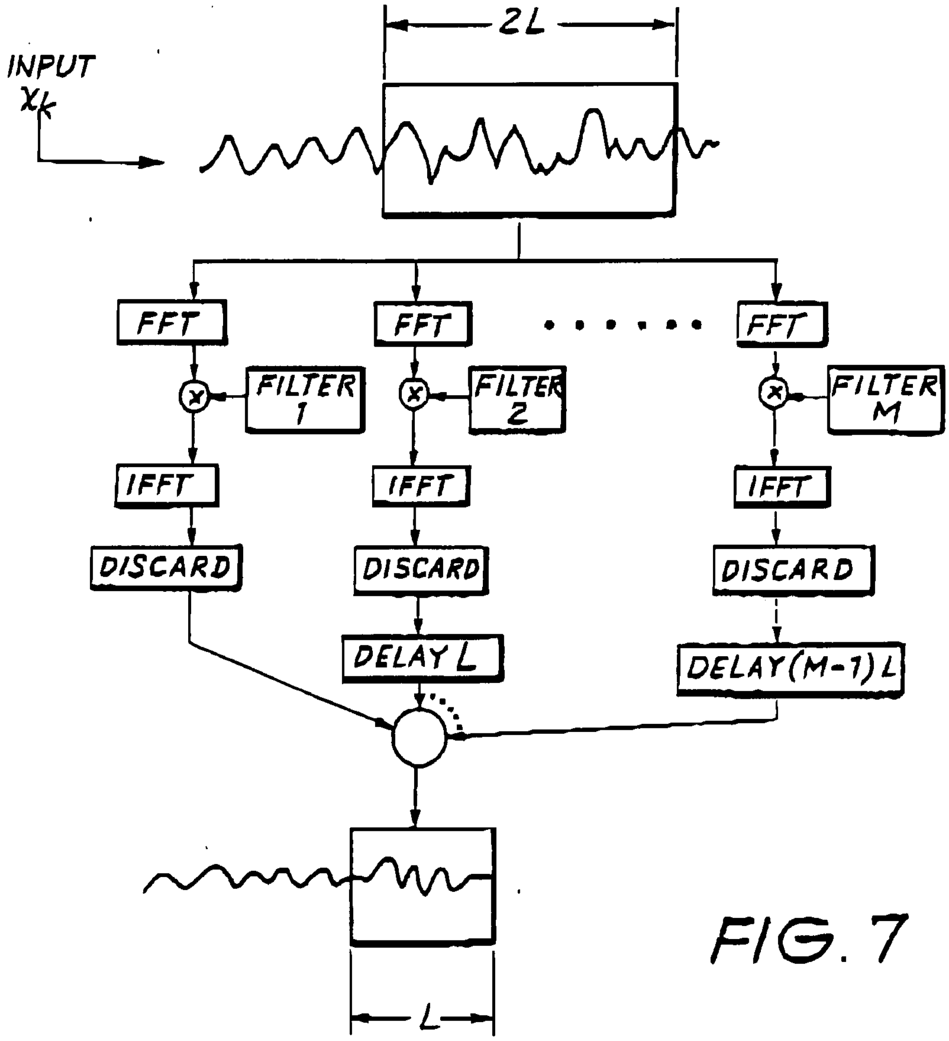

2L
INPUT
$x_k$
FFT
FFT
FFT
FILTER 1
FILTER 2
FILTER M
IFFT
IFFT
IFFT
DISCARD
DISCARD
DISCARD
DELAY L
DELAY (M-1) L
L

FIG. 7

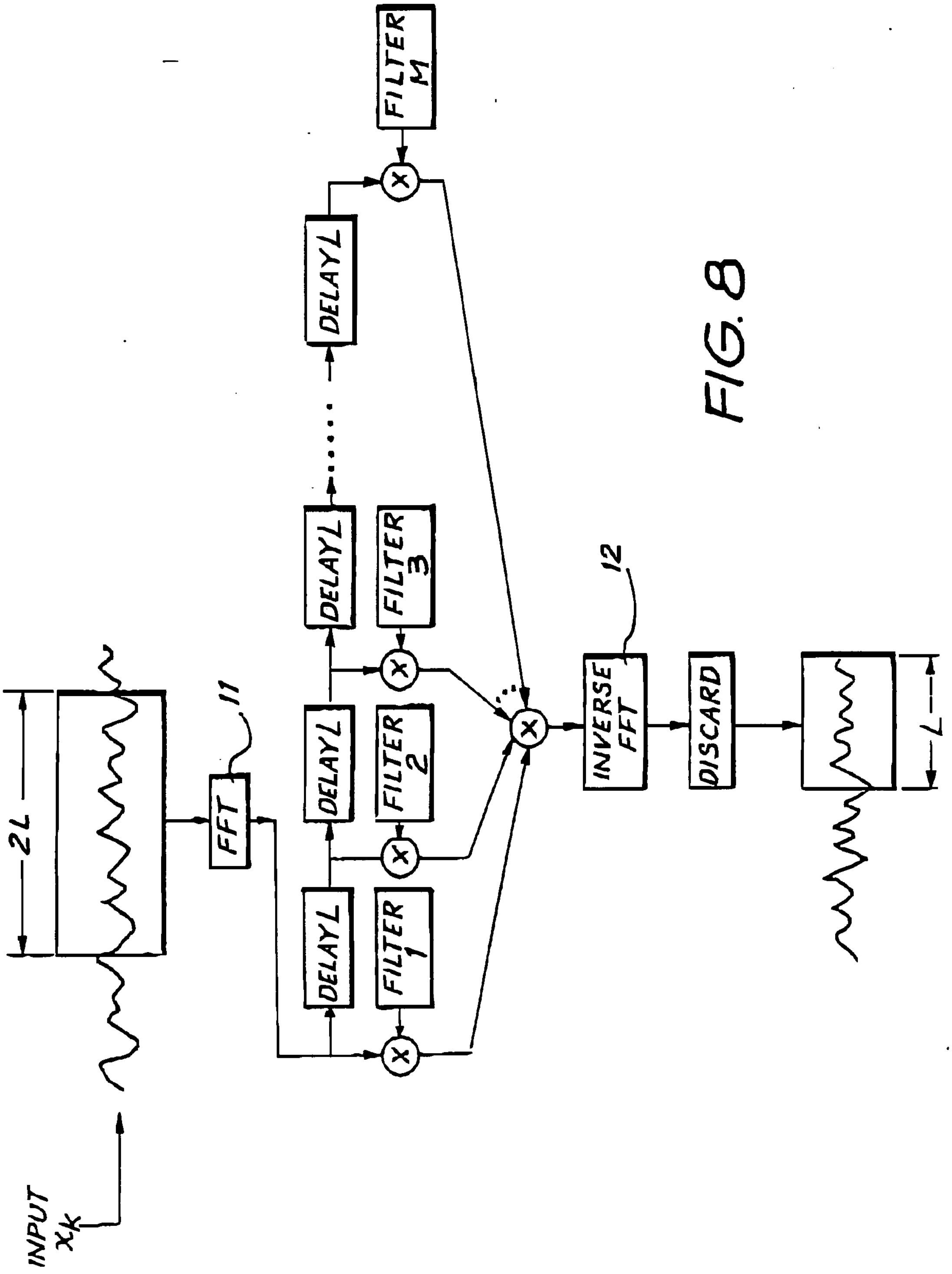

2L
FFT
11
DELAY L
DELAY L
DELAY L
DELAY L
FILTER 1
FILTER 2
FILTER 3
FILTER M
×
×
×
×
×
INVERSE FFT
12
DISCARD
L
INPUT $x_k$

FIG. 8

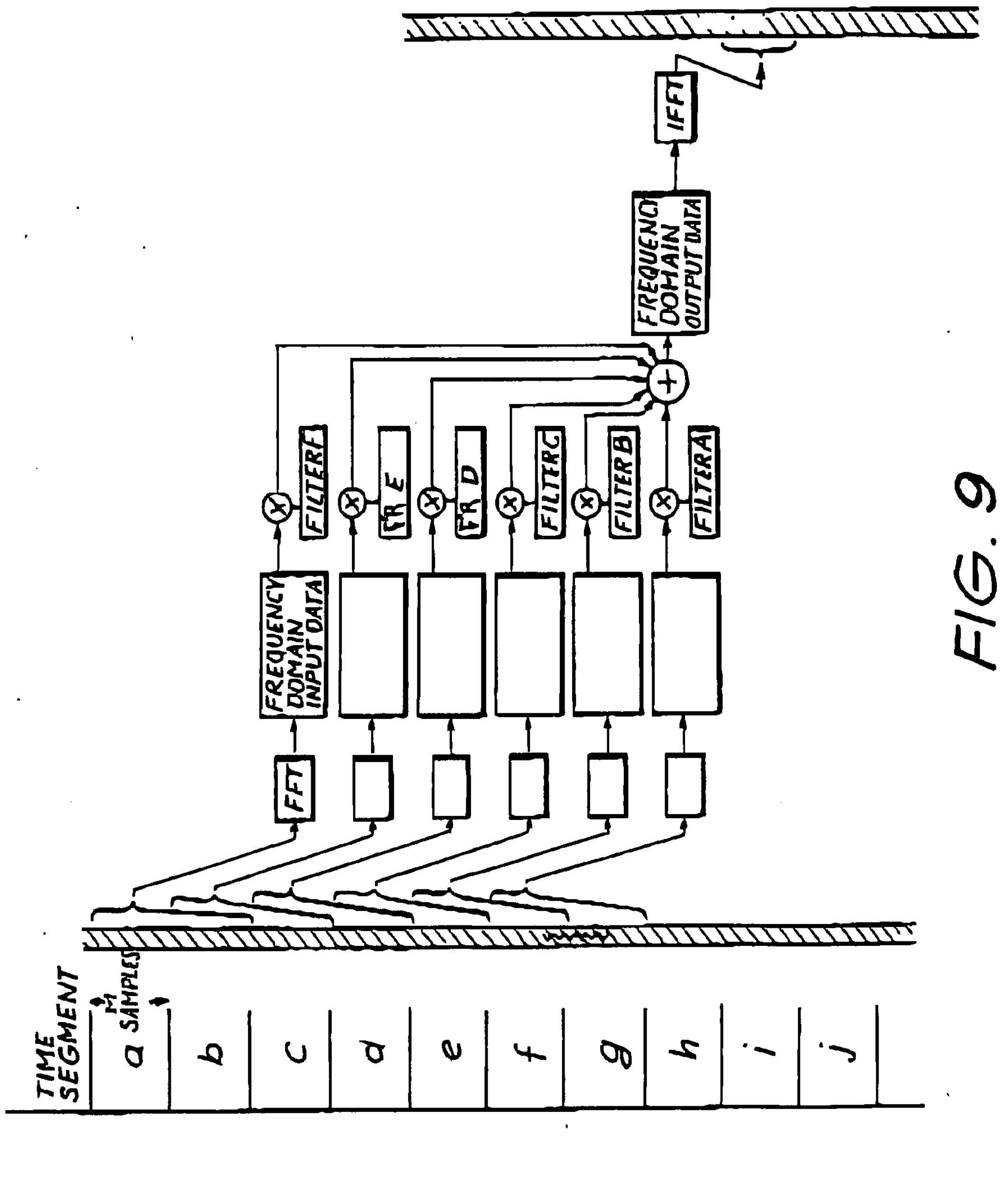


FIG. 9

SUBSTITUTE SHEET

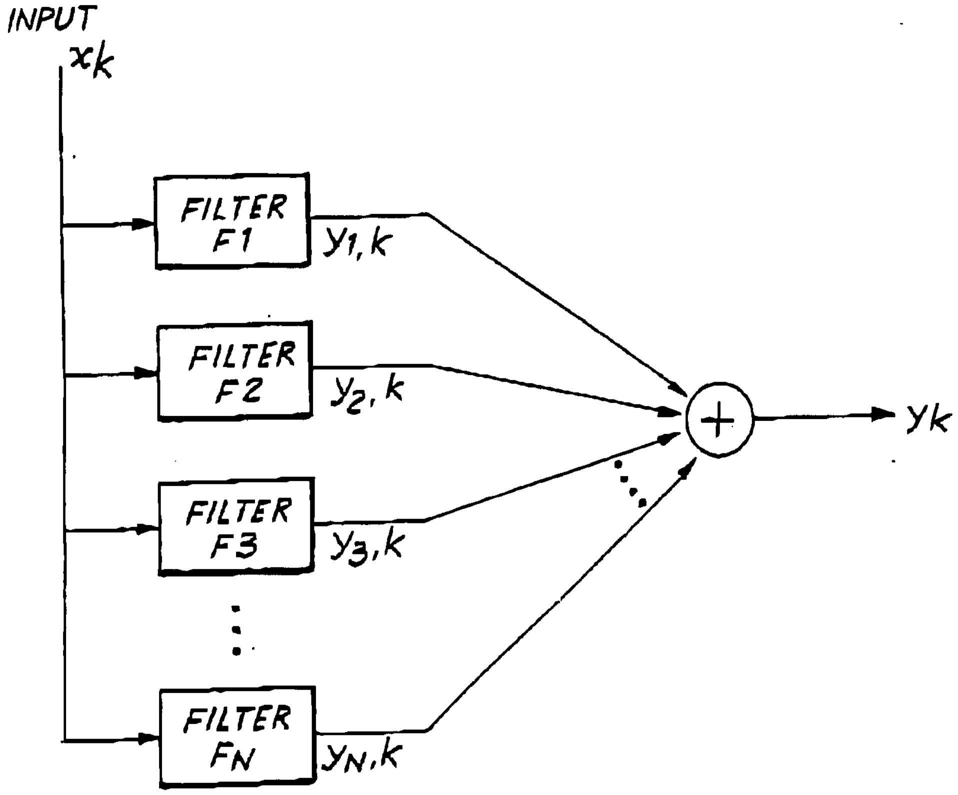

INPUT
$x_k$
FILTER F1
FILTER F2
FILTER F3
FILTER FN
$y_{1,k}$
$y_{2,k}$
$y_{3,k}$
$y_{N,k}$
+
$y_k$

FIG. 10

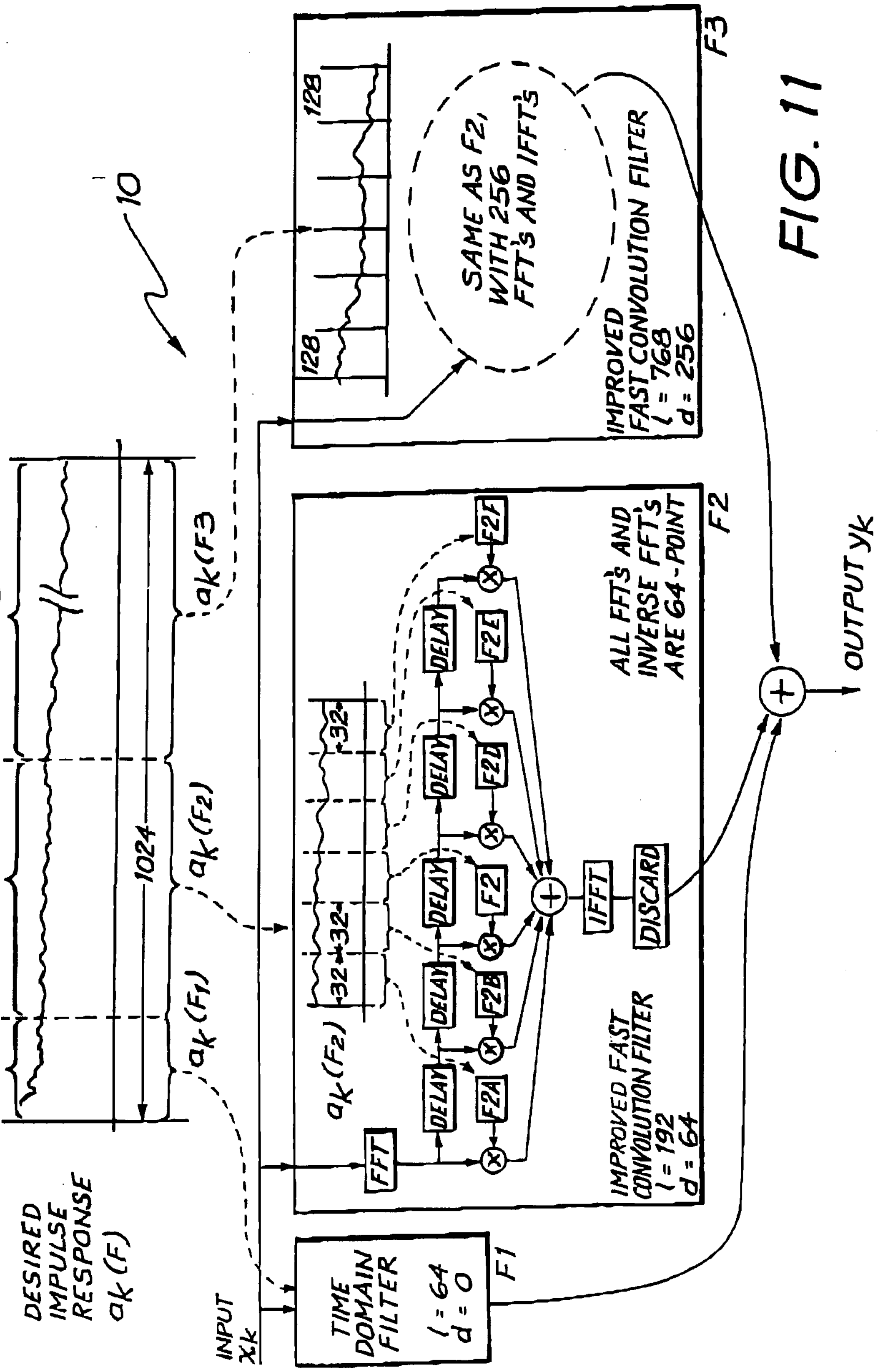


FIG. 11

SUBSTITUTE SHEET

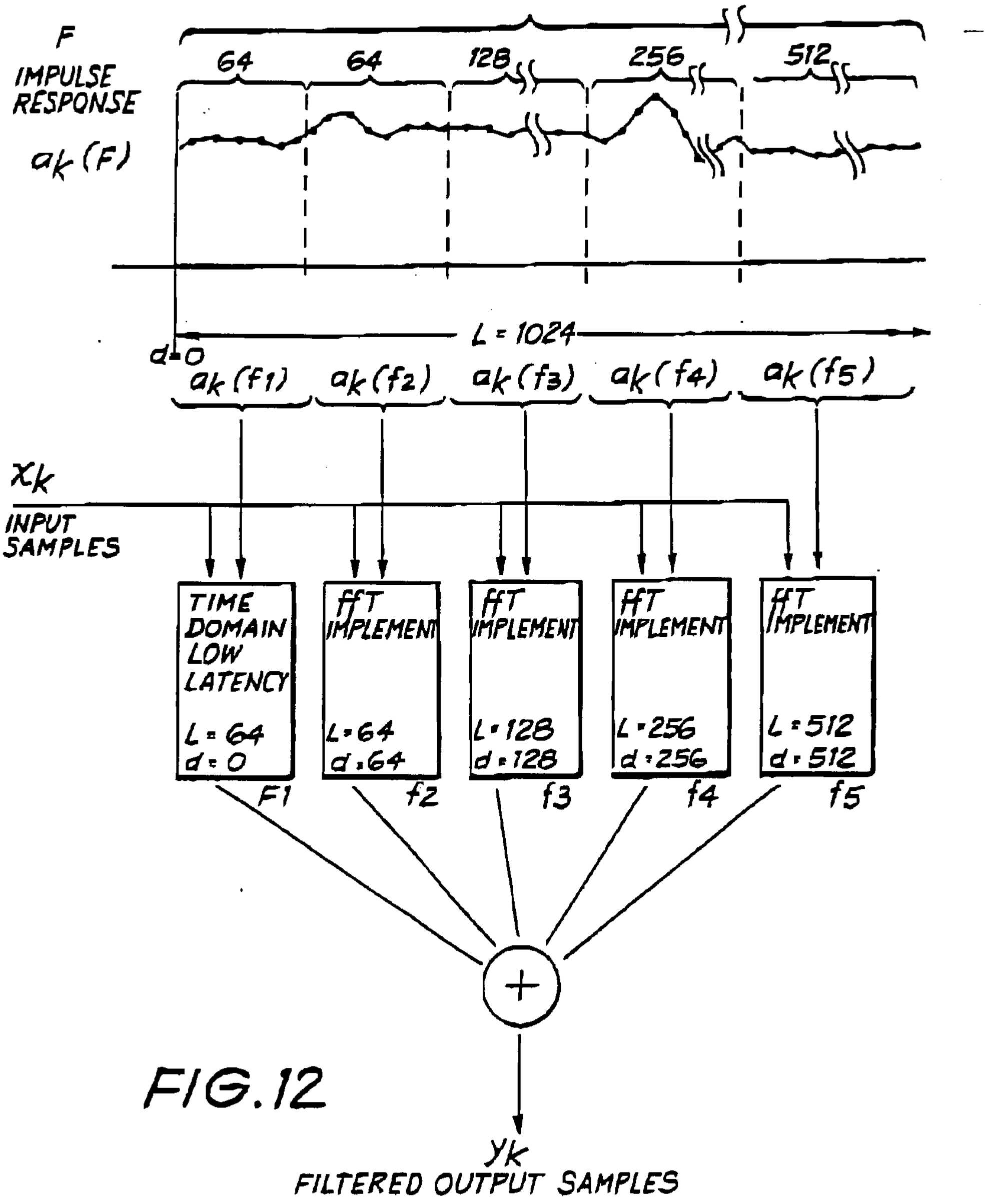
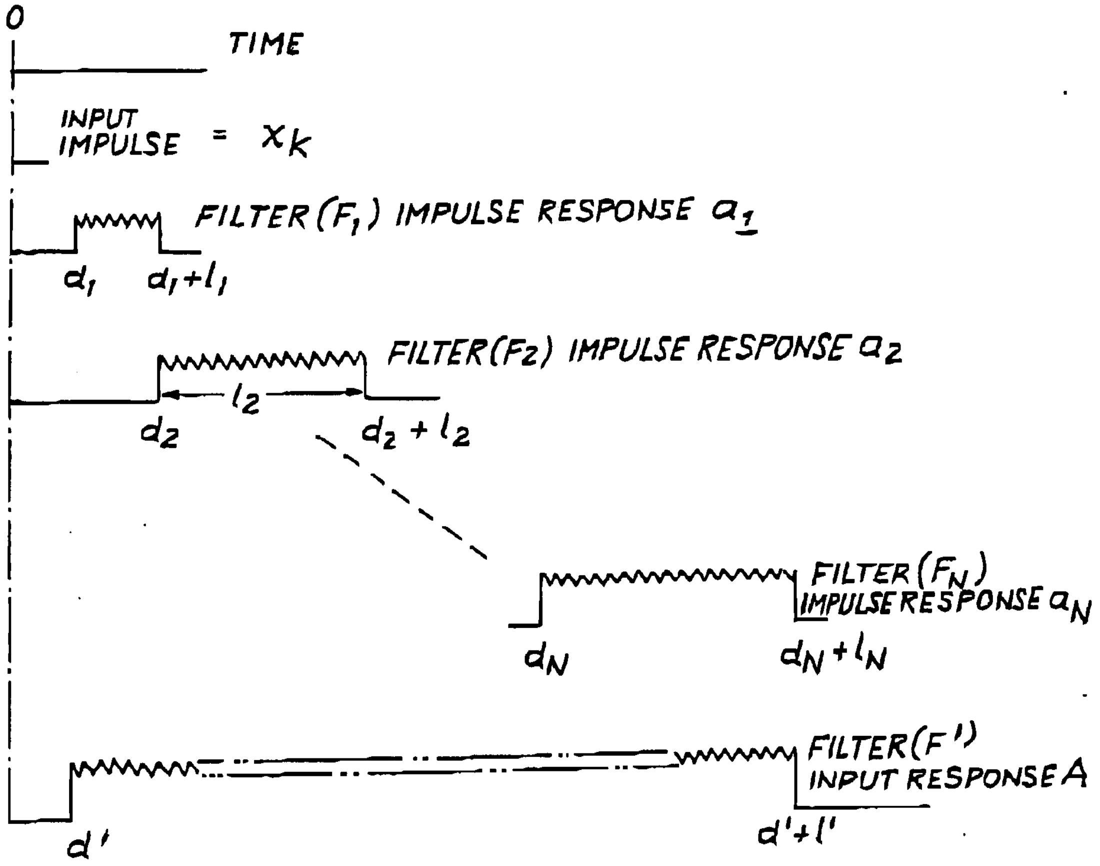


FIG.12

SUBSTITUTE SHEET



0
TIME
INPUT IMPULSE = $x_k$
FILTER ($F_1$) IMPULSE RESPONSE $a_1$
$d_1$
$d_1+l_1$
FILTER($F_2$) IMPULSE RESPONSE $a_2$
$l_2$
$d_2$
$d_2+l_2$
FILTER ($F_N$) IMPULSE RESPONSE $a_N$
$d_N$
$d_N+l_N$
FILTER($F'$) INPUT RESPONSE A
$d'$
$d'+l'$

THE FILTER $F'$ IS MADE BY SUMMING TOGETHER THE OUTPUTS OF FILTERS $F_1, F_2, \ldots\ldots, F_N$.

EACH FILTER $F_i$ DEFINES OUTPUTS $y_{i,k}$ THE OUTPUT OF FILTER $F_i$ AT TIME SAMPLE $k$, AS BEING

$$y_{i,k} = \sum_{n=d_i}^{d_i+l_i-1} a_{i,n}\, x_{k-n}$$

IF WE SET $y'_k = \sum_{n=1}^{N} y_{i,k}$

THEN WE HAVE $y'_k = \sum_{n=d'}^{d'+l'-1} A_n\, x_{k-n}$

FIG. 13

SUBSTITUTE SHEET

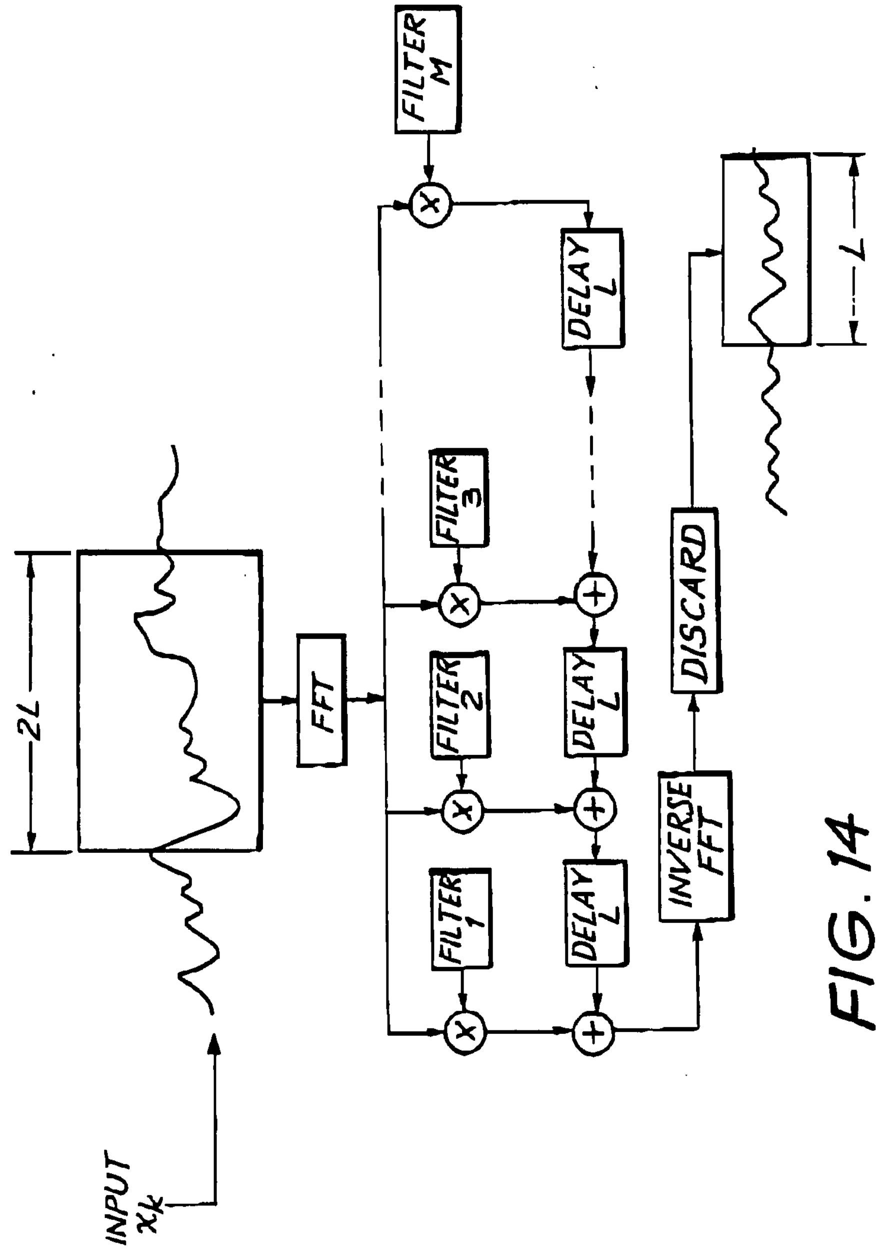

INPUT $x_k$
2L
FFT
FILTER 1
FILTER 2
FILTER 3
FILTER M
DELAY L
DELAY L
DELAY L
INVERSE FFT
DISCARD
L

FIG. 14

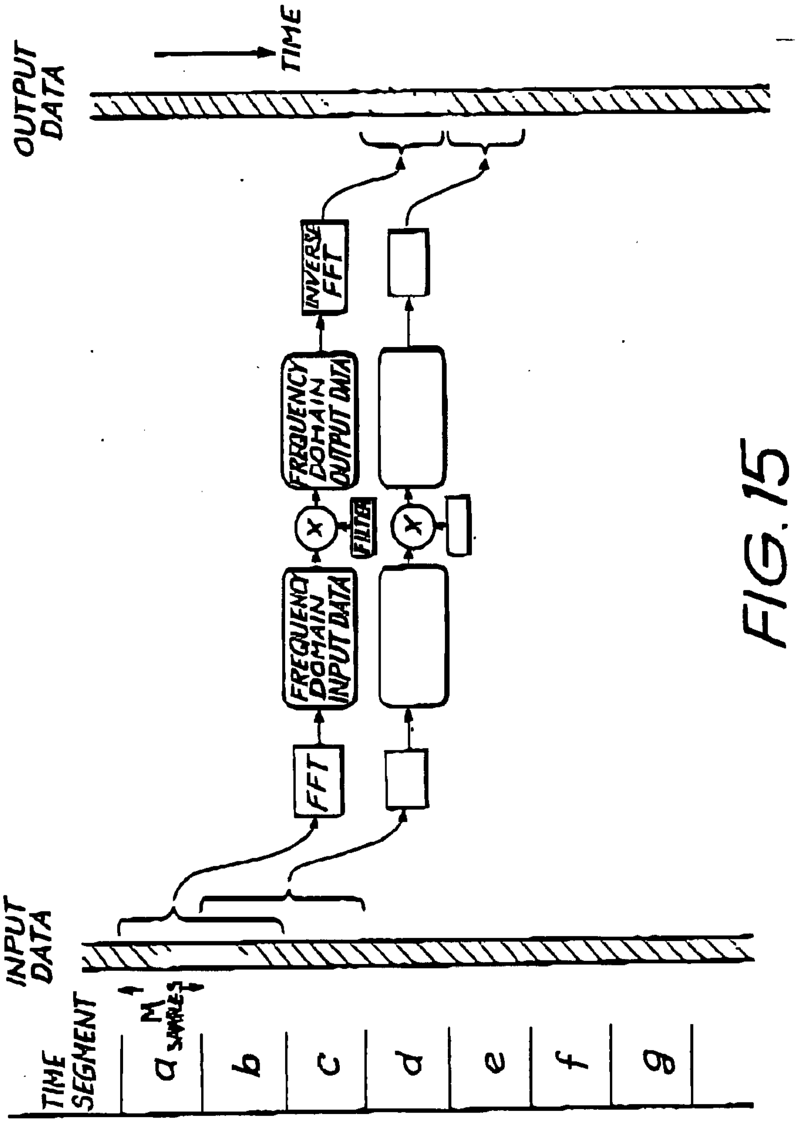

OUTPUT DATA
TIME
INVERSE FFT
FREQUENCY DOMAIN OUTPUT DATA
x
FILTER
x
FREQUENCY DOMAIN INPUT DATA
FFT
INPUT DATA
M SAMPLES
TIME SEGMENT
a
b
c
d
e
f
g

FIG.15

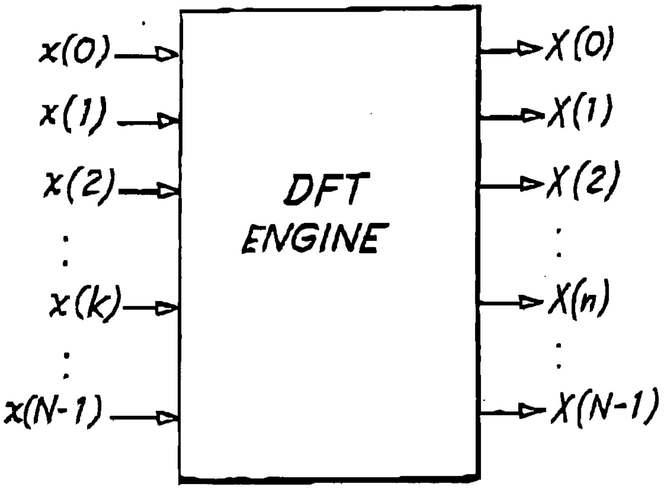


FIG. 16

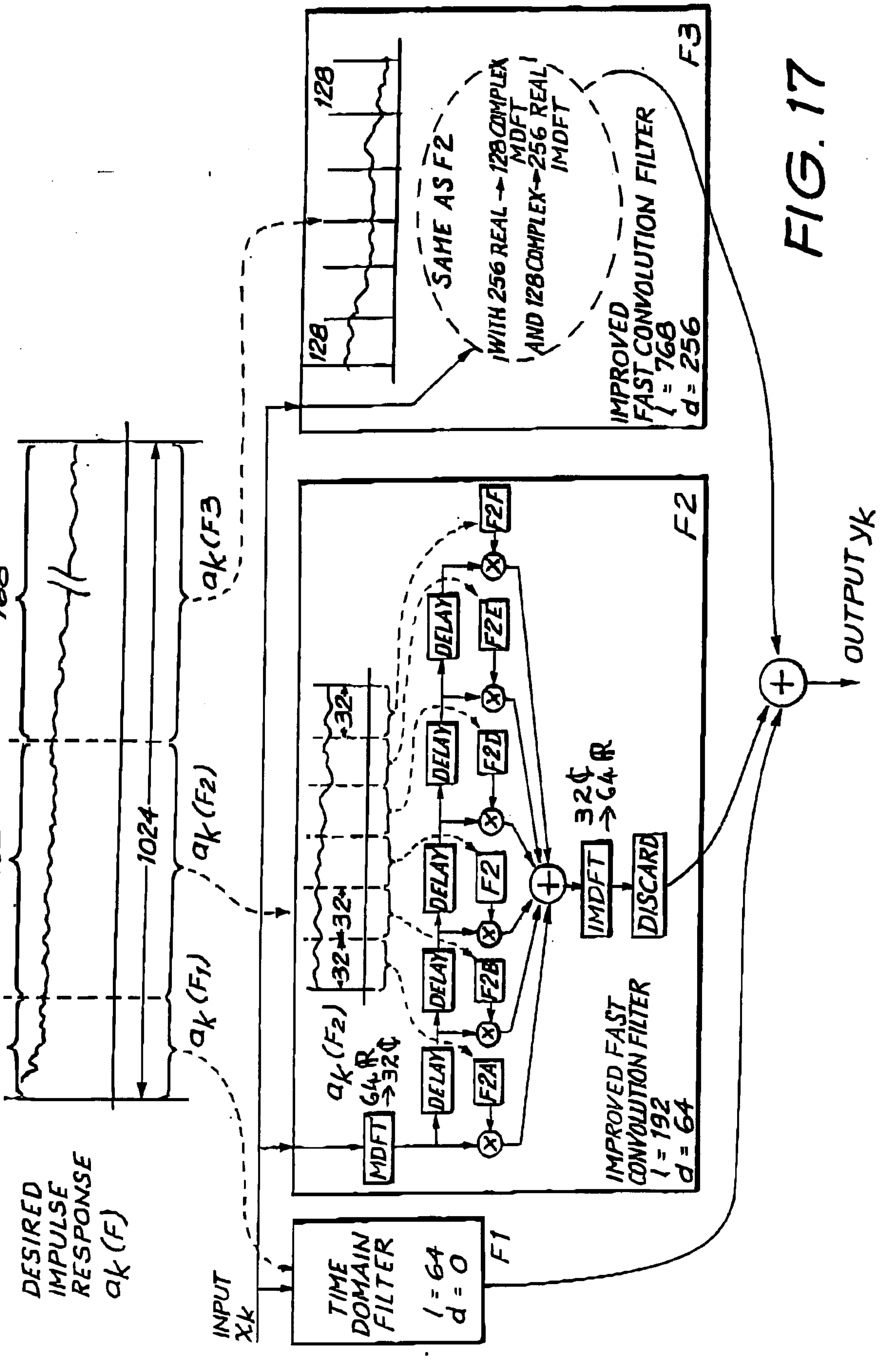

DESIRED IMPULSE RESPONSE ak(F)
64
192
768
1024
ak(F1)
ak(F2)
ak(F3)
INPUT xk
TIME DOMAIN FILTER
l = 64
d = 0
F1
MDFT
ak(F2)
64 R → 32 C
DELAY
F2A
F2B
F2
F2D
F2E
F2F
32
32 C → 64 R
IMDFT
DISCARD
IMPROVED FAST CONVOLUTION FILTER
l = 192
d = 64
F2
128
SAME AS F2
WITH 256 REAL → 128 COMPLEX MDFT
AND 128 COMPLEX → 256 REAL IMDFT
IMPROVED FAST CONVOLUTION FILTER
l = 768
d = 256
F3
OUTPUT yk

FIG. 17

# INTERNATIONAL SEARCH REPORT

International application No.
PCT/AU 93/00330

A. CLASSIFICATION OF SUBJECT MATTER
Int. Cl.$^5$ H03H 17/6, 17/2, G06F 15/332

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC: H03H 17/6, 17/2, 7/28, G06F 15/332, 15/33

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
AU: IPC as above

Electronic data base consulted during the international search (name of data base, and where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

| Category* | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to Claim No. |
|---|---|---|
| | US,A,4992967 (AUVRAY) 12 February 1991 (12.02.91) | |
| X | Claims 1-3, Figs. 9-10 | 19-25,27-28 |
| Y | Whole document | 1-2,6-10,14-18 |
| | EP,A,250048 (N.V. PHILLIPS' GLOELLAMPENFABRIEKEN GROENEWOUDSEWEG) 23 December 1987 (23.12.87) | |
| X | Page 4 lines 1-20, Fig. 1 | 19-25,27-28 |
| | EP,A,448758 (STANDARD ELEKTRIK LORENZ AKTIENGESELLSCHAFT) 2 October 1991 (02.10.91) | |
| X | Claim 1 | 27,28 |

☒ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

| | |
|---|---|
| Date of the actual completion of the international search<br>17 September 1993 (17.09.93) | Date of mailing of the international search report<br>30 SEP 1993 (30.09.93) |
| Name and mailing address of the ISA/AU<br>AUSTRALIAN INDUSTRIAL PROPERTY ORGANISATION<br>PO BOX 200<br>WODEN ACT 2606<br>AUSTRALIA<br>Facsimile No. (06) 2853929 | Authorized officer<br>R. CHIA<br>Telephone No. (06) 2832185 |

Form PCT/ISA/210 (continuation of first sheet (2)) (July 1992) copljm

INTERNATIONAL SEARCH REPORT

International application No.
PCT/AU 93/00330

| C(Continuation). | DOCUMENTS CONSIDERED TO BE RELEVANT | |
|---|---|---|
| Category* | Citation of document, with indication, where appropriate of the relevant passages | Relevant to Claim No. |
| | WO,A,88/03341 (FUJITSU LIMITED) 5 May 1988 (05.05.88) | |
| X | Page 11 lines 2-30 | 19-25,27-28 |
| Y | Page 14 lines 1-10 | 1-2,6-10,14-16 |
| | US,A,4623980 (VARY) 18 November 1986 (18.11.86) | |
| X | Figs. 1-2, column 3 lines 10-40 | 19-25,27-28 |
| Y | Column 2 lines 59-65 | 17-18 |
| | Proceedings of The IEEE, Vol. 75, No. 9, issued September 1987 (New York), R.C. Agarwal, "Vectorized Mixed Radix Discrete Fourier Transform Algorithms", pages 1283-1292 | |
| X | Whole document | 33 |

**INTERNATIONAL SEARCH REPORT**

International application No.
**PCT/AU 93/00330**

**Box I Observations where certain claims were found unsearchable (Continuation of Item 1 of first sheet)**

This international search report has not established in respect of certain claims under Article 17(2)(a) for the following reasons:

1. □ Claims Nos.:
because they relate to subject matter not required to be searched by this Authority, namely:

2. □ Claim Nos.:
because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out, specifically:

3. □ Claims Nos.:
because they are dependent claims and are not drafted in accordance with the second and third sentences of Rule 6.4(a).

**Box II Observations where unity of invention is lacking (Continuation of Item 2 of first sheet)**

This International Searching Authority found multiple inventions in this international application, as follows:

1. Claims 1-32
Apparatus for and method of performing digital filtering,

2. Claim 33
Fourier transform processor for transforming strings of real numbers, as reasoned on the extra sheet.

(See Box II continued)

1. □ As all required additional search fees were timely paid by the applicant, this international search report covers all searchable claims

2. ☒ As all searchable claims could be searched without effort justifying an additional fee, this Authority did not invite payment of any additional fee.

3. □ As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims for which fees were paid, specifically claims Nos.:

4. □ No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.:

**Remark on Protest**

□ The additional search fees were accompanied by the applicant's protest.

□ No protest accompanied the payment of additional search fees.

**INTERNATIONAL SEARCH REPORT**

International application No.

PCT/AU 93/00330

(continuation) BOX II

The international application does not comply with the requirements of unity of invention because it does not relate to one invention or to a group of inventions so linked as to form a single general inventive concept.

In coming to this conclusion the International Searching Authority has found that there are two inventions:

1. Claims 1-32 directed to digital filtering of sampled data by operating a number of component filters in parallel, each associated with a different delay. It is considered that the combination of the outputs of the parallel filter components by addition comprises a first "special technical feature".

2. Claim 32 directed to a Fourier transform processor for transforming string of real numbers. It is considered that the application of a particular algorithm to create a complex vector from a series of input data comprises a second separate "special technical feature".

Since the abovementioned groups of claims do not share either of the technical features identified, a "technical relationship" between the inventions, as defined in PCT Rule 13.2 does not exist. Accordingly the international application does not relate to one invention or to a single inventive concept.

INTERNATIONAL SEARCH REPORT
Information on patent family memb

International application No.
PCT/AU 93/00330

This Annex lists the known "A" publication level patent family members relating to the patent documents cited in the above-mentioned international search report. The Australian Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

| Patent Document Cited in Search Report | | Patent Family Member | | | | | |
|---|---|---|---|---|---|---|---|
| US | 4992967 | EP | 312463 | FR | 2622069 | | |
| EP | 250048 | AU<br>NL | 74466/87<br>8601604 | CA<br>US | 1266893<br>4807173 | JP | 63004710 |
| EP | 448758 | US | 5159565 | | | | |
| WO | 88/03341 | EP | 288577 | US | 4951269 | | |
| US | 4623980 | CA<br>JP | 1201178<br>58030219 | DE | 3118473 | EP | 65210 |

END OF ANNEX

PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

A3

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

| | | |
|---|---|---|
| (51) International Patent Classification 4 :<br>H04B 3/23, H03H 21/00<br>G06F 15/332 | A1 | (11) International Publication Number: WO 88/ 03341<br>(43) International Publication Date: 5 May 1988 (05.05.88) |

(21) International Application Number: PCT/JP87/00833

(22) International Filing Date: 29 October 1987 (29.10.87)

(31) Priority Application Numbers: 61/256750
61/265470

(32) Priority Dates: 30 October 1986 (30.10.86)
10 November 1986 (10.11.86)

(33) Priority Country: JP

(71) Applicant *(for all designated States except US)*: FUJITSU LIMITED [JP/JP]; 1015, Kamikodanaka, Nakahara-ku, Kawasaki-shi, Kanagawa 211 (JP).

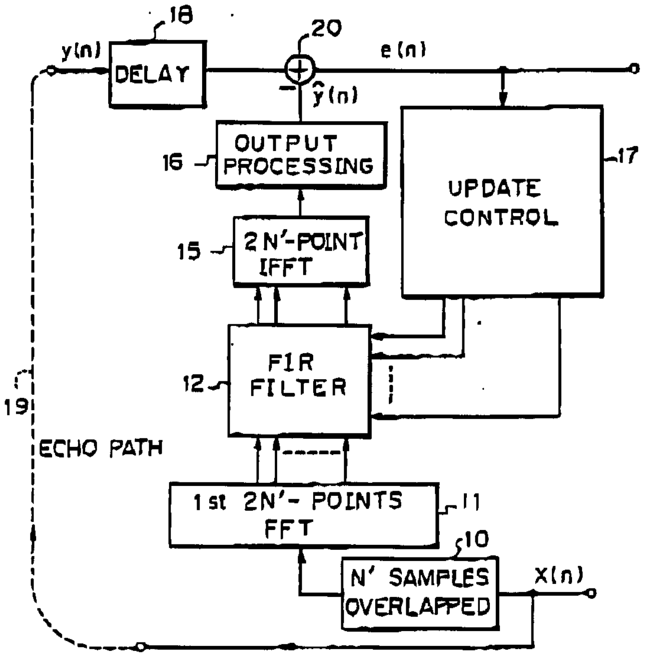
(72) Inventors; and
(75) Inventors/Applicants *(for US only)* : AMANO, Fumio [JP/JP]; 1-17-1, Chitosedai, Setagaya-ku, Tokyo 157 (JP). ASHARIF, Mohammad, Reza [IR/JP]; Kopo Tamagawa 403, 1065-5, Shimonoge, Takatsu-ku, Kawasaki-shi, Kanagawa 213 (JP). UNAGAMI, Shigeyuki [JP/JP]; 34-10, Morinosato 2-chome, Atsugi-shi, Kanagawa 243-01 (JP). SAKAI, Yoshihiro [JP/JP]; Ito-so, 1-671, Kosugijinya-cho, Nakahara-ku, Kawasaki-shi, Kanagawa 211 (JP).

(74) Agents: AOKI, Akira et al.: Seiko Toranomon Bldg., 8-10, Toranomon 1-chome, Minato-ku, Tokyo 105 (JP).

(81) Designated States: AT (European patent), BE (European patent), CH (European patent), DE (European patent), FR (European patent), GB (European patent), IT (European patent), JP, LU (European patent), NL (European patent), SE (European patent), US.

**Published**
*With international search report.*
*Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.*

(54) Title: ECHO CANCELLER WITH SHORT PROCESSING DELAY AND DECREASED MULTIPLICATION NUMBER



(57) Abstract

An echo canceller in a system having a long impulse response such as in an acoustic system, employing a fast Fourier transform advantageously used in view of the amount of calculations. To solve the problem of a long delay, the impulse response length is divided into a plurality of blocks. On the decreased number of samples in each block, a fast Fourier transform and finite impulse response type digital filtering are effected so that the processing delay is decreased while the amount of calculations is kept small.

***FOR THE PURPOSES OF INFORMATION ONLY***

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

| | |
|---|---|
| AT | Austria |
| AU | Australia |
| BB | Barbados |
| BE | Belgium |
| BG | Bulgaria |
| BJ | Benin |
| BR | Brazil |
| CF | Central African Republic |
| CG | Congo |
| CH | Switzerland |
| CM | Cameroon |
| DE | Germany, Federal Republic of |
| DK | Denmark |
| FI | Finland |
| FR | France |
| GA | Gabon |
| GB | United Kingdom |
| HU | Hungary |
| IT | Italy |
| JP | Japan |
| KP | Democratic People's Republic of Korea |
| KR | Republic of Korea |
| LI | Liechtenstein |
| LK | Sri Lanka |
| LU | Luxembourg |
| MC | Monaco |
| MG | Madagascar |
| ML | Mali |
| MR | Mauritania |
| MW | Malawi |
| NL | Netherlands |
| NO | Norway |
| RO | Romania |
| SD | Sudan |
| SE | Sweden |
| SN | Senegal |
| SU | Soviet Union |
| TD | Chad |
| TG | Togo |
| US | United States of America |

## DESCRIPTION

### TITLE OF THE INVENTION

Echo Canceller with Short Processing Delay and Decreased Multiplication Number

### TECHNICAL FIELD

The present invention relates to an echo canceller with a short processing delay and a decreased number of multiplications.

It is well known that the frequency domain adaptive filter (FDAF) is superior to the time domain adaptive filter (TDAF) concerning in the computation number. This is more realizable when the filter length is increased, as in the case of the acoustic noise or acoustic echo cancelling in speech communication systems having a long impulse response. The reduction in the computation number is obtained by replacing the convolution in the time domain by multiplication in the frequency domain, using a fast Fourier transform (FFT).

However, to realize a linear convolution of the time domain in the frequency domain, a circular convolution must be considered, and the FFT used with an overlapped-save method and a block LMS algorithm. If the length of the block data introduced each time to the system is long, then the input-output system propagation delay also will be long, and if the length of the shifted data is short then the propagation delay will be short but the multiplication number will be increased. Note, in the FDAF, the FFT length will be too long when the acoustic impulse response is too long. The impulse response of an acoustic echo path is about several hundred milliseconds.

Accordingly, for the above reasons, a new echo canceller which requires a decreased computation number and a short propagation delay is required.

### BACKGROUND ART

The prior arts will be explained with reference to

Figs. 9 - 14.

An echo canceller in the time domain is, as is well known and as shown in Fig. 9, constructed by a digital filter having a finite impulse response (FIR) type having coefficients $\hat{h}_k$ which are the estimated values of an impulse response $h_k$ of an echo path to be estimated, where $k = 0, 1, \ldots N-1$, and N is ths length of the FIR digital filter.

Each coefficient $\hat{h}_k$ is obtained by an adaptive control in which the difference signal e(n) between an echo signal y(n) passed through the echo path to be estimated and the estimated output $\hat{y}(n)$ of the FIR filter is adaptively controlled to be zero. Known adaptive controls are, a successive adaptive control method and a block adaptive control method. In the successive adaptive control method, the coefficients $\hat{h}_k$ are adaptively updated upon each input of one sample of the input data x(n). In the block adaptive control method, the coefficients are not updated until L samples of the input data have been input, so that the coefficients are updated as a lump at one time when L samples of input data have been received.

In the successive adaptive control method, the necessary multiplication number for each input sample is 2N. That is, N times multiplications are necessary for calculating the output signal $\hat{y}(n)$, and N times multiplications are necessary for updating the coefficients, resulting in the need for 2N times multiplications.

In the block adaptive control method, a fast Fourier transform (FFT) is introduced to effect the processing shown in Fig. 9 in the frequency domain, so that the necessary number of multiplications can be reduced. The FFT by the block adaptive control is well known and is described in, for example, "Fast Implementation of LMS Adaptive Filters" IEEE TRANSACTION ON ACOUSTICS, SPEECH, AND SIGNAL PROCESSING, VOL. ASSP-28, No. 4, AUGUST 1980.

The basic concept thereof is as follows. The convolution expressed by:

$$\hat{y}(n) = \sum_{i=0}^{N-1} h_i \cdot x(n-i) \qquad \ldots (1)$$

can be considered, as shown in Figs. 10B - 10E, as a sum of products of the input signal x(n) (Fig. 10A) and the respective coefficients, where the coefficient series is shifted by one sample for each input signal x(n). In the figure, N represents the length of the impulse response. For L samples after one updating , the coefficients are not updated by the adaptive control.

Now, consider a series of the input samples x(n) with a periodic period (N+L-1), each period including (N+L-1) samples x(n-N+1) through x(n+L-1) of the input data x(n), as shown in Fig. 11A. On the other hand, with respect to the coefficients $\hat{h}_k$ , a series with a periodic period (N+L-1) is obtained by inserting (L-1) "0"s between the coefficients $\hat{h}_0$ and $\hat{h}_{N-1}$. Then, by shifting the coefficients series one sample by one sample, as shown in Figs. 11B - 11F, (N+L-1) series are obtained. Each sample of the input data x(n-N+1) through x(n+L-1) is multiplied to each of the (N+L-1) series of the coefficients shown in Figs. 11B - 11F. Then, the multiplied results are summed to obtain an output series z(n) having a period of (N+L-1). As is apparent from Figs. 10A - 10E and 11A - 11F, the latter half L samples of the series z(n) is equal to the series $\hat{y}(n)$.

In another aspect, the well known cyclic convolution shown in Figs. 11A - 11F is such that the discrete Fourier transform (DFT) of the output is expressed by the products of the DFT of each input (see, for example, "digital signal processing" ...). Therefore, the above-mentioned expression (1) can be realized by a constitution shown in Fig. 12.

Referring to Fig. 12,

$$(H_k^0, H_k^1, \ldots, H_k^{N+L-2}) = \mathrm{FFT}(\underbrace{h_0(k), h_1(k), \ldots, h_{N-1}(k)}_{N}, \underbrace{0, \ldots, 0}_{L-1} \quad \ldots (2)$$

can be written.

That is, an L sample overlap part 121 receives (N-1) samples of one block of input data x(n) and L samples of the previous block of input data, to generate (N+L-1) samples of overlapped elements in which L samples of the input data are overlapped. An (N+L-1)-point FFT 122 effects a fast Fourier transform on the (N+L-1) samples with the overlapped L elements to generate (N+L-1) elements of $X_0(k)$, $X_1(k)$, $X_{(N-1)}(k)$, $X_N(k)$, ..., $X_{(N+L-2)}(k)$. To the outputs $X_0(k)$, $X_1(k)$, ..., estimated coefficients $H_k^0$, $H_k^1$, ..., $H_k^{N+L-2}$ are respectively multiplied, where the estimated coefficients are updated once after introducing each block of data, as described later in more detail. The results of the multiplications are then subject to inverse fast Fourier transform (IFFT) in the (N+L-1)-point IFFT 123. The output $\hat{y}(n)$ is obtained at the outputs of an output processing part 124, by considering only the last half L elements of the outputs of the (N+L-1)-point IFFT 123, or by discarding the first (N-1) elements of the outputs of the IFFT 123. The error signal e(n) is obtained by subtracting the estimated output $\hat{y}(n)$ from the echo signal y(n) transmitted via the echo path. The adaptive control is effected so as to lead the error signal e(n) to be zero.

The update control of the coefficients $H_k^i$ is as follows.

The equation for the update control of the coefficients is expressed as:

$$\begin{bmatrix} h_0((k+1)\cdot L) \\ h_1((k+1)\cdot L) \\ \vdots \\ h_{N-1}((k+1)\cdot L) \end{bmatrix} = \begin{bmatrix} h_0(k\cdot L) \\ h_1(k\cdot L) \\ \vdots \\ h_{N-1}(k\cdot L) \end{bmatrix} + 2\mu \begin{bmatrix} \sum_{\ell=0}^{L-1} e(k\cdot L+\ell)\cdot x(k\cdot L+\ell) \\ \sum_{\ell=0}^{L-1} e(k\cdot L+\ell)\ x(kL+\ell-1) \\ \vdots \\ \sum_{\ell=0}^{L-1} e(k\cdot L+\ell)\ x(kL+\ell-N+1) \end{bmatrix} \quad \ldots (3)$$

The calculation of the second term in the right hand side of the above equation is effected, as shown in Figs. 13A - 13F, as follows. That is, consider a series of the input data x(n) with a period (N+L-1), and consider a series with (N+L-1) period including L samples of error data e(n), e(n+1), ..., e(n+L-1) and (N-1) "0"s padded. Each element of the input data series x(n-N+1), ..., x(n+L-1) is multiplied to the series including the error data and the "0"s.

The series with the (N+L-1) period including error data and "0"s is shifted one element by one element in response to an input data sample so that (N+L-1) series are obtained as shown in Figs. 13B - 13F. Then, each element of the input data series in the (N+L-1) period is multiplied with each of the (N+L-1) series shown in Figs. 13B - 13F, and the sum of the products is obtained from this sum, the preceding N samples are output as the second term of the right hand side of the equation (3).

A theory is known for a periodic correlation function which is obtained by the product of the DFT of one input and the complex conjugate of the DFT of the other input (see, for example, "Theory and Application of Digital Signal Processing" written by R. Rabiner and B. Gold, 6.18, p 403).

Figure 14 is a block diagram of a conventional echo canceller using the block LMS algorithms and including the updating portion of the coefficients utilizing the above mentioned theory. In Fig. 14, 140 is an overlap processing part; 141 is an (N+L-1)-point FFT; 142 is a complex multiplication part; 143 is an (N+L-1) point inverse FFT; 144 is an output processing part for outputting the last L samples; 145 is a zero padding part; 146 is an (N+L-1) point FFT, 147 is a complex multiplication part; 148 is an IFFT; and 149 is an FFT.

Assuming that the response length is N, then the overlap processing part 140 receives an input signal x(n) and makes (N+L-1) samples as one block. Then, by overlapping L samples of one block and the successive block, the overlap processing part 140 outputs (N+L-1) samples of the input data x(n).

The (N+L-1) point FFT 141 receives each block of the input data x(n) to carry out a Fourier transform on each received block so that the input data x(n), which is represented in the time domain, is transformed to the signals $X_0(k)$, $X_1(k)$, ... $X_{(N+L-2)}(k)$, which are represented in the frequency domain.

In the complex multiplication part 142, the coefficients $H_k^0$, $H_k^1$, ..., $H_k^{N+L-2}$ are respectively multiplied to the signals $X_0(k)$, $X_1(k)$, ..., $X_{(N+L-2)}(k)$ in the frequency domain.

The (N+L-1) point IFFT 143 receives the results of the multiplications and transforms the signals from the frequency domain representation to the time domain representation.

The output processing part 144 receives the (N+L-1) samples of the time domain representation and outputs the last L samples thereof. The output signal $\hat{y}(n)$ is subtracted from the echo signal y(n) passed through the acoustic echo path, resulting in an error signal e(n).

The error signal e(n) has L samples. In the zero padding part 145, (N-1) "0"s are padded in the portion preceding the L samples of the error signal e(n).

The (N+L-1) point FFT 146 receives the (N+L-1) samples of the "0"s and error signal e(n) to transform them into signals $E_k^0$, $E_k^1$, ..., $E_k^{(N+L-2)}$ in the frequency domain.

At the complex multiplication part 147, the signals $E_k^0$, $E_k^1$, ..., $E_k^{(N+L-2)}$ and the complex conjugates of the signals $X_0(k)$, $X_1(k)$, $X_{(N+L-2)}k$ are respectively multiplied. The multiplied results are applied to the IFFT 148 so that the signals are transformed from the frequency domain representation to the time domain representation. The preceding N samples of the output of the IFFT 148 are supplied to the FFT 149, while the FFT 149 receives (L-1) samples of "0"s as the last (L-1) samples of the input signal. Thus, at the output of the IFFT 149, updating parts of the tap coefficients $H_k^0$, $H_k^1$, ..., $H_k^{N+L-2}$ are obtained. The tap coefficients are once updated, or in other words, refreshed, after introducing each block of the input data x(n).

The refreshing of the tap coefficients is further described. The tap coefficients are updated at each input of one block. Assuming that the L samples constitute one adaptation block, and assuming that k is an integer, then, as the coefficient values in the period between kL and {KL+(L-1)}, the values of which were updated at the time KL, are used. The next coefficient values at the next time (K+1)L are determined in such a way that the sum of the squares of the errors during the period between KL and {KL+(L-1)} is made minimum. In this case, assuming that the evaluation number is D, then,

$$D = e_{KL}^2 + e_{KL+1}^2 + \cdots + e_{KL+(L-1)}^2 \qquad \ldots (4)$$

can be obtained.

The error $e_{KL+1}$ can be expressed as:

$$e_{KL+\ell} = \sum_{i=0}^{N-1} (h^i - h^i_{KL}) x_{KL+\ell-i} \qquad \ldots (5)$$

where $h^i$ represents an impulse response of the actual system; $h^i_{KL}$ represents an estimated impulse response; and $x_{KL+\ell-i}$ represents a sample of the series of the input data. Whereas, the following equation stands.

$$-\frac{\partial D}{\partial h^i} = 2 \sum_{\ell=0}^{L-1} e_{KL+\ell} x_{KL+\ell-i} \qquad \ldots (6)$$

Therefore, the updating of the coefficients is expressed by

$$h^i_{(K+1)L} = h^i_{KL} + 2\mu \sum_{\ell=0}^{L-1} e_{KL+1} x_{KL+\ell-i} \qquad \ldots (7)$$

where μ is a constant value. Expression of the equation (7) in the vector form is as follows:

$$\begin{bmatrix} h^0_{(K+1)L} \\ h^1_{(K+1)L} \\ \vdots \\ h^{N-1}_{(K+1)L} \end{bmatrix} = \begin{bmatrix} h^0_{KL} \\ h^1_{KL} \\ \vdots \\ h^{N-1}_{KL} \end{bmatrix} + 2\mu \begin{bmatrix} \sum_{\ell=0}^{L-1} e_{KL+\ell} x_{KL+\ell} \\ \sum_{\ell=0}^{L-1} e_{KL+\ell} x_{KL+\ell-1} \\ \vdots \\ \sum_{\ell=0}^{L-1} e_{KL+\ell} x_{KL+\ell-(N-1)} \end{bmatrix} \qquad \ldots (8)$$

In the algorithm utilizing an FFT, assume that:

$$y_{KL+\ell} = \sum_{i=0}^{N-1} h^i_{KL} x_{KL+\ell-i} \qquad \ldots (9)$$

$$\mathbb{H}^T_K = (h^0_{KL}, h^1_{KL}, \ldots, h^{N-1}_K, 0, 0, \ldots) \qquad \ldots (10)$$

$$\mathbb{X}_K^T = (x_{KL-(N-1)}\ ,\ \ldots,\ x_{KL-1}\ ,\ x_{KL}\ ,\ \ldots,\ x_{KL+(L-1)}) \qquad \ldots (11)$$

$$\mathbb{Y}_K^T = (y_{KL}\ ,\ y_{KL+1}\ ,\ y_{KL+2}\ ,\ \ldots,\ y_{KL+(L-1)}) \qquad \ldots (12)$$

Then, the output vector $\mathbb{Y}_K^T$ is obtained by the latter half L samples of the cyclic convolution with a periodic period equal to (N+L-1) of the vectors $\mathbb{H}_K^T$ and $\mathbb{X}_K^T$ expressed by the equations (10) and (11). The FFT of the output of the cyclic convolution is the products of the FFT of each element which is subject to convolution. Therefore, the fundamental structure of the echo canceller shown in Fig. 14 is constructed by the parts 140 - 144. The coefficients $H_K^0$ - $H_K^{(N+L-2)}$ are updated by the outputs of the FFT 149. To update the coefficients, (L-1) zeros are padded in the FFT 149 to the second term in the equation (8), which is thus changed as follows.

$$\begin{bmatrix} \sum_{\ell=0}^{L-1} e_{K,L+\ell}\, x_{K,L+\ell} \\ \sum_{\ell=0}^{L-1} e_{K,L+\ell}\, x_{K,L+\ell-1} \\ \sum_{\ell=0}^{L-1} e_{K,L+\ell}\, x_{K,L+\ell-(N-1)} \\ 0 \\ \vdots \\ 0 \end{bmatrix} \qquad \ldots (13)$$

It is known that the IFFT 148 and the FFT 149 for updating the coefficients can be omitted (see "Unconstrained Frequency-Domain Adaptive Filter" D.

Mansour et al., IEEE vol ASSP-30, No. 5, Oct. 1982).

In the construction shown in Fig. 14, the necessary number of multiplications for each input of one sample is expressed as follows:

$$\frac{N+L-1}{L}\{5\log_2\left(\frac{N+L-1}{2}\right) + 14\} \qquad \ldots (14)$$

where (N+L-1) is assumed to be an n-th power of 2. By the equation proposed by Mansour et al., this is expressed as:

$$\frac{N+L-1}{L}\{3\log_2\{\frac{N+L-1}{2}\} + 10\} \qquad \ldots (15)$$

On the other hand, by a usual adaptive control effected sample by sample in the time domain, the necessary number of multiplications is as large as 2N.

When L and N are nearly equal in order of magnitude, the value expressed by the equation (14) or (15) is small in comparison with the number 2N.

However, in an echo canceller for an acoustic system, the length of the impulse response is so long that the number N becomes several thousand or more. Whereas, the block length L should be as small as possible, because it provides, as it is, a processing delay, and in order to minimize the influence due to fluctuation of the system. If L is sufficiently small in comparison with N, the value expressed by the equation (14) or (15) becomes almost equal to or larger than 2N.

Also, if (N+L-1) is not an n-th power of 2, the echo canceller having the structure shown in Fig. 14 cannot operate effectively. For example, assume that N = 4096 and L = 50, then the FFT portion shown in Fig. 14 must be arranged to execute its function at 8096, resulting in an increase of the useless portion.

SUMMARY OF THE INVENTION

In view of the above problems in the prior art, an object of the present invention is to provide a new echo canceller in which the number of multiplications is decreased and the processing delay is shortened.

To attain the above object, there is provided, according to the present invention, an echo canceller for cancelling an echo signal passed through an echo path with an impulse response N, having an input end for receiving an input digital signal series and an output end for providing an error signal. The echo canceller comprises: an N' sample overlap processing part 10, connected to the input end, for receiving the input digital signal series, and for outputting 2N' samples of said input digital signal series with N' samples being overlapped; a 2N'-point fast Fourier transform part, connected to the N' sample overlap processing part, for effecting a fast Fourier transform on the 2N' samples output from the overlap processing part so as to output 2N' points of signals expressed in the frequency domain; an FIR filtering part for dividing said impulse response N in said input digital signal series into k blocks each consisting of N' samples where k and N are integers, connected to the 2N'-point fast Fourier transform part, for adding the multiplied output signals of said 2N'-point fast Fourier transform part; a coefficient updating part, connected to the output end, for updating the coefficient of the FIR filtering part; a 2N'-point inverse fast Fourier transform part, connected to the FIR filtering part, for effecting an inverse fast Fourier transform on the output of the FIR filtering part; an output processing part, connected to the 2N'-point inverse fast Fourier transform part (15), for deleting the first N' samples from the outputs of the 2N'-point inverse fast Fourier transform part and for outputting the last N' samples as an estimated echo signal; a delay circuit, connected to the echo path, for delaying the echo signal passed through the echo path by the N' samples; and a subtractor, connected to the output processing part and the delay circuit, for obtaining an error signal corresponding to the difference between the output of the delay circuit and the estimated echo signal.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a block diagram showing a general structure of an echo canceller according to an embodiment of the present invention;

Fig. 2 shows an FIR filter having a tap number 2N', for explaining the basic idea of the present invention;

Fig. 3 is an equivalent circuit of the FIR filter shown in Fig. 2, for explaining the basic idea of the present invention;

Fig. 4 is a block diagram showing an echo canceller including the FIR filter shown in Fig. 3, for explaining the basic idea of the present invention;

Fig. 5 is a block diagram showing an echo canceller realized by the basic idea shown in Figs. 2 and 3, according to an embodiment of the present invention;

Fig. 6 is a diagram for explaining the concept of dividing the FIR filter into k blocks, according to an embodiment of the present invention;

Fig. 7 is a block diagram showing an echo canceller realized by the concept shown in Fig. 6, according to another embodiment of the present invention;

Fig. 8 is a block diagram showing an echo canceller according to still another embodiment of the present invention;

Fig. 9 is a diagram showing a conventional time domain echo canceller;

Figs. 10A to 10E are diagrams for explaining a conventional calculation of a convolution;

Figs. 11A to 11F are diagrams for explaining a conventional calculation of a cyclic convolution;

Fig. 12 is a block diagram showing a conventional block controlution calculation control utilizing FFT;

Figs. 13A to 13F are diagrams for explaining the updating of the coefficients in a block adaptive control; and

Fig. 14 is a block diagram showing a conventional echo canceller with FFT and block adaptive control;

BEST MODE FOR CARRYING OUT THE INVENTION

Figure 1 illustrates a general structure of an echo canceller according to an embodiment of the present invention. In the figure, a tap number N, which is determined by the impulse response of the system to be estimated, is considered to consist of k blocks of N' samples. Assuming that a frame unit consists of N/k = N' samples, then, the part 10 outputs 2N' samples (or 2 frames) in which N' samples are overlapped in the current input and the preceding input. A first 2N'-point FFT 11 receives the 2N' samples from the part 10 to effect a fast Fourier transform on the received 2N' samples so that the signals in the time domain are converted into the signals in the frequency domain. In a finite impulse response (FIR) filter 12, each of the 2N' points output from the first 2N'-point FFT 11 is delayed by N' x P samples, where 0 ≤ P ≤ k-1. The delayed 2N' samples are respectively multiplied with coefficients output from an update control part 17, and the multiplied outputs are added to obtain 2N' outputs of the FIR filter 12. A 2N' points IFFT effects an inverse FFT on the 2N' outputs from the FIR filter 12 so that the signals of 2N' samples in the frequency domain are converted into signals in the time domain.

An output processing part 16 deletes the first half N' samples from the outputs of the 2N' points IFFT 15 and outputs the last N' samples.

The update control part 17 effects an updating process of the coefficients of the FIR filter 12.

A delay circuit 18 receives an echo component y(n) passed through the echo path 19 and delays it by N' samples.

A subtractor 20 subtracts the output of the last N' samples of the output processing part 16 from the output of the delay circuit 8 to obtain error signals.

Since the FIR filtering is executed on the shortened block of N' samples by dividing the impulse response having the length N into k blocks, the process delay is greatly shortened while the merits of the echo canceller utilizing FFT, i.e., the small number of multiplications, are maintained. It should be noted that the estimated value $\hat{y}(n)$ is considered to be a sum of the convolution of each block and the input data x(n) when the impulse response length of the system to be estimated is N, which is divided into the blocks each having N' samples.

The basic concept of the present invention is described with reference to Figs. 2 to 4.

Figure 2 shows an FIR filter having a tap number 2N'. In the figure, 21-0 through 21-(2N'-2) are delay elements, each providing one sample of delay for the input signal x(n). 22-0 through 22-(2N'-1) are multipliers, each operating to multiply the input sample and an estimated value $\hat{h}_k$ of the impulse response $h_k$ of the system to be estimated, where k = 0, 1, 2, ..., or 2N'-1. The multiplied results are added by an adder 23. The added result is subtracted from the echo signal y(n) by a subtractor 20 to obtain an error signal e(n). An adaptive control is effected to make the error signal e(n) zero.

The FIR filter having the tap number 2N' shown in Fig. 2 can be considered to be a composite filter including a block B1 with a tap number N' and a block B2 with a tap number N'. Therefore, the FIR filter shown in Fig. 2 can be expressed as shown in Fig. 3.

In Figs. 2 and 3, the same reference symbols represent the same parts. The left side of Fig. 3 includes the block B1, and the right side of Fig. 3 includes the block B2. The outputs of the block B1 are added by an adder 23-I, and the output of the adder 23-1 is subtracted from the echo signal y(n) by a subtractor 20-1. The input signal x(n) is delayed by N' samples by delay elements 24-0 through 24-(N'-1) to be inputted into the block B2. The outputs of the block B2 are added by an adder 23-2, and the output of the adder 23-2 is subtracted from the output of the subtractor 20 to provide the error signal e(n).

The constitution of the circuit in the right side or in the left side in Fig. 3 is substantially the same as the conventional constitution shown in Fig. 14 except for the number of samples to be processed. Accordingly, the detailed construction of the echo canceller shown in Fig. 3 can be depicted as the construction shown in Fig. 4.

In Fig. 4, reference symbols 31-1 and 31-2 are N' overlap processing parts and 2N'-point FFTs; 32-1 and 32-2 are zero padding parts for padding N' zeros to the portion preceding the error signal e(n) at the output OUT; 33-1 and 33-2 are 2N' point FFT processing parts for effecting FFT on the outputs of the zero padding parts 32-1 and 32-2; 34-1 and 34-2 are coefficients multiplying parts or, in other words, FIR filtering parts; 35-1 and 35-2 are 2N' points inverse FFT processing parts; 36-1 and 36-2 are output processing parts for outputting the last N' samples by deleting the preceding N' samples; 37-1 and 37-2 are inverse FFT processing parts; and 38-1 and 38-2 are FFT processing parts. The IFFT processing parts 37-1 and 37-2 and the FFT processing parts 38-1 and 38-2 operate to provide the updated coefficients $H_0^i$ and $H_1^i$. As mentioned before in the description of the prior art, the IFFT processing parts 37-1 and 37-2 and the FFT processing parts 38-1 and 38-2 for updating the coefficients can be omitted.

The difference between the left side circuit and the right side circuit in Fig. 4 is that the outputs of the part 31-2 are respectively delayed by one sample by delay elements 39-0, 39-1, ..., and 39-(2N'-1). This corresponds to the N' samples of the delay elements shown in the right side circuit in Fig. 3.

Figure 5 shows an equivalent system to the construction shown in Fig. 4 except that the IFFT processing parts 37-1 and 37-2 and the FFT parts 38-1 and 38-2 in Fig. 4 are omitted in Fig. 5 and the separate parts 31-1 and 31-2, 32-1 and 32-2, 33-1 and 33-2, 34-1 and 34-2, 35-1 and 35-2, 36-1 and 36-2 in Fig. 4 are respectively combined as unit parts 31 through 36 in Fig. 5.

In Fig. 5, the FIR filtering part 34 has two-divided portions as already shown in Fig. 2. In one of the two-divided portions, the 2N' outputs of the N' sample overlap processing part and the 2N' point FFT 31 and the estimated coefficients $H_0^0$ through $H_0^{2N'-1}$ are multiplied to obtain these products. In another one of the two-divided portions, the above-mentioned 2N' outputs are delayed by the delay elements 39-0 through 39-(2N'-1). Then, the delayed outputs are multiplied to the estimated coefficients $H_1^0$ through $H_1^{2N'-1}$ to obtain these products. The products with respect to the first output are summed by the adder 40-0. Similarly, the products with respect to the second, third, ..., and (2N'-1)th outputs are respectively summed by the corresponding adders 40-1 through 40-(2N'-1). Thus, at the outputs of these adders 40-0 through 40-(2N'-1), 2N' outputs of the FIR filter 34 are obtained.

By the system construction shown in Fig. 5, the input signal having the impulse response of N samples is divided into two so that the processing is effected on each block with N' samples, and the process delay is shortened to be N' samples.

In the above-described embodiment of the present invention, the FIR filter is divided into only two for the sake of simplicity of the description. The present invention, however, is not restricted to the above embodiment, and similar considerations are possible even when the number of divisions is increased.

Figures 6A through 6F are diagrams for explaining the concept of dividing the FIR filter into k blocks.

As shown in Fig. 6A, the estimated impulse response h is divided into k blocks $h_0$ , $h_1$ , ..., $h_{k-1}$ , each block consisting of N' samples. Figure 6B shows the input signal x. The output of the FIR filter is the sum of the convolutions of each block $h_i$ and the input signal x. The convolutions are shown in Figs. 6C, 6D, 6E, 6F, ... as the expressions $h_0*x$, $h_1*x$, ... and $h_{k-1}*x$.

For each component $h_i$ of the impulse response, the convolution output is delayed by ixN' samples. Therefore, under the assumption that the updating period of the tap coefficients of the block LMS algorithm is N' samples, the output of the FFT for i-th block must be delayed by i samples.

Figure 7 is a block diagram showing an echo canceller in which an FIR filter is divided into k blocks, taking into account the consideration described with reference to Figs. 6A through 6F, according to another embodiment of the present invention.

In Fig. 7, the construction is similar to the conventional one shown in Fig. 13 except that, in Fig. 7, the number of samples in one block is N' and the FIR filter includes k delay elements 13 connected in series for each block. The tap coefficient between two delay elements 13 is multiplied by a multiplier 14 to the estimated coefficient $H_i^j$ , where i = 0, 1, ..., or 2N'-1 and j = 0, 1, ..., or k-1. The multiplied results are summed by adders 12-0 through 12-(2N'-1). Reference number 10 represents an N' sample overlap processing part; 11 is a 2N' point FFT processing part; 15 is a 2N' point IFFT processing part; 16 is an output processing part for deleting the preceding N' samples and outputting the latter half N' samples; 17 is a coefficient updating part; 171 is a zero padding part for padding N' zeros into the preceding half N' samples of the error signal e(n); and 172 is a 2N' point FFT processing part for processing the output of the zero padding part.

In Fig. 7, the constitution of the coefficient updating part 17 is simple because the IFFT 37-1 or 37-2 and the FFT 38-1 or 38-2 are omitted.

The operation of the echo canceller shown in Fig. 7 is as follows.

The input signal x(n) from the input terminal IN is processed by the N' sample overlap processing part 10 so that a unit consisting of 2N' samples of the input signals is output at each output timing. In the overlapping process, the latter half N' samples of the current unit of 2N' samples are overlapped with the preceding half N' samples of the immediately before output (see Figs. 6B through 6F). The 2N' samples output from the N' sample overlap processing part 10 are received by the 2N' point FFT processing part 11 and are processed by a fast Fourier transform so that the input signal expressed in the time domain is transformed into the signal expressed in the frequency domain. The complex conjugates of the 2N' outputs from the 2N'-point FFT processing part 11 are respectively multiplied with the error coefficients $E_0$ through $E_{2N'-1}$ which are output from the 2N'-point FFT processing part 172 in the coefficient updating part 17 at the error signal side, resulting in the updating parts of the coefficients $H_0^0$ through $H_{2N'-1}^0$ which correspond to the block $h_0$. With respect to the outputs which are delayed by one sample by the delay elements 13, the complex conjugates thereof are multiplied with the error coefficients $E_0$ through $E_{2N'-1}$ so that the updating parts of the coefficients $H_0^1$ through $H_{2N'-1}^1$ corresponding to the block $h_1$ are obtained. Similarly, the updating parts of the coefficients $H_0^i$ through $H_{2N'-1}^i$ corresponding to the blocks $h_i$ (where i = 2, 3, ..., or k-1) are obtained. The coefficients obtained in such a way as above are classified into groups respectively corresponding to the outputs of the 2N' point FFT processing part 11. Then, the multiplied data by the classified coefficients $H_0^0$

through $H_0^{k-1}$, $H_1^0$ through $H_1^{k-1}$, ... and $H_{2N'-1}^0$ through $H_{2N'-1}^{k-1}$ are summed respectively by the adders 12-0 through 12-(2N'-1). The added results are input to the 2N' point IFFT processing part 15 and are processed therein by an inverse fast Fourier transform. Then, in the output processing part 16, the preceding half N' samples are deleted so that the last half N' samples are output as the estimated echo signal $\hat{y}(n)$. By subtracting the value $\hat{y}(n)$ from the echo signal y(n) passed through the echo path, the error signal e(n) is obtained.

Since the impulse response is divided into a plurality of blocks, each block consisting of N' samples, the delay in the N' delay circuit 18 can be made as short as N'. The delay N' can be made smaller and smaller by increasing the number of divided blocks so that the processing delay can be reduced.

In the embodiment shown in Fig. 7, the coefficient updating part 17 is simplified.

Figure 8 shows an another embodiment of the present invention in which only a coefficient updating part 17a is different from the coefficient updating part 17 shown in Fig. 7. The other parts are similar to those shown in Fig. 7.

In Fig. 8, the coefficient updating part 17a includes the zero padding part 171, the 2N'-point FFT processing part 172, multipliers 173, 174, ..., 2N'-point IFFT processing parts 175-0 through 175-(k-1), and 2N'-point FFT processing parts 176-0 through 176-(k-1).

By the coefficient updating part 17a shown in Fig. 8, the coefficients $H_k^i$, where k = 0, 1, 2, ..., or 2N'-1, and i = 0, 1, 2, ..., or k-1, can be derived by dividing the conventional equation (3) into a plurality of blocks each consisting of N' samples, as follows.

$$\left[\begin{array}{c}
H^{0}_{(k+1)\cdot N'} \\
H^{1}_{(k+1)\cdot N'} \\
\vdots \\
H^{N'-1}_{(k+1)\cdot N'} \\
\hline
H^{N'}_{(k+1)\cdot N'} \\
\vdots \\
H^{2N'-1}_{(k+1)\cdot N'} \\
\hline
H^{2N'}_{(k+1)\cdot N'} \\
\vdots \\
H^{KN'-1}_{(k+1)\cdot N'}
\end{array}\right]
=
\left[\begin{array}{c}
H^{0}_{k\cdot N'} \\
H^{1}_{k\cdot N'} \\
\vdots \\
H^{N'-1}_{k\cdot N'} \\
\hline
H^{N'}_{k\cdot N'} \\
\vdots \\
H^{2N'-1}_{k\cdot N'} \\
\hline
H^{2N'}_{k\cdot N'} \\
\vdots \\
H^{KN'-1}_{k\cdot N'}
\end{array}\right]$$

$$
+ 2\mu \left[ \begin{array}{l}
\sum_{L=0}^{N'-1} e_{k\cdot N'+L}\, x_{k\cdot N'+L} \\
\sum_{L=0}^{N'-1} e_{k\cdot N'+L}\, x_{k\cdot N'+L-1} \\
\vdots \\
\sum_{L=0}^{N'-1} e_{k\cdot N'+L}\, x_{k\cdot N'+L-(N'-1)} \\
\hline
\sum_{L=0}^{N'-1} e_{k\cdot N'+L}\, x_{k\cdot N'+L-N'} \\
\vdots \\
\sum_{L=0}^{N'-1} e_{k\cdot N'+L}\, x_{k\cdot N'+L-(2N'-1)} \\
\hline
\sum_{L=0}^{N'-1} e_{k\cdot N'+L}\, x_{k\cdot N'+L-2N'} \\
\vdots \\
\sum_{L=0}^{N'-1} e_{k\cdot N'+L}\, x_{k\cdot N'+L-(KN'-1)}
\end{array} \right] \quad \ldots \; (16)
$$

In the equation (16), the number N' in the coefficient $H^{i}_{k,N'}$ or $H^{i}_{k+1,N'}$ denotes that each block consists of N' samples. The m-th block in the above equation plus N' zeros, gives the following equation (17).

$$\begin{bmatrix} H^{mN'}_{(k+1)\cdot N'} \\ H^{mN'+1}_{(k+1)\cdot N'} \\ \vdots \\ H^{(m+1)N'-1}_{(k+1)\cdot N'} \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \begin{bmatrix} H^{mN'}_{k\cdot N'} \\ H^{mN'+1}_{k\cdot N'} \\ \vdots \\ H^{(m+1)N'-1}_{k\cdot N'} \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

$$
+\,2\,\mu
\begin{bmatrix}
\sum_{L=0}^{N'-1} e_{k\cdot N'+L}\; x_{k\cdot N'+L-mN'} \\
\sum_{L=0}^{N'-1} e_{k\cdot N'+L}\; x_{k\cdot N'+L-mN'-1} \\
\cdot \\
\cdot \\
\cdot \\
\sum_{L=0}^{N'-1} e_{k\cdot N'+L}\; x_{k\cdot N'+L-(m+1)N'+1} \\
0 \\
0 \\
\cdot \\
\cdot \\
\cdot \\
0
\end{bmatrix}
\quad \ldots\ (17)
$$

By comparing the second term in the above equation (17) with the conventional expression (13), it can be appreciated that the tap coefficients corresponding to the m-th block can be calculated by a structure similar to the conventional one shown in Fig. 14, except that the number of samples to be processed in the coefficient updating part 17a is 2N' which is smaller than the number of samples processed in the conventional coefficient updating part shown in Fig. 14. That is, in the 2N'-point FFT processing part 176-k shown in Fig. 8, N' "0"s are added to the outputs of the IFFT processing part 175-0 so that the updating parts of the coefficients $H_i^k$ (i = 0, 1, 2, ..., and 2N'-1) are output. The update of these coefficients is effected in a lump each time N' samples of input data are received.

It should be noted that the functions of the aforementioned parts shown in Figs. 5 and 8 can be realized by arithmetic functions in a program controlled type processor.

INDUSTRIAL APPLICABILITY

From the foregoing description, it is apparent that, according to the present invention, the necessary numbers of multiplications corresponding to the expressions (14) and (15) are respectively:

$$(4k + 6)\log_2 N' + 16k + 12 \quad \ldots (18)$$

$$6\log_2 N' + 8k + 12 \quad \ldots (19),$$

where k is a number of divisions; N' is the block length to be adapted, and N = N' x k. This value is generally much smaller than 2N even when the processing delay N' is small.

That is, according to the present invention, a block adaptive algorithm can be realized with a smaller number of multiplications than conventionally used.

Further, by selecting the value N' to be a power of 2, an effective construction can be realized. That is, the construction shown in Fig. 5, Fig. 7 or Fig. 8 can be said to have a higher flexibility in comparison with the conventional construction.

CLAIMS

1. An echo canceller for cancelling an echo signal passed through an echo path (19) with an impulse response (N), having an input end for receiving an input digital signal series (x(n)) and an output end for providing an error signal, comprising:

an N' sample overlap processing part (10), connected to said input end, for receiving said input digital signal series (x(n)), and for outputting 2N' samples of said input digital signal series with N' samples being overlapped;

a 2N'-point fast Fourier transform part (11), connected to said N' sample overlap processing part (10), for effecting a fast Fourier transform on the 2N' samples output from said overlap processing part so as to output 2N' points of signals expressed in the frequency domain;

an FIR filtering part (12) for dividing said impulse response N in said input digital signal series into k blocks each consisting of N' samples where k and N are integers, connected to said 2N'-point fast Fourier transform part (11), for adding the multiplied output signals of said 2N'-point fast Fourier transform part (11);

a coefficient updating part (17), connected to said output end, for updating the coefficient of said FIR filtering part (12);

a 2N'-point inverse fast Fourier transform part (15), connected to said FIR filtering part (12), for effecting an inverse fast Fourier transform on the output of said FIR filtering part;

an output processing part (16), connected to said 2N'-point inverse fast Fourier transform part (15), for deleting the first N' samples from the outputs of said 2N'-point inverse fast Fourier transform part (15) and for outputting the last N' samples as an estimated echo signal;

a delay circuit (18), connected to said echo path, for delaying the echo signal (y(n)) passed through said echo path by the N' samples; and

a subtractor (20), connected to said output processing part (16) and said delay circuit (18), for obtaining an error signal (e(n)) corresponding to the difference between the output of said delay circuit (18) and said estimated echo signal.

2. An echo canceller as set forth in claim 1, wherein said coefficient updating part (17) comprises:

a zero padding part (171), connected to said output end, for padding N' "0"s before the N' samples of said error signal included in the output so as to form a frame consisting of 2N' samples; and

another 2N'-point fast Fourier transform part (172), connected to the output of said zero padding part (171), for effecting a fast Fourier transform on the outputs of said zero padding part (171) so as to output 2N' updating error coefficients ($E_0$-$E_{2N'-1}$).

3. An echo canceller as set forth in claim 2, wherein said coefficient updating part (17) further comprises:

a 2N'-point inverse fast Fourier transform part (175-0 through 175-(K-1)), connected to the multiplied output of said another 2N'-point fast Fourier transform part (172), for converting the multiplied output signals of said another 2N'-point fast Fourier transform part (172) from the frequency domain expression to the time domain expression; and

another 2N'-point fast Fourier transform part, connected to the output of said another 2N'-point inverse fast Fourier transform part, effecting a fast Fourier transform on 2N' samples of data consisting the first N' samples of the output of said still another inverse fast Fourier transform part and N' samples of "0"s so as to output the updating part of the coefficients ($H^i$) which are applied to said FIR filtering part (12).

4. An echo canceller as set forth in claim 1, wherein said integer N' is selected to be a power of two.

5. An echo canceller for cancelling an echo signal passing through a system to be estimated, comprising:

an overlap processing part (10) for receiving input signal series X(n) of 2N' samples and for overlapping said N' samples;

a fast Fourier transform part (11) for effecting a Fourier transform on the 2N' samples overlapped by said overlap processing part (10);

an FIR filtering part (12) for dividing an impulse response length N of said system into k blocks each consisting of N' samples, for adding the multiplied outputs of said fast Fourier transform part (11);

an inverse fast Fourier transform part (15) for effecting an inverse fast Fourier transform on the output signals of said FIR filtering part (12);

an output processing part (16) for obtaining an echo cancelling signal by selecting the last N' samples from the 2N' samples of the output signal of said inverse fast Fourier transform part (15); and

a coefficient updating part (17) for effecting an updating process of the coefficients of said FIR filtering part (12).

6. An echo canceller as set forth in claim 5, wherein the number N' in said N' samples is selected to be a power of two.

Fig. 1

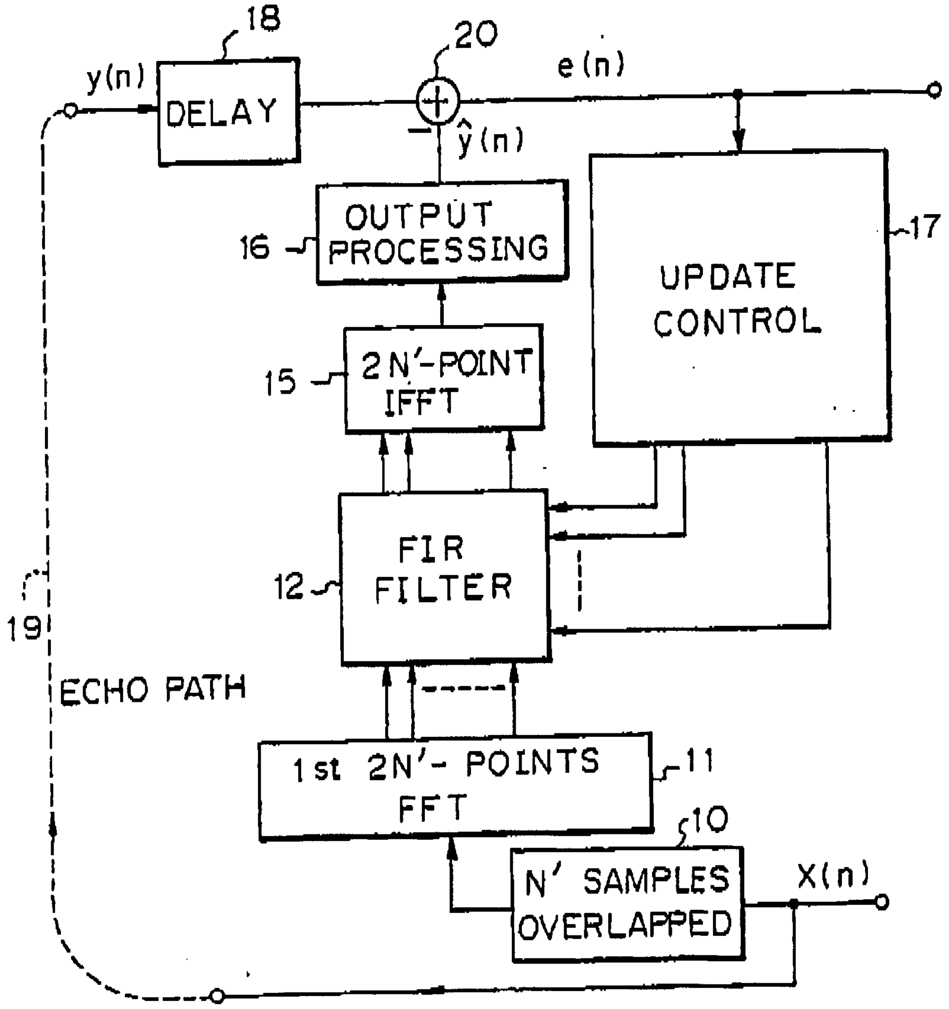

18
20
y(n)
DELAY
e(n)
ŷ(n)
OUTPUT PROCESSING
16
17
UPDATE CONTROL
2N'-POINT IFFT
15
FIR FILTER
12
19
ECHO PATH
1st 2N'- POINTS FFT
11
10
N' SAMPLES OVERLAPPED
X(n)

Fig. 2

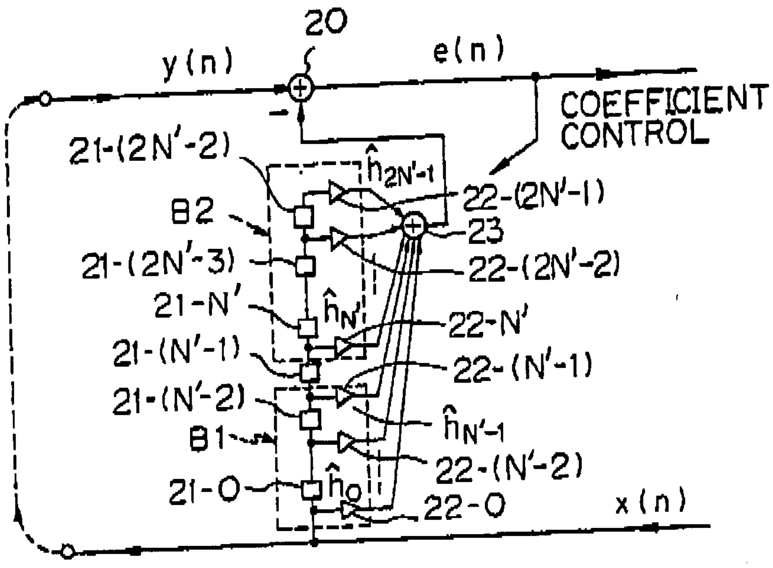
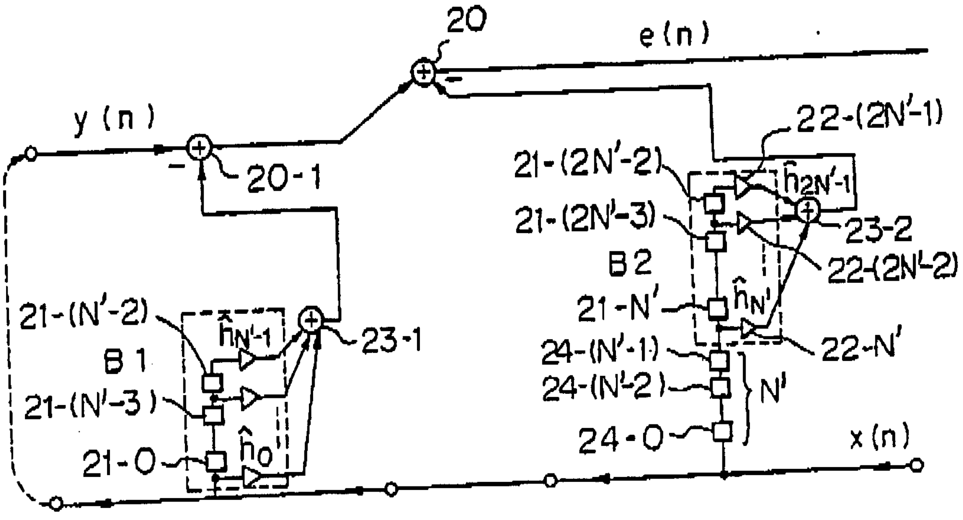

20
y(n)
e(n)
COEFFICIENT CONTROL
21-(2N'-2)
$\hat{h}_{2N'-1}$
22-(2N'-1)
B2
23
21-(2N'-3)
22-(2N'-2)
21-N'
$\hat{h}_{N'}$
22-N'
21-(N'-1)
22-(N'-1)
21-(N'-2)
$\hat{h}_{N'-1}$
B1
22-(N'-2)
21-0
$\hat{h}_0$
22-0
x(n)

Fig. 3


20
e(n)
y(n)
20-1
22-(2N'-1)
21-(2N'-2)
$\hat{h}_{2N'-1}$
21-(2N'-3)
23-2
B2
22-(2N'-2)
21-N'
$\hat{h}_{N'}$
21-(N'-2)
$\hat{h}_{N'-1}$
23-1
22-N'
B1
24-(N'-1)
21-(N'-3)
24-(N'-2)
N'
$\hat{h}_0$
21-0
24-0
x(n)

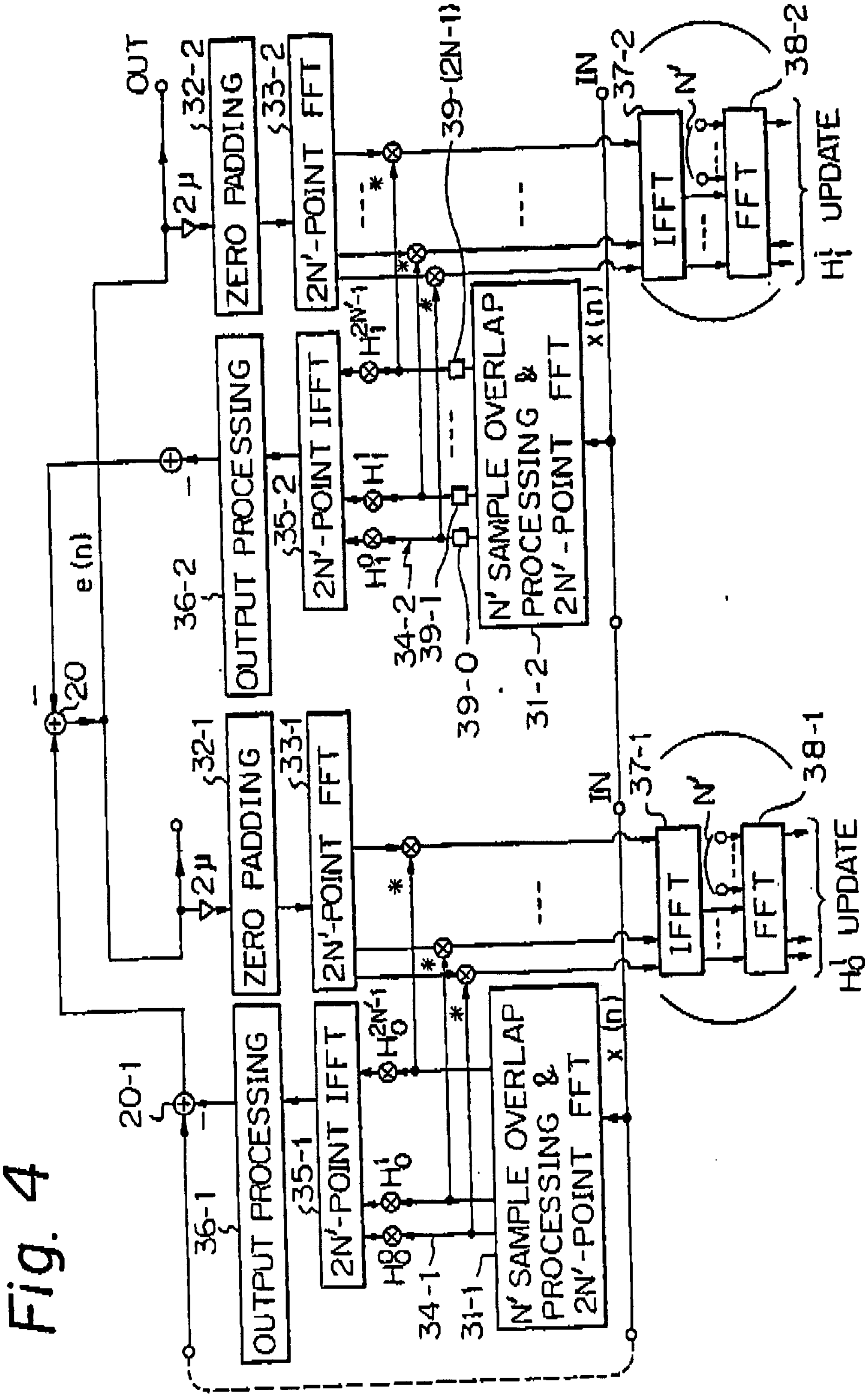
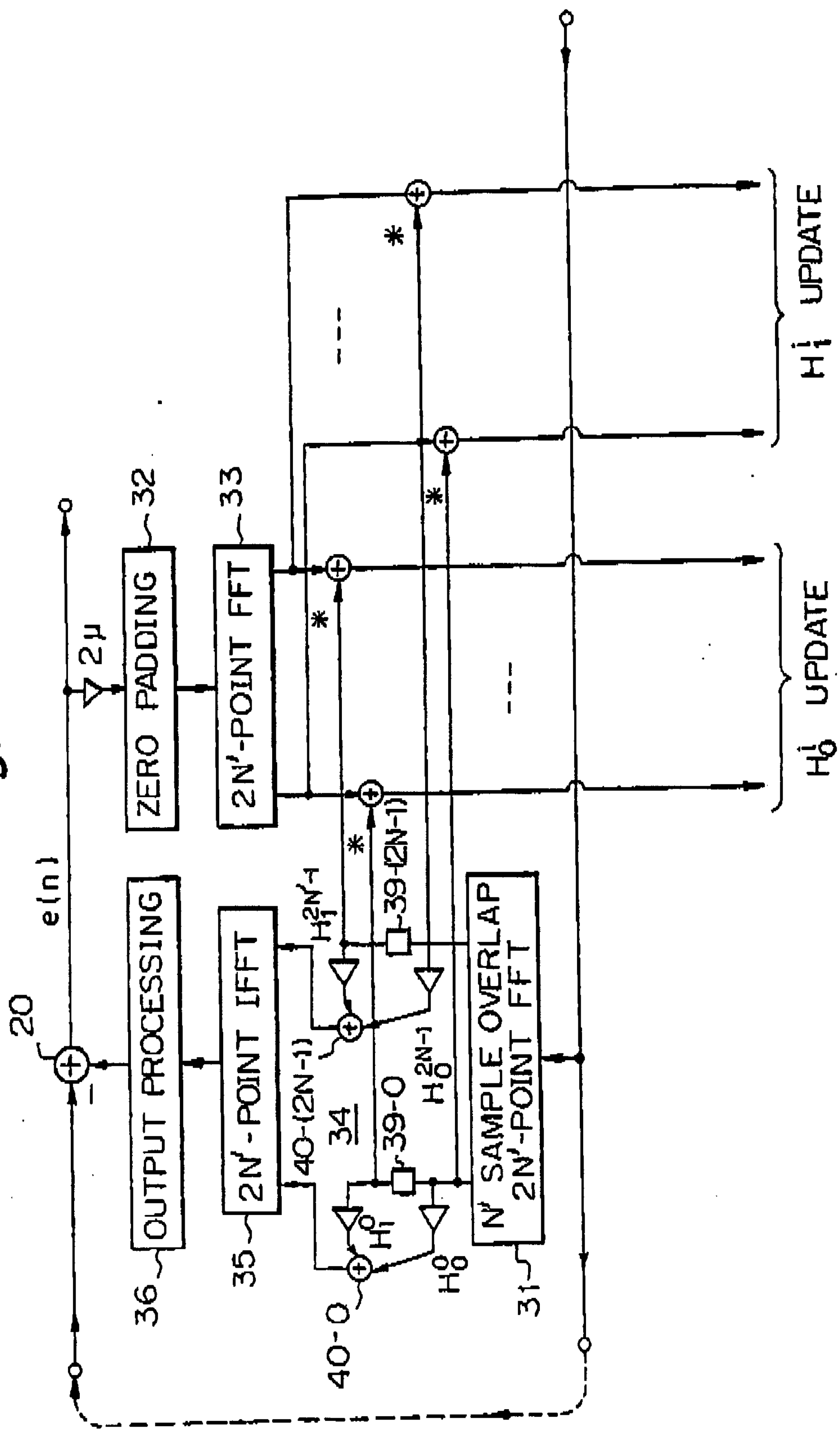

OUT
IN
IN
ZERO PADDING
ZERO PADDING
2N'-POINT FFT
2N'-POINT FFT
OUTPUT PROCESSING
OUTPUT PROCESSING
2N'-POINT IFFT
2N'-POINT IFFT
N' SAMPLE OVERLAP PROCESSING & 2N'-POINT FFT
N' SAMPLE OVERLAP PROCESSING & 2N'-POINT FFT
IFFT
FFT
IFFT
FFT
UPDATE
UPDATE
$H_1^i$
$H_0^i$
$H_1^0$
$H_1^1$
$H_1^{2N'-1}$
$H_0^0$
$H_0^1$
$H_0^{2N'-1}$
2μ
2μ
e (n)
x (n)
x (n)
N'
N'
20
20-1
32-1
32-2
33-1
33-2
34-1
34-2
35-1
35-2
36-1
36-2
31-1
31-2
37-1
37-2
38-1
38-2
39-0
39-1
39-(2N-1)
*

Fig. 4

Fig. 5


20
e(n)
2μ
ZERO PADDING
32
2N'-POINT FFT
33
OUTPUT PROCESSING
36
2N'-POINT IFFT
35
40-(2N-1)
40-0
34
39-0
39-(2N-1)
$H_1^{2N-1}$
$H_0^{2N-1}$
$H_1^0$
$H_0^0$
N' SAMPLE OVERLAP 2N'-POINT FFT
31
*
$H_0^i$ UPDATE
$H_1^i$ UPDATE

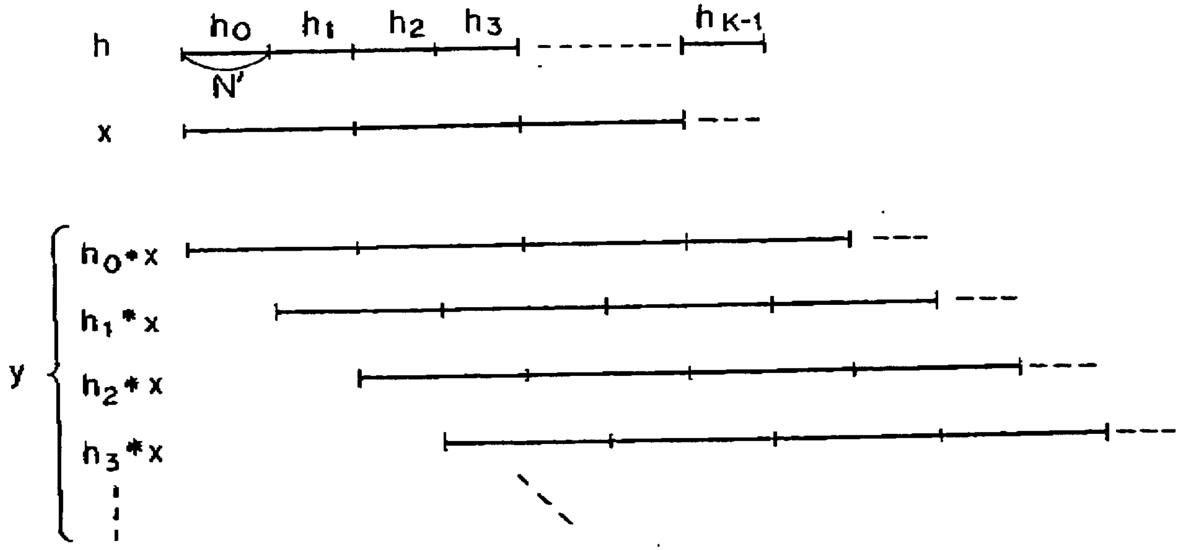


Fig. 6A
Fig. 6B
Fig. 6C
Fig. 6D
Fig. 6E
Fig. 6F

Fig. 7

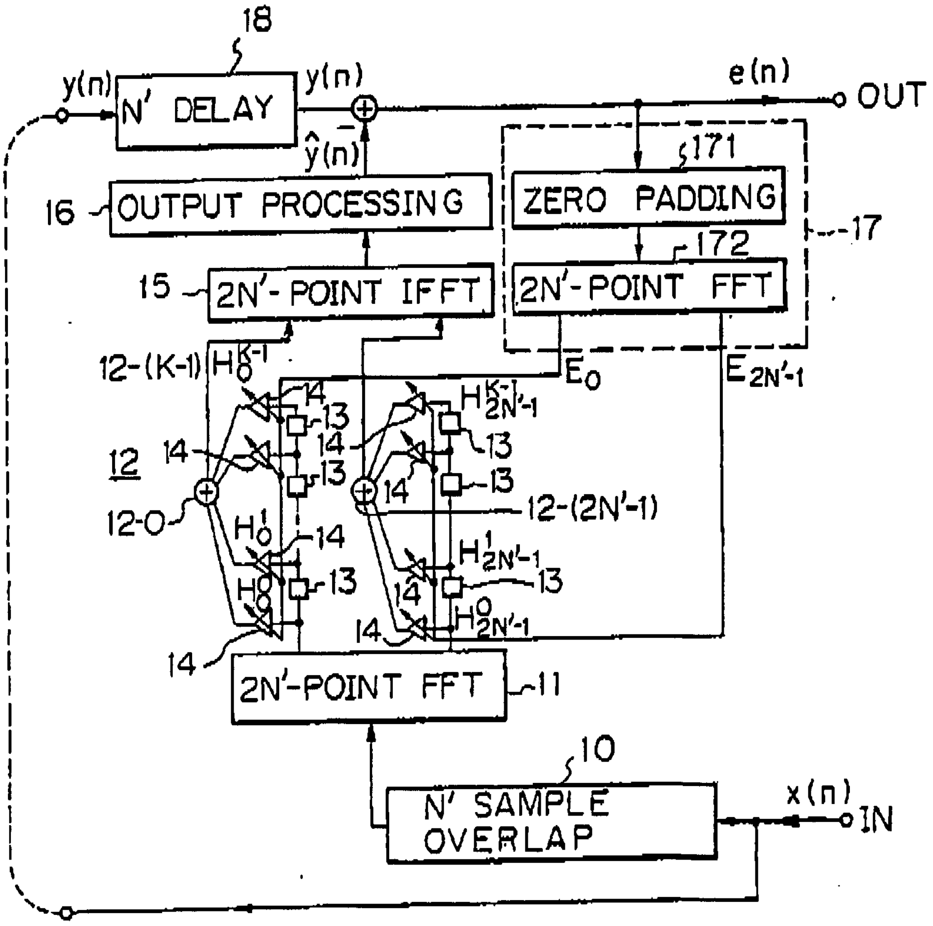

18
y(n)
N' DELAY
y(n)
ŷ(n)
e(n)
OUT
16
OUTPUT PROCESSING
171
ZERO PADDING
17
172
15
2N'- POINT IFFT
2N'- POINT FFT
E0
E2N'-1
12-(K-1)
H0K-1
H2N'-1K-1
14
13
12
12-0
12-(2N'-1)
H01
H00
H2N'-11
H2N'-10
2N'-POINT FFT
11
10
N' SAMPLE OVERLAP
x(n)
IN

Fig. 8

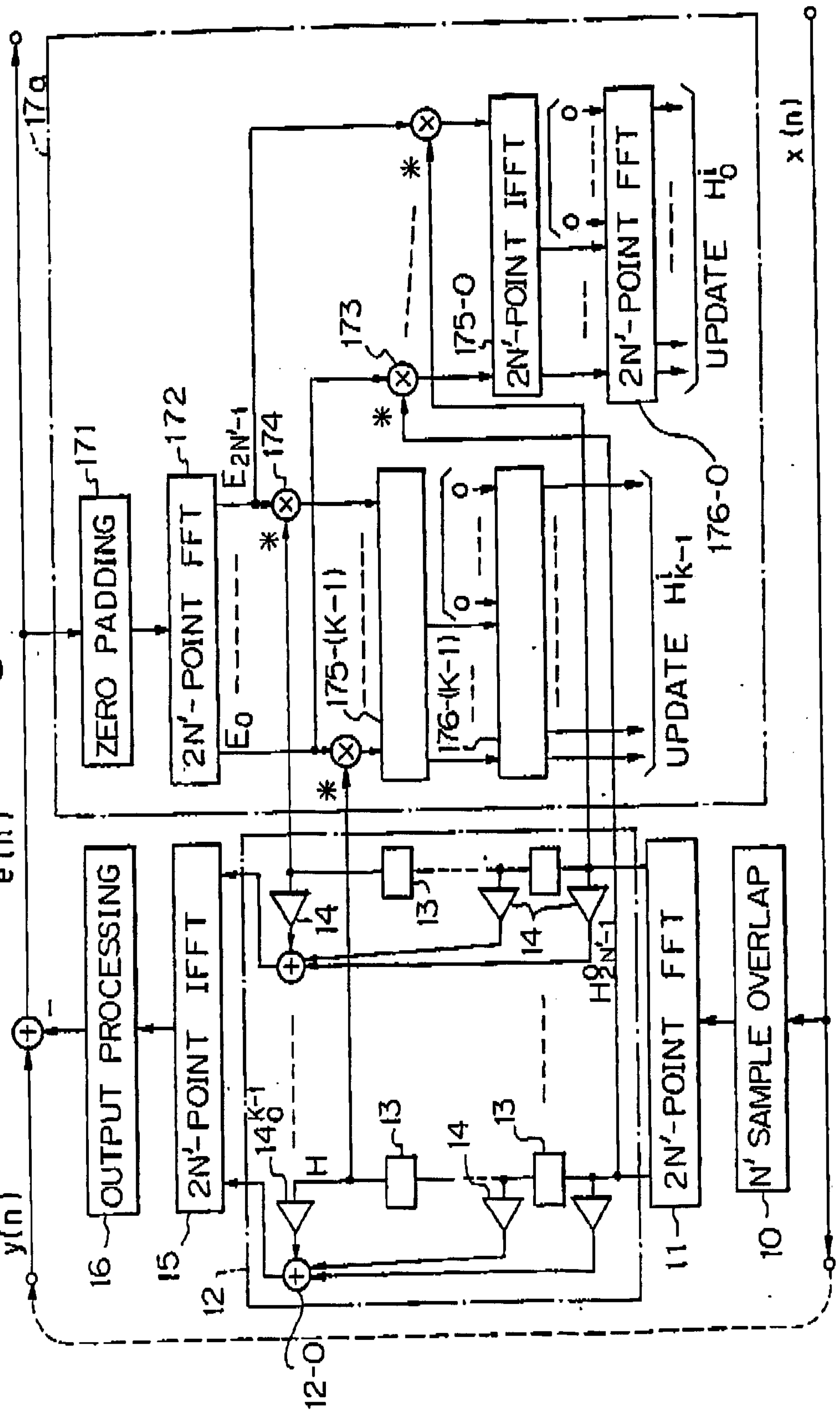

17a
ZERO PADDING
171
2N'-POINT FFT
172
$E_{2N'-1}$
$E_0$
174
173
175-O
175-(K-1)
176-(K-1)
176-O
2N'-POINT IFFT
2N'-POINT FFT
UPDATE $H^i_0$
UPDATE $H^i_{k-1}$
x(n)
e(n)
y(n)
OUTPUT PROCESSING
2N'-POINT IFFT
2N'-POINT FFT
N' SAMPLE OVERLAP
$H^{K-1}_0$
$H^0_{2N'-1}$
H
13
14
16
15
12
12-O
11
10

Fig. 9

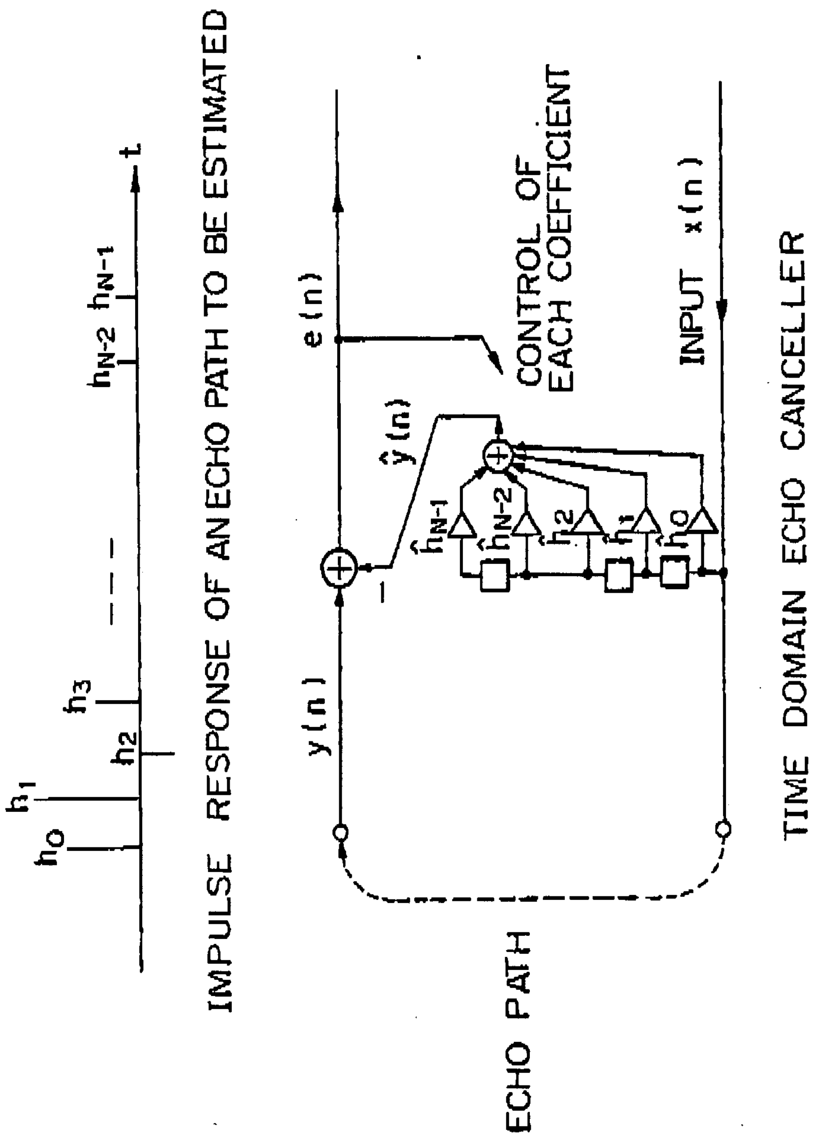
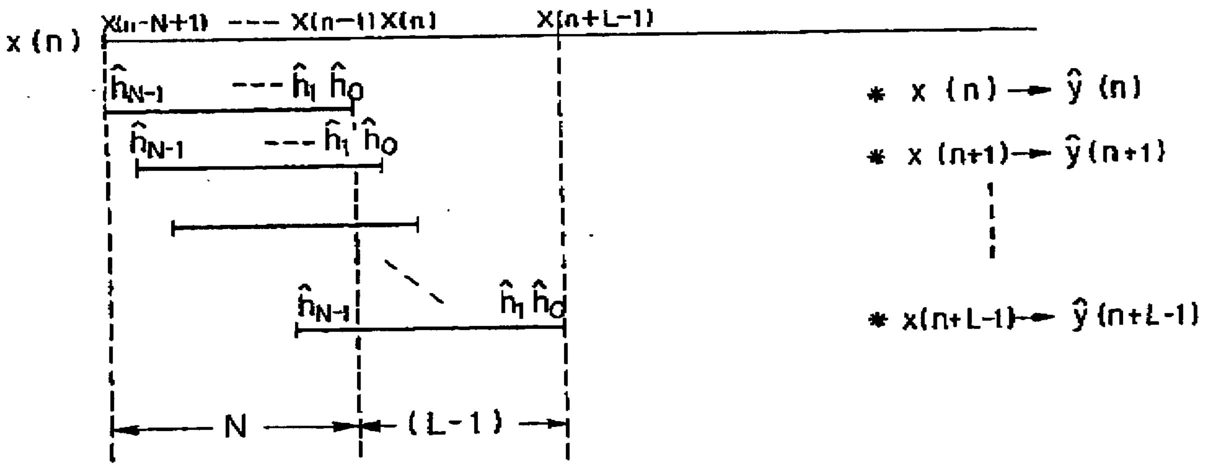

$h_0$
$h_1$
$h_2$
$h_3$
$h_{N-2}$
$h_{N-1}$
t
IMPULSE RESPONSE OF AN ECHO PATH TO BE ESTIMATED
y(n)
e(n)
$\hat{y}(n)$
-
$\hat{h}_{N-1}$
$\hat{h}_{N-2}$
$\hat{h}_2$
$\hat{h}_1$
$\hat{h}_0$
CONTROL OF EACH COEFFICIENT
INPUT x(n)
ECHO PATH
TIME DOMAIN ECHO CANCELLER

Fig. 10A
Fig. 10B
Fig. 10C
Fig. 10D
Fig. 10E


x (n)
X(n-N+1) --- X(n-1) X(n)
X(n+L-1)
$\hat{h}_{N-1}$ --- $\hat{h}_1$ $\hat{h}_0$
$\hat{h}_{N-1}$ --- $\hat{h}_1$ $\hat{h}_0$
$\hat{h}_{N-1}$ $\hat{h}_1$ $\hat{h}_0$
N
(L-1)
* x (n) → $\hat{y}$ (n)
* x (n+1) → $\hat{y}$ (n+1)
* x(n+L-1) → $\hat{y}$ (n+L-1)

CONVOLUTION OF x (n) and $\hat{h}_k$

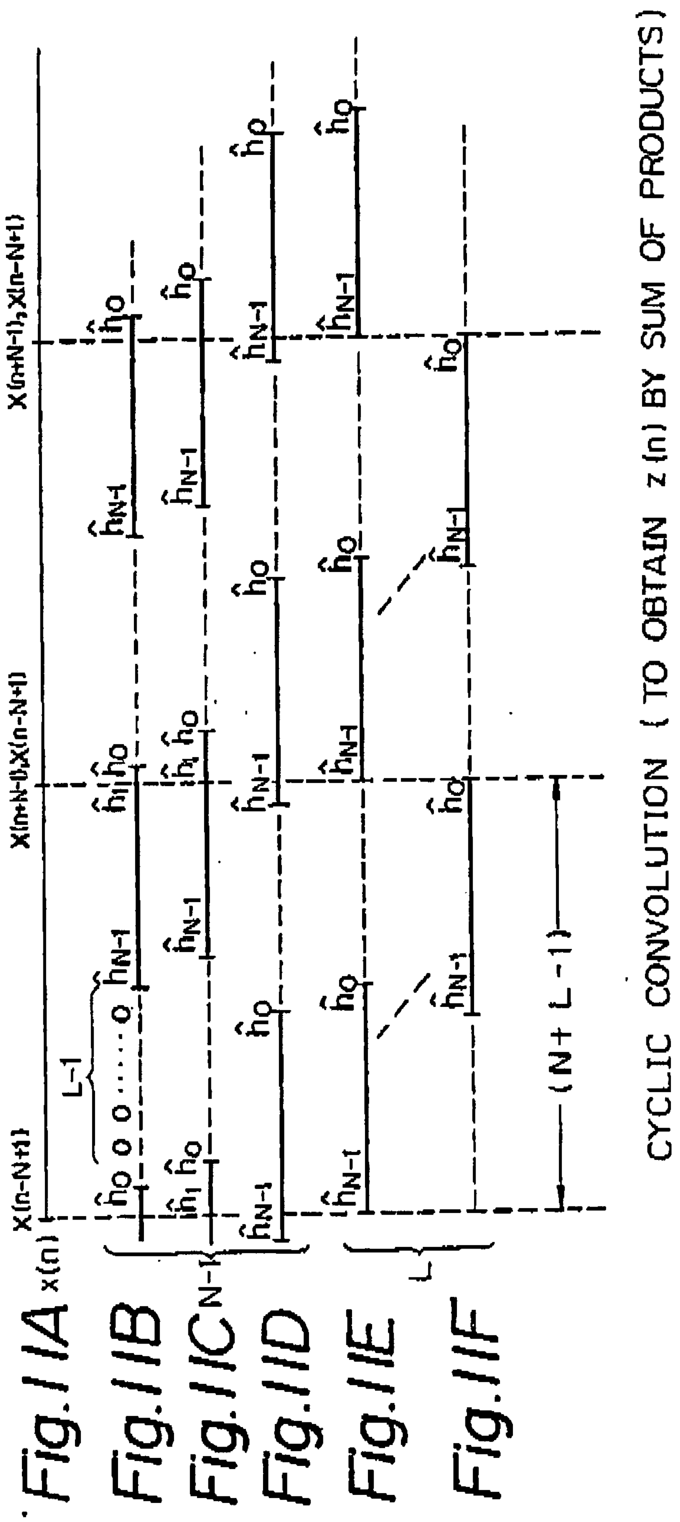
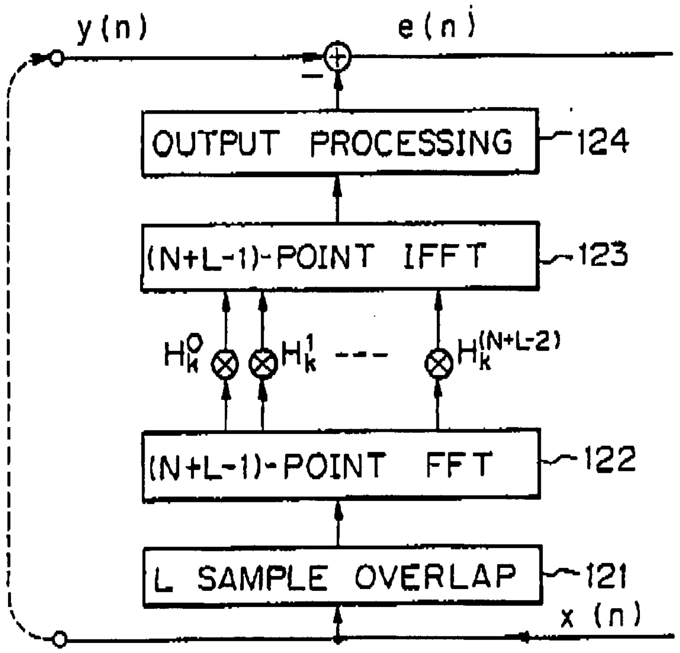


Fig.11A
Fig.11B
Fig.11C
Fig.11D
Fig.11E
Fig.11F

Fig. 12



CONVOLUTION CALCULATION WITH FFT

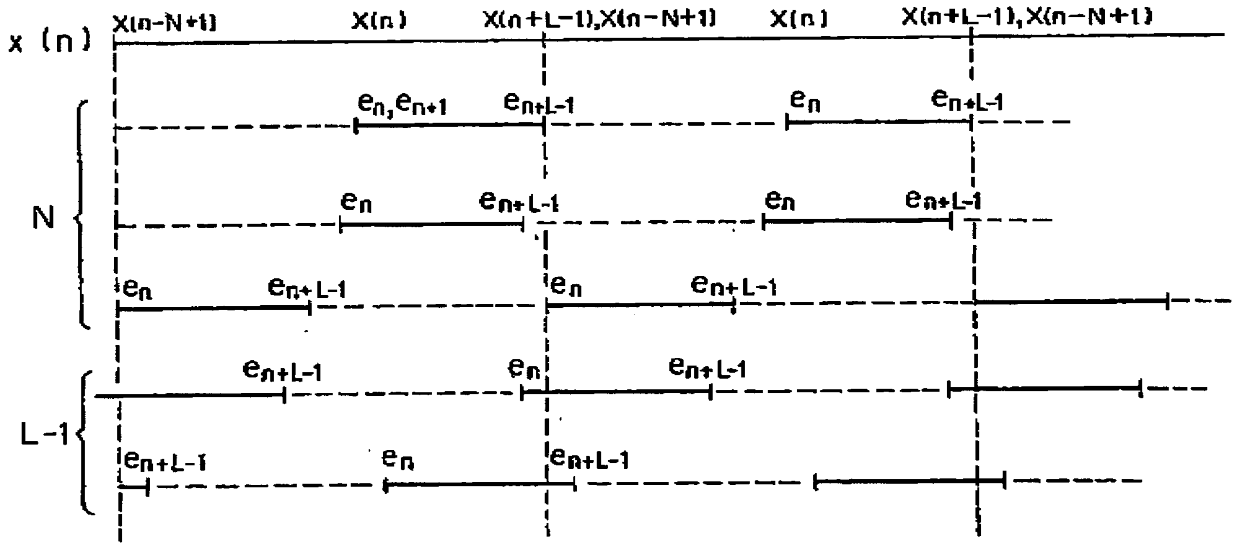
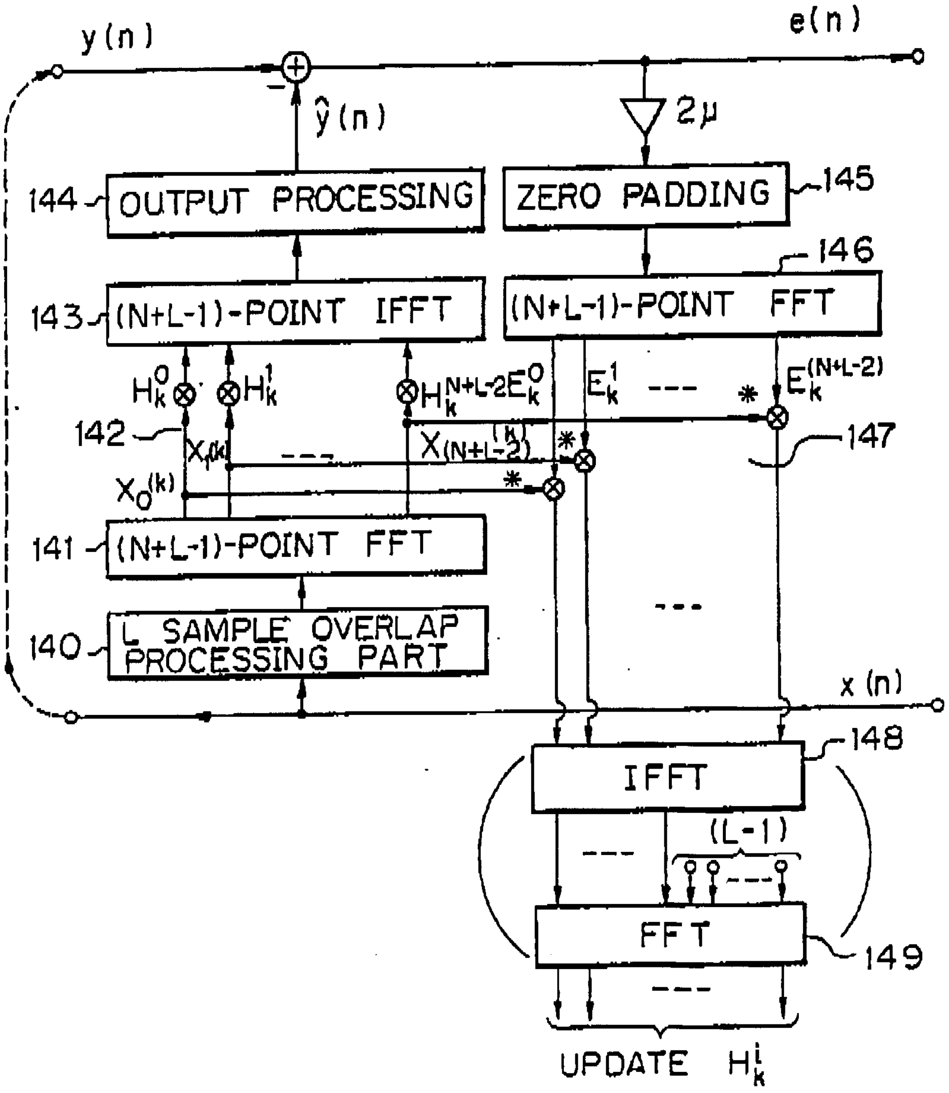


Fig.13A
Fig.13B
Fig.13C
Fig.13D
Fig.13E
Fig.13F

Fig. 14


y(n)
e(n)
ŷ(n)
2μ
144
OUTPUT PROCESSING
ZERO PADDING
145
146
143
(N+L-1)-POINT IFFT
(N+L-1)-POINT FFT
$H_k^0$
$H_k^1$
$H_k^{N+L-2}$
$E_k^0$
$E_k^1$
$E_k^{(N+L-2)}$
142
147
$X_1(k)$
$X_{(N+L-2)}(k)$
$X_0(k)$
*
141
(N+L-1)-POINT FFT
140
L SAMPLE OVERLAP PROCESSING PART
x(n)
148
IFFT
(L-1)
FFT
149
UPDATE $H_k^i$

## List of Reference Numbers

| | |
|---|---|
| 10 ..... | N' sample overlap processing part |
| 11 ..... | 2N'-point fast Fourier transform part |
| 12 ..... | FIR filtering part |
| 15 ..... | 2N'-point inverse fast Fourier transform part |
| 16 ..... | output processing part |
| 17 ..... | coefficient updating part |
| 18 ..... | delay circuit |
| 19 ..... | echo path |

# INTERNATIONAL SEARCH REPORT

International Application No PCT/JP 87/00833

**I. CLASSIFICATION OF SUBJECT MATTER** (if several classification symbols apply, indicate all) [6]

According to International Patent Classification (IPC) or to both National Classification and IPC

$IPC^4$: H 04 B 3/23; H 03 H 21/00; G 06 F 15/332

**II. FIELDS SEARCHED**

Minimum Documentation Searched [7]

| Classification System | Classification Symbols [9] |
|---|---|
| $IPC^4$ | H 04 B; H 03 H; G 06 F |

Documentation Searched other than Minimum Documentation to the Extent that such Documents are Included in the Fields Searched [8]

**III. DOCUMENTS CONSIDERED TO BE RELEVANT** [9]

| Category * | Citation of Document, [11] with indication, where appropriate, of the relevant passages [12] | Relevant to Claim No. [13] |
|---|---|---|
| A | IEEE Transactions on Acoustics, Speech, and Signal Processing, volume ASSP-30, no. 5, October 1982, IEEE, (New York, US), D. Mansour et al.: "Unconstrained frequency-domain adaptive filter", pages 726-734 see the whole document cited in the application<br>-- | 1,5 |
| A | US, A, 4593161 (DESBLACHE et al.) 3 June 1986 see claim 1; figure 4<br>-- | 1,5 |
| A | IEEE Communications Magazine, volume 20, no. 3, May 1982, IEEE, (New York, US), F.J. Harris: "The discrete Fourier transform applied to time domain signal processing", pages 13-22 see page 17, right-hand column - page 18, left-hand column, line 24; figure 10<br>-------- | 1,5 |

* Special categories of cited documents: [10]

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

**IV. CERTIFICATION**

| | |
|---|---|
| Date of the Actual Completion of the International Search<br>18th January 1988 | Date of Mailing of this International Search Report<br>26 FEB 1988 |
| International Searching Authority<br>EUROPEAN PATENT OFFICE | Signature of Authorized Officer<br>P.C.G. VAN DER PUTTEN |

# ANNEX TO THE INTERNATIONAL SEARCH REPORT ON INTERNATIONAL PATENT APPLICATION NO. JP 8700833

SA 19240

This annex lists the patent family members relating to the patent documents cited in the above-mentioned international search report.
The members are as contained in the European Patent Office EDP file on 11/02/88
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

| Patent document cited in search report | Publication date | Patent family member(s) | Publication date |
|---|---|---|---|
| US-A- 4593161 | 03-06-86 | EP-A- 0130263<br>JP-A- 60010929 | 09-01-85<br>21-01-85 |

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82

A4

# DIGITAL EQUALIZATION USING FOURIER TRANSFORM TECHNIQUES

2694 (B-2)

Barry D. Kulp
Zoran Corporation
Needham Heights, Massachusetts

**Presented at
the 85th Convention
1988 November 3-6
Los Angeles**

AUDIO
ES

*This preprint has been reproduced from the author's advance manuscript, without editing, corrections or consideration by the Review Board. The AES takes no responsibility for the contents.*

*Additional preprints may be obtained by sending request and remittance to the Audio Engineering Society, 60 East 42nd Street, New York, New York 10165, USA.*

**AES**

**AN AUDIO ENGINEERING SOCIETY PREPRINT**

# DIGITAL EQUALIZATION USING FOURIER TRANSFORM TECHNIQUES

Barry D. Kulp
Zoran Corporation
Needham Heights, Massachusetts

## Abstract

Equalization using time domain digital convolution becomes increasingly computationally intensive as impulse response length increases. Fourier transform techniques greatly reduce the computational load. The corresponding theory is reviewed, and various applications are detailed, including room, loudspeaker, instrument, and ambience equalization. A practical real-time implementation using an off-the-shelf digital signal processing integrated circuit is described. Theoretical and practical limitations of the applications and implementation are discussed.

## INTRODUCTION

In this paper, we will explore both techniques and applications for performing digital equalization using Fourier transform theory. Some of the applications are ones that are not commonly thought of when the term "equalization" is used, which we normally think of as just manipulation of frequency spectrum characteristics, or filtering. For example, we will look at "ambience equalization", which we will take to mean manipulation of ambient, or reverberant, characteristics. This manipulation can include both the cancellation of a room's undesirable ambient characteristics and/or the creation of a desired ambient response.

All of the applications to be described, from simple filtering to reverb generation, can be achieved in the digital domain using time domain

convolution techniques. However, in many cases, the amount of computation required to perform the convolution directly in the time domain would be prohibitive, relative to practical constraints of processing time and/or hardware cost. Fortunately, frequency domain processing provides a solution that greatly reduces the computational load involved in performing these convolutions.

It will be assumed that the reader has some basic knowledge of digital signal processing: that signals can be sampled, that the samples can be combined using delay registers, multipliers and adders in some way to do filtering, and that Fourier transforms exist and that they transform a signal from its time domain representation (the samples) to the frequency domain (a sampled spectrum) in a manner similar to the Laplace transforms of analog signal processing. Building on this basic knowledge, we will briefly review digital convolution and the Finite Impulse Response (FIR) filter structure. From there we will explore the theory behind using Fourier transforms to perform "fast convolution", greatly reducing the computational load as promised.

After exploring the theory, we will than look at some of the many applications using these techniques. Next, we will look at a hardware solution enabling real-time implementation of the algorithms involved. Finally, we will examine some of the limitations and problems encountered in both the applications and the implementation, and discuss some ways to solve them.

## CONVOLUTION THEORY AND FILTER STRUCTURES

Normally, when one thinks of a filter, one conceptualizes it as a change in the frequency domain characteristics (spectrum) of the signal. For example, a low-pass filter will allow the lower portions of the spectrum to pass through from input to output, but not the higher portions. However, due to the duality characteristics of the time and frequency domains, we know that multiplication in the frequency domain is equivalent to convolution in the time domain. In other words, to filter a signal with a filter having a certain frequency characteristic, the result being a signal with a spectrum that is the product of the spectra of the input signal and the filter, we must convolve (in the time domain) the input signal with the impulse response of the filter. The impulse response can be thought of as a signal whose spectrum is the frequency response of the filter.

In the continuous time domain (the analog world), we don't actually set out to perform the convolution. Instead, we try to design a circuit that has the desired frequency response (spectrum), which just happens to have a certain impulse response that we can measure, which it convolves with any input signal, the result of which we can observe. So, while the circuit is actually doing the convolution, we don't normally think of the filter in that way. Instead, when we talk about the filter, we talk about its frequency-domain function, e.g. low-pass, high-pass, band-reject, etc. Further evidence of this mind-set is the fact that analog filter design programs are designed to convert back and forth between spectral characteristics and component values for resistors and capacitors in a particular circuit topography, with the concept of impulse response not always directly addressed (although some filter design packages will plot the impulse response and/or the step response of the circuit).

However, in the discrete time domain (the digital world), in addition to thinking in terms of the frequency-domain function, we also often think in terms of convolutions and impulse responses. This is evidenced by the fact that digital filters are often described as infinite-impulse-response (IIR) or finite-impulse-response (FIR). In fact, FIR filters are usually implemented in such a way as to directly perform the convolution on the input samples by doing a sum-of-products calculation. Also, FIR filter design programs convert back and forth between spectral characteristics and impulse response coefficient values.

Within the realm of digital filtering, there are various tradeoffs between the IIR and FIR filter structures. The advantages of FIR filters include: absolute guarantee of stability, well-behaved round-off error characteristics, ability to realize arbitrary frequency responses, linear phase, etc. The disadvantages of FIR filters are basically just delay time (particularly if using the linear phase structure) and computational load, which increases directly with impulse response length, which in turn grows as the "precision" of the filter increases. By "precision", I am referring to how closely the actual filter's frequency response varies from an "ideal" one, in such areas as ripple and transition band width.

Typically, FIR filters are implemented as a sum of products. This is easily conceptualized as follows: Because the filter is a linear system, we may use superposition principles to manipulate our analysis of it. Consider the input sequence as a sum of many different sequences, each one consisting of all zeroes except for one unique point. In other words, the input to the system is a sum of many different impulses, each with its own amplitude and unique position in time. Therefore, the output of the filter will be simply a sum of many copies of the impulse response, each one shifted to a unique starting point in time, with a corresponding amplitude gain factor. If we look at any given output point, we note that it will have contributions from many of these various copies of the impulse response. In fact, if the length of the impulse response (the number of points between the first and last non-zero ones, inclusive) is N samples, then every input point will affect N output points, and conversely the output at any point in time will be affected by N input points (the current one and the N - 1 previous ones).

As a simple example, suppose the impulse response is only 3 samples long, and their values are 0.4, 0.3, 0.2. This means that each input point will contribute 0.4 times its own amplitude to the current output point, 0.3 times to the next one, and 0.2 times to the following one. Turn this around and we can see that each output point is the sum of 0.4 times the current input point, 0.3 times the previous input point, and 0.2 times the input point from 2 sample periods prior. The structure normally used to implement this is shown in Figure 1. It demonstrates both of these relationships: as each input point travels down the series of delay taps, it will contribute first 0.4, then 0.3, and finally 0.2 times its own amplitude to the output value before being "forgotten"; conversely, the output is a sum of the 3 most recent inputs, with the weighting factors correspondingly applied.

To repeat an important point for emphasis: the longer the impulse response, the more multiplications and additions must be done in the same amount of time, each sample period.

One thing to keep in mind about an FIR filter: it is not necessarily a "filter" in the normal sense of the term. It is, in fact, just a circuit or system that convolves an input signal with some finite impulse response. The impulse response, in turn, is just a series of delayed impulses, which create a series of delayed copies of the original input signal, multiplied by various weighting factors (coefficients, or you could even call them gain factors). Therefore, a sufficiently long FIR "filter" could perform functions normally thought of as "delay-line" functions, or be used to realize any arbitrary impulse response for purposes other than what would normally be called "filtering". Again, the only drawback is the computational load normally involved in implementing long FIR filters.

All of this brings me to the main point of this paper: It is possible, using some mathematical trickery, to perform long FIR filter functions with much less computational load than would normally be required using a direct sum-of-products implementation. In the following sections, we will explore these tricks, and some of the various functions that can be performed with this structure.

## FAST CONVOLUTION THEORY

In the preceding section, we noted that the amount of computation that must be performed in order to implement an FIR filter structure grows in a direct linear fashion with the length of the filter's impulse response: one multiply and one add per filter tap per input point. Fortunately, FIR filters with long impulse response lengths may be performed with far less computational overhead using Fourier transform techniques. The basic idea at work here is the exploitation of the transform theorem, which states that "Convolution in the time domain is equivalent to multiplication in the frequency domain."

In order to perform multiplication in the frequency domain, we must be able to convert our signals back and forth between the time domain and the frequency domain. In order to do this, we will use the Fast Fourier Transform (FFT), the theory and development of which is beyond the scope of this paper. Once we have used the FFT to convert our input signal and our impulse response to their frequency domain representations, we will multiply them together to obtain the frequency domain representation of our output signal, and then perform an inverse FFT operation on that to obtain the time-domain output signal.

This sounds simple enough, but in order to properly exploit this phenomenon, we must understand some additional concepts: the difference between linear convolution and circular convolution, and the two methods that may be used, overlap-and-add and overlap-and-discard. Also, it must be shown that this method, which certainly sounds more complicated than the direct time-domain sum-of-products implementation, is in fact more efficient in terms of computational load than that method.

Linear convolution is what we normally think of first when we think of convolution. It is what was described above in the "Convolution Theory and Filter Structures" section. Each input point, when passed through the filter and convolved with the impulse response, will contribute to N (where N is the number of taps, or length, of the impulse response) output points, starting with the one at the same time, and including N - 1 more following it. In this case, we can assign a second meaning to the "linear" in "linear convolution". We can say that it also means that the output sequence stretches out "linearly", i.e. in a straight line, in time following the end of the input sequence.

Let's look at an example. Suppose the length of our "filter" is 100. Therefore, each input point will effect 100 points, the "current" one and the 99 ones following it. Another way to think of this is to say that the impulse response starts at t0 and goes to t99. Now, suppose we want to "filter" an input signal that is 400 samples long (from T0 to T399). We know that the output sequence will start at T0 with a contribution from the input point at T0, but it will also stretch out timewise past T399. This is because the input point at T399 will affect 100 output points, starting at T399 and 99 more following it out to T498. Overall, we can say that the 400 point input signal got "stretched" by 99 points by the "filter" to become a 499 point output signal. This is demonstrated in Figure 2.

Keeping that in mind, let's examine what we call "circular convolution". This is what happens when Fourier transform techniques are used to perform convolutions. The basic problem is that the transform operates on a constant number of points. That is, if we do a Fourier transform on 400 input points, we get a spectrum with 400 frequency "bins". If we then do an inverse Fourier transform on those 400 bins (even if we did the multiplication by the Fourier transform of the impulse response), we still end up with only 400 output points, not 499. What happened to the other 99 points? They ended up getting wrapped around back to the beginning of the 400 point buffer, as though it were a circular buffer with the end continuously spliced back to the beginning. These 99 points got added to the first 99 points of the output buffer, rendering both the beginning and the end of the signal invalid (there is no way to separate them back out, without just recalculating the whole thing a different way anyway).

To see how this circular convolution comes about, look at Figure 3. Keep in mind that a finite duration Fourier series carries with it the implicit assumption that it has sampled one period of a periodic waveform, as shown in Figure 3a. Figure 3b shows the results of a linear convolution on each of the periods of this implied periodic waveform. Note that the period of the waveform is still 400 samples, but the length of each slice of it has grown to 499 points. Therefore, the end of one slice overlaps the beginning of the next slice. Figure 3c shows the actual resulting 400 point output sequence. The extra 99 output points generated by the convolution have extended over into the next implied period, and the first 99 output points have been corrupted by the extended output of the preceding implied period. Since each implied period is identical, the net effect is that the extra points generated past the end of the sequence get wrapped around, circular buffer style, to overlap (and get added to) the beginning points of the output sequence.

The easy way to avoid the problems of circular convolution is to use Fourier transforms of a length that equals or exceeds the length of the expected resulting output sequence. In the case of our example, we expect an output of 499 points. Since we would like to maintain computational efficiency, we want to use the FFT to do our transforms (as opposed to just a direct form discrete Fourier transform). Since many FFT algorithms are based on a radix that is a power of two, a 512 point FFT would make a lot of sense. In order to use the longer length FFT, we simply need to append zeroes to the input signal to pad it to the proper length. Figure 4 shows this being done. Figure 4a shows the input signal, padded to length 512, and its FFT. Figure 4b shows the impulse response, padded to length 512, and its FFT. Figure 4c shows the result of multiplying the two FFTs in order to get the FFT of the output sequence, which is then derived by performing an inverse FFT operation.

This is all very convenient for operating on short, fintie duration input sequences, but what happens if we have a signal that is practically infinite (i.e. expected to go on for a very long time relative to the length of the impulse response) and we want to start getting results out now? Then we break up the input signal into shorter segments for immediate processing. This is where the techniques known as overlap-and-add and overlap-and-discard get used.

The overlap-and-add method is the easier one to use to explain this concept, so that's where we'll start. To use the example we have been using, suppose we have an impulse response with a length of 100 and we want to segment the input sequence into chunks of length 400 and use FFTs of length 512 (actually, we will segment the input sequence into chunks of length 413 without any problem). We simply take each segment of the input sequence, pad it out to length 512, and process it as before. Note that each chunk

produces an output that extends into the time of the next chunk. In this region of overlap, we must add the results of the two overlapping segments to get the actual values of the output sequence. What has happened is this: we have created a turn-on transient of invalid data (since it is missing the contribution from the final 99 points in the preceding segment) and a turn-off transient of invalid data (since it is only the contribution from the preceding points in this segment, and not the contribution from the "current" points in the following segment). In between we only have 314 points of valid data. In order to get 413 valid output points from the 413 valid input points we started with, we must add the turn-off transient from one segment to the turn-on transient of the following segment in order to get valid output points in the overlap region. This is shown in Figure 5.

The overlap-and-discard method is a little bit different from the overlap-and-add method. It allows us to do basically the same thing without having to do the extra addition steps in the overlap region. This is done by actually allowing circular convolution to occur. We still segment the input sequence every 413 samples, but now we take a whole 512 point sequence to do the processing on. As a result, we get 413 valid output points. The easy way to picture this is to realize that, by segmenting the input sequence in this way, we produce a sequence that would be 611 points long if it were linearly convolved: 99 points of turn-on transient, 413 points of valid output, and 99 points of turn-off transient. Since we are using a 512 point FFT, circular convolution will cause the turn-on and turn-off transients to overlap each other and become totally invalid. These 99 points get discarded, leaving us with our 413 valid output points, without having to do any addition in the overlap region (we did our overlapping in the input segmentation and circular convolution). This is shown in Figure 6.

To conclude our discussion of the fast convolution technique, let's examine the computational differences between this technique and the direct time-domain sum-of-products convolution technique. To continue with our example of a 413 point input sequence and a 100 point impulse response, it is easy to see that, since we must do a multiply for each "tap" in the filter for each output point we generate, we will do 51,200 multiplies, processing in the time-domain (if we are very smart about not processing input points that don't exist, we can get away with only 41,300 multiplies, or 100 multiplies per output point, the same result we would have if we were just continually processing an "infinite" input sequence).

To see how much processing is needed in the frequency domain, we first need to figure out how many multiplies are involved in doing a 512 point FFT.

If we implement it in radix-2 form, we do 9 passes, 256 butterflies per pass, and 4 multiplies (actually one complex multiply) per butterfly, for a total of 9216 multiplies. If we then say we have to do three FFTs, one on the input sequence, one on the impulse response, and the inverse one on the result of multiplying the other two, we get 3 X 9216 = 27,648 multiplies. If we also do 512 complex multiplies to multiply the FFTs, we add 2048 real multiplies for a grand total of 29,696 multiplies. This is not yet a very dramatic improvement.

Suppose, however, that we are processing a longer sequence (or a bunch of shorter ones) with the same filter. In this case, we can pre-compute the FFT of the impulse response just once and store it for use many times, just like we would store the impulse response for the time domain convolution, and not keep recalculating it from the filter design parameters for every sample. This reduces our number of multiplies to 20,480. Okay, we've gotten down to about 2 or 1.5 to 1 improvement. Still not great.

There is one final improvement we can make. Notice that we are convolving a real signal with a real signal and getting real results. Notice also that we are doing complex FFTs, which assume that the input sequence may be complex. Well, if we convolve a complex input signal with a real impulse response, we get a complex output signal whose real part is the convolution of the real part of the input sequence with the impulse response, and whose imaginary part is the convolution of the imaginary part of the input sequence with the impulse response. Note that the real and imaginary parts stay separate! (As long as the impulse response is real). We can exploit this by actually processing two input sequences at once (or two segments of a longer input sequence), by loading one in as the real part and the other as the imaginary part, doing complex FFTs and multiplies, and storing out the real part of the result as the first output sequence, and the imaginary part of the result as the second output sequence. Therefore, the computational load gets cut in half of what we just computed, for a total of 10,240 multiplies per 512 point input chunk. This is less than 25 multiplies per output point. Now we have an improvement of about 4 or 6 to 1!

The improvement gets even more drastic as the filter length increases. As an example, suppose the impulse response is 16,385 samples long! Also assume that we will be continually processing an essentially infinite input sequence and the FFT of the impulse response will be pre-computed and stored. The easy part of figuring out how to compare the two methods is to note that the time-domain method takes 16,385 multiplies per output point. If we use 32K point FFTs, we can process 16,384 output points at once in the real part,

and another 16,384 output points in the imaginary part. To do so takes 2 FFTs (one forward and one inverse), each one 16,384 butterflies per pass, 15 passes, and 4 real multiplies per butterfly, for 983,040 per FFT. Twice that is 1,966,080. Add another 131,072 to do the 32K point complex multiply, and get a grand total of 2,097,152 multiplies per 32,768 output points, or only 64 multiplies per output point. This represents an improvement of over 256 to 1!

A similar analysis done for an impulse response length of 131,073 points and using 262,144 point FFTs yields a result of only 76 multiplies per output point, an improvement of 1724.5 to 1 over the time domain requirements.

Note that the computational load, which grows in a direct linear relationship with filter length in the time domain method, grows very slowly as the filter length increases when using the frequency domain method. This is a major advantage of frequency domain processing: it allows us to perform certain tasks that we could not previously do in the time domain under specific practical constraints of computation time and hardware cost (e.g. relatively inexpensive real-time audio processing).

## APPLICATIONS

In the following sections, we will discuss how these convolution structures can be used to implement various functions, starting with ordinary filtering and including various functions other than what is normally thought of as filtering. For example, a pattern of delayed impulses, which can emulate the early reflections of a room's ambient response, can be created using convolution techniques. Ideally, this would provide no frequency coloration, and therefore this would not normally be thought of as a filter, but the FIR filter structure (implemented as a fast convolution using Fourier transform techniques) can be used to create it.

## INSTRUMENT EQUALIZATION

The first application area we will look at is one that is simply a filtering application. This application, which I will call "instrument equalization", is where frequency spectrum shaping is applied to a musical instrument (or voice or any other sound) during recording or sound reinforcement for the purpose of enhancing the sound in some (presumably) desirable way. Typically this is done by a graphic or parametric equalizer, or by built in tone controls in the instrument itself. Recently, digital equalizers have appeared on the market.

If this function were to be done utilizing the techniques presented here, we could obtain all of the advantages of the FIR filter structure, and avoid the disadvantage of greater computational load. The remaining disadvantage, processing delay, will be addressed in the "Limitations and Problems" section.

Calculating the impulse response, or filter coefficients, for a desired frequency response is not a trivial matter, but various techniques are well documented in standard digital signal processing textbooks [1,2] and are implemented in various filter design software packages available on the market.

One trap to avoid falling into is to think that one could implement the filter directly in the frequency domain (and not bothering to calculate the impulse response) by just taking the FFT of the signal, multiplying it by the desired frequency response, and taking the inverse FFT of the result to obtain the filtered signal. The reason why this will not work in general is because of circular convolution. Any arbitrary frequency response that one might apply in this manner will most likely transform (via inverse FFT) into an impulse response that would occupy the full length of the FFT window. In fact, a desired "ideal" frequency response would possibly have an infinitely long impulse response, which would wrap around on itself in a circular buffer manner as well. At any rate, circular convolution would occur, with two nasty side effects: the output would exhibit non-causal characteristics, in the sense that the circular convolution of an input event that occurs late in the FFT time window would cause a response to it to appear at an earlier time in that window; and each boundary point between FFT windows would have a splicing discontinuity, since the first point in a window would be affected by the points at the end of that window, and not by the points at the end of the previous window, at it should be for a proper linear convolution. Therefore, the best way to proceed is to use the existing filter design algorithms to create an impulse response of a certain length (and in the process making the frequency response non-ideal) and then proceed as described with an FFT window of longer length.

## ROOM EQUALIZATION

Another application that we may use these techniques for is in the area of room equalization. Traditionally, room equalization for sound reinforcement has been done with either graphic equalizers (to generally "level" the frequency response of the room) and/or parametric equalizers (to notch out particularly troublesome resonances). Using long "filter" lengths, it is possible to almost exactly eliminate the early reflections and resonances, essentially "de-convolving" the early parts of the impulse response of the room. This results in a greater clarity of sound, while preserving the overall later reverberation characteristics. The increased clarity is a result of the delay between the original "direct" sound and the onset of the reverberant field, similar to the use of the "pre-delay" parameter found on some digital reverb units on the market.

The basic problem here is to come up with some finite impulse response that will perform the desired function. There are two ways to come up with a solution. Both ways involve first sampling the impulse response of the room itself. Then, determine what early characteristics are undesirable and window the impulse response to include only this early part of the impulse response. We now have a finite impulse response that we wish to cancel. We must create another finite impulse response that will cancel it, i.e. will create just a single impulse when convolved with the previously windowed impulse response of the room. Actually, this is impossible to do exactly, since this "inverse" impulse response that we are trying to determine will almost always be infinite in length. However, in practical terms, we can almost always determine some finite impulse response that will provide an acceptable attenuation of the undesirable characteristics of the reverberant field.

The first way to create this inverse impulse response is to calculate it by brute force. That is, take the windowed impulse response of the room and try to reduce it to just the original impulse by adding or subtracting time shifted and amplitude scaled copies of itself in an iterative fashion. The coefficients that are determined as the amplitudes (and signs) of the shifted copies become the inverse impulse response. Eventually, the error signal that remains uncorrected gets shifted out timewise and reduced in amplitude to an acceptable degree.

The second way to create the inverse impulse response is to take the Fourier transform of the windowed impulse response, take its inverse, and perform an inverse Fourier transform to generate the inverse impulse response. In theory this works because multiplying the two transforms would give a transform with a constant value of 1, which corresponds to a single impulse when inverse transformed back to the time domain. In practice, this procedure has one serious pitfall: circular convolution! That's right, we are actually generating an inverse impulse response that works when CIRCULARLY convolved with the windowed impulse response. The only way to get this to work for a linear convolution on a real signal is to minimize the difference between the circular convolution and the linear convolution. To do this, we must first have a windowed impulse response whose inverse, while infinite in length, has some finite length beyond which there is some acceptably minimal amount of energy. The only way to determine this is probably trial and error. Once a length has been chosen, pad the windowed impulse response to that length with zeroes, then perform the transform, inversion, and inverse transform. The result will be a finite length impulse response that consists of the infinite inverse impulse response wrapped around on itself in a circular buffer fashion. Remember, though, that by picking a sufficiently long transform, the "tails" of this impulse response (the portions from distant times that got wrapped around) that cause error in the cancellation have been kept to a minimum level. Now, to perform the fast convolution on an actual signal (our normal operating condition after having determined the inverse response), we will of course have to use a transform length that is even longer still.

## LOUDSPEAKER EQUALIZATION

A recent trend in the hi-fi audio industry is to make loudspeakers with increasing amounts of functionality integrated in, such as self-powered speakers or "digital" speakers that have a digital-to-analog converter (DAC), power amp and volume control integrated in. It should now be possible to build self-equalizing speakers as well, in order to provide the best reproduction possible from a given driver. The idea here is to use the same approach as described for "Room Equalization", in which the impulse response of the speaker is inverted for convolution with the actual input signal to be reproduced.

Two basic application approaches present themselves here. The first is to sample the impulse response of the speaker (drivers and enclosure) in an anechoic chamber, compute the inverse, and permanently store it in the

"brain" of the speaker when shipped from the factory, so that the speaker system itself is as good as it can be. The other approach is to let the consumer perform the "calibration" routine by placing a microphone at the preferred listening location, plugging it in to the speaker after installation in the preferred position, and pushing a button on the back of the speaker, which causes it to run its own calibration routine. This way, the combination of the speaker, its position, and the room acoustics is corrected for in the optimal way (again, limiting the length of the cancellation so that it only has to correct for resonances, early reflections, and frequency response variations, and not the entire reverberant field of the room until the sound is completely gone.

Note that this technique only corrects linear imperfections in the speaker (resonances, reflection and diffraction effects, frequency response variations) and not non-linear effects such as harmonic and inter-modulation distortions. Also note that it can only correct for a given listening position (in the room mode) or axis (in the anechoic chamber mode) and can not correct simultaneously for variations in room acoustics or axial response.

## AMBIENCE EQUALIZATION

One more application area we will look at, which is the one that requires the longest "filter" length, is that of ambience, or reverberation, synthesis. In this case, the long "filter" impulse response would be the pattern of reflections that make up the reverberant field. In this case, the actual reverberation density, or number of reflections per unit time, could be made as heavy or light as desired. The exact placement of every single reflection could also be selected. Certain extra tricks could be played, such as creating binaural pairs of impulses that would impart a more natural directionality to each reflection, rather than just producing "stereo" reverb, with reflections occurring randomly on either the left or right side. It would even be possible to "sample" the reverberant field of an actual space for use. In the extreme, a binaural sample could be taken from the "best seat in the house" for each of many stage positions. Each instrument in a multi-track recording could be processed with a different reverb sample, corresponding to its desired stage position, resulting in an extremely realistic aural image.

To really live up to the name "ambience equalization", and not just ambience generation, we could add the concept of "room equalization" and come up with an impulse response that would first de-convolve a room's own lousy reverberant field, and then generate the desired ambient field. To do this, we could either first generate the room-equalizing de-convolving impulse response by either of the two methods previously described, and then convolve it with the desired ambient response; or we could directly generate it by the brute force method previously described, where we would shift and scale the room response in an iterative fashion to create the closest approximation to the desired response, rather than the closest approximation to a dry impulse as before.

## REAL-TIME IMPLEMENTATION

As we saw in a preceeding section, using Fourier transform techniques to perform fast convolutions greatly reduces the computational load compared to performing the convolutions directly in the time domain. The question we wish to address now is: is this lower computational load one that is practical to implement? The answer is very emphatically yes!

Taking the last example given in the comparison of computational loads, suppose we do have an impulse response of length 131,072. At any typical digital audio sampling rate in the range of of 44.1 to 48 or even 50 KHz, this would represent a period of 2.6 to 2.97 seconds, enough for processing a respectable reverb sample.

Assuming a 50 KHz sampling rate, we would have 20 usec to perform the calculations necessary for each output point. In the time domain, each output point requires 131,072 multiplies (and adds). This would require hardware capable of performing a multiply (and accumulate) in 152 picoseconds! This is definitely not practical with today's technology.

However, in the frequency domain, we only require 76 multiplies per output point for this long impulse response. This only requires hardware capable of performing a multiply in 263 nsec. Today's single chip digital signal processing (DSP) microprocessors have typical multiplication cycle times on the order of 100 nsec or less. This is enough to easily handle the multiplication requirements, but the algorithm does require other processing.

For example, the radix-2 butterfly in the FFT requires 6 ALU operations (adds and subtracts) for each 4 multiply operations. Therefore, we need to find a DSP integrated circuit (IC) that can process FFTs efficiently, i.e. use its multiplier most of the time, not letting it sit idle while the IC is busy doing other things.

One such family of DSP microprocessors is the Zoran family of Vector Signal Processors. With a clock rate of 25 MHz, the multiplication throughput time is 80 nsec. Other parts of its architecture make it ideally suited for processing FFTs and vector multiplies. There is a dual ALU architecture embedded in the execution unit that allows the radix-2 butterfly to be performed in only 4 clock cycles, rather than 6. There is a separate bus interface unit that can perform bit-reversed addressing, and dual internal RAM sections that allow data I/O to be performed concurrently with FFT execution. Another feature of the architecture is that it only uses one external bus, for both instructions and data, and only uses it a fraction of the time while performing FFTs at full speed. This allows multiple processors to share a common bus and still run in parallel without fighting for bus time. In fact, FFT and vector multiply operations are both single instructions in the instruction set!

As an example, the ZR34325, a 32-bit floating point Vector Signal Processor, will perform a 1K complex point FFT (including all I/O and bit-reversed addressing for data re-ordering) in 1.732 msec, or about 21650 internal clock cycles (at 80 nsec). This FFT operation itself requires: (10 passes) X (512 butterflies/pass) X (4 multiplies/butterfly) = 20480 multiplies. This means that the '325 achieves almost 95% utilization of the multiplier bandwidth while performing an FFT. This results in an effective multiplier throughput rate that is still under 100 nsec, which is more than twice as fast as necessary for the example mentioned above.

The examples shown in Figures 7, 8, and 9 were performed on the ZR34161, a 16 bit block floating point/integer member of the Zoran Vector Processor family. Figure 7 shows a short "bump" (representative of a percussive sound) being convolved with a pattern of randomly scattered and weighted impulses (representative of an early reflection pattern in a reverb application). This example allows one to see very clearly the effect of the convolution in creating the many duplicate copies of the input signal. Figure 8 shows a rectangular pulse being convolved with a shorter triangular impulse response so that the turn-on and turn-off transients may be clearly seen. Figure 9 shows the same thing as Figure 8, except the rectangular pulse input occupies almost the entire length of the FFT. This allows us to see the results of circular convolution very clearly, since the turn-on and turn-off transients now overlap each other at the beginning of the output buffer area. All three examples use FFTs of length 4096; only the real parts of the complex signals are plotted (except Figure 9b, where the zero imaginary part is also plotted to provide a reference baseline at 0, since the real signal never gets down that low, for easy comparison to Figure 8c), both the real and imaginary parts of the FFTs in Figure 7 are plotted.

## LIMITATIONS AND PROBLEMS

As with any technique that appears to do great and wonderful things on the surface (and reducing computational requirements by a factor of over 1700 should certainly qualify as wonderful!), there are certain disadvantages involved here.

The first disadvantage is that of processing pipeline delay for performing the FFT based convolution. In the time domain, we can have immediate results—an output point is calculated from the current and N - 1 most recent input samples. However, in the frequency domain, since we are calculating blocks of output points by segmenting and overlapping the input data, we introduce delay. In our 100-tap filter example, we are taking input segments every 413 points. If we look at the first point in one of those segments, we see that we can't start the processing for it until 412 more input points have come in. Then, there is more delay while we do the processing—yes, we do very few multiplies per output point, but we have to process the whole block at once, so we have a lot of multiplies to do before we get the results that include that first output point that corresponds to that first input point we were talking about.

For certain specific applications, this delay would be objectionable, for instance in sound reinforcement for live music, if the impulse response used is long enough to create a noticeable delay. In other cases, a delay would not be a significant problem. Examples would include: a self-equalizing speaker for home use with a recorded or broadcast source, in which there is no live action to refer to; or any sound reinforcement application using a sufficiently short impulse response, and therefore negligible delay.

Another interesting application area that would not find the delay objectionable is the digital audio workstation. As an example, suppose you were experimenting with various reverbs on a track, or even doing "ambience equalization" to fix a recording with bad source room acoustics. As long as the processing was done at a rate that is at least as fast as the playback rate, the pipeline delay would only add a few seconds to the "rewind time" between playbacks with different settings. Since the processing is done in digital scratchpad memory, any other tracks could easily be synced with the delayed processed track. The reverbed track could be stored in another area of scratchpad memory, so that subsequent playbacks of the same version would have no extra delay. Compare this with other reverb approaches: typical digital reverb techniques use several recirculating delays with cross-feedback to create many reflections, but while this is also easily done in real time with relatively cost-effective hardware, the reflection patterns are still synthetic and not real samples of actual spaces; time-domain convolution of a real reverb characteristic could also be done in software on whatever microprocessor happens to be in the workstation, but it would be excruciatingly slow when compiling a track with a new reverb characteristic of any appreciable length.

In those applications where the delay is a problem, there is one obvious solution that has its own other problem. If we simply reduce the number of input and output points that we are processing in one chunk, we do reduce the delay that comes from waiting for a whole lot more points. The problem is, we still have to do the same amount of processing (using an FFT that is still bigger than the length of the impulse response) for that smaller number of points. Therefore, the computational advantage of fewer multiplies per point is lessened.

In fact, it can be shown that the greatest computational advantage for a given FFT size occurs when the number of points processed and the number of filter taps are both approximately half the FFT size. For example, suppose we are using 256 point FFTs and we convolve 128 input points with 129 taps. The equivalent number of multiplies in the time domain is 128 X 129 = 16512. If instead we convolved 192 input points with 65 taps, the equivalent number of time domain multiplies (using the same number of actual multiplies to do the frequency domain processing) is only 192 X 65 = 12480. Similarly, increasing the number of taps for the same length FFT reduces the number of points that can be processed, also reducing the computational efficiency advantage.

The converse is not true, however. For a given impulse response length, we get greater computational advantage as the FFT size gets bigger. To continue the example, 2 FFTs and a complex vector multiply of length 128 take

4096 multiplies; with length 256 they take 9216 multiplies (an increase of 2.25X). However, with a 65 tap filter, we can either process 128 input points (64 in each half, real and imaginary) or 384 input points (192 in each half). This is an increase of 3X the number of points. Therefore, the number of multiplies per point goes down from 32 to 24! The problem with this observation is that increasing the FFT length increases the pipeline delay problem, as well as the noise problem, which will be discussed shortly.

The other way to reduce the pipeline delay is to segment the impulse response. In this method, we would take a short input segment and separately convolve it with each of the segments of the impulse response. The first result, or "current" segment, i.e. the result of the convolution with the first segment of the impulse response, would get played immediately; the remaining results ("future segments") would be saved for future use. While playing the "current" segment, we would also have to add to it the previously saved future output segments (which came from older input segments and later impulse response segments) that correspond to the same time slot. This reduces the pipeline delay by as much as desired, but does have the drawback of sacrificing some of the computational advantage. For instance, if we took our example of the 131,073 (128K + 1) tap filter that needed 76 multiplies per output point, and broke it up into 8 of the 16385 (16K + 1) tap filters at 64 multiplies per point, it would appear that we would need 8 X 64 = 512 multiplies per point (and an additional 7 adds per point). This is a lot more than 76, but still a whole lot less than 131,073.

Actually, that number can be improved substantially be removing some redundant calculation. First of all, when doing 8 convolutions with the same input points, we only need to do the forward FFT on the input segment once. Analyzing the requirements of the 16K convolution, we determine that, of the 64 multiplies per point required, 30 are for the forward FFT, 30 are for the inverse FFT, and 4 are for the multiplication of FFTs. By only doing the forward FFT once, we save 7 X 30 = 210 multiplies per output point.

Secondly, instead of saving the final convolution results of each segment, why not just save the multiplied FFTs? Then, when it comes time to play back a certain segment in time, we would first add together all the multiplied FFTs that correspond to that time segment, and then perform just one inverse FFT per output segment. This is permissible because of the principles of linearity and superposition. We will do more extra adds, since we will be adding complex vectors, not real ones, but we will save an additional 7 X 30 = 210 multiplies per output point, for a total of 512 - 210 - 210 = 92 multiplies per output point. This number could also be derived as

follows: 30 for one forward FFT per segment, 30 for one inverse FFT per segment, and 8 X 4 = 32 for the 8 multiplications of FFTs. 30 + 30 + 32 = 92, as expected. Not that much of a penalty after all!

Another technique to increase throughput is the use of multiple processors in parallel. One nice feature of the Zoran Vector Signal Processor family mentioned earlier is that multiple processors may be put on the same bus without sacrificing performance. Multiple bus architectures may also be used with multiple processors.

The other important practical limitation involved in frequency domain convolution is the same one encountered by any complex digital signal processing task, build-up of excessive computational noise. This is caused by rounding off multiplication results, especially when those rounded off results get passed through more multiplication stages, such as what happens in an FFT—each butterfly pass reuses the rounded off results of the previous pass. This is especially a problem when using 16-bit devices, such as the Zoran ZR34161 which was used to generate the examples shown previously. Figure 10 shows the computational noise that results in the "quiet space" between the bumps in the output of the example in Figure 7. Note that the amplitude scale has been greatly magnified, but the result would still be audible noise. The obvious answer to this problem is to go to a device that has greater precision, such as the new members of the Zoran Vector Signal Processor family: the ZR34322, which has a 32 bit integer/block floating point data format, and the ZR34325, which has a 32 bit single precision (IEEE-754 compatible) floating point data format (much greater dynamic range, but only 24 bits of precision). Determining which of those two data formats is better for this application is beyond the scope of this paper, but either one should yield a dramatic improvement in the computational noise performance, compared to a 16 bit integer machine.

## SUMMARY

In summary, it has become feasible to attack an old set of problems (those requiring equalization in some form or other) with an old set of theoretical magic (discrete-time frequency domain transform theory), using a semi-old trick (Fast Fourier Transform algorithms) and new hardware (integrated circuits that can perform this stuff in real time).

In fact, the scope of the "problems requiring equalization" that can be attacked in this manner has been expanded by the efficiency of this approach. Specifically, the problems that are more often thought of as time domain problems, attacked with delay line solutions (like reverb generation), can now be solved in the frequency domain using a "filter" approach.

In this paper, we have explored the theory behind this approach, determined its computational advantages, shown how it can be used in a variety of applications using today's technology, and explored some of the problems that arise, with proposed solutions.

## REFERENCES

[1] L.R. Rabiner and B. Gold, Theory and Application of Digital Signal Processing, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1975.

[2] Alan V. Oppenheim and Ronald W. Schafer, Digital Signal Processing, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1975.

[3] Zoran; Digital Signal Processors Data Book, Zoran Corp. 3450 Central Expressway, Santa Clara, CA, 1987.

[4] Zoran; ZR34325 Vector Signal Processor Product Description, Zoran Corp. 3450 Central Expressway, Santa Clara, CA, 1986.

[5] Zoran; ZR34322 Vector Signal Processor Product Description, Zoran Corp. 3450 Central Expressway, Santa Clara, CA, 1986.

[6] Zoran Technical Note: Fast Convolution, TN92046, Zoran Corp. 3450 Central Expressway, Santa Clara, CA, 1987.

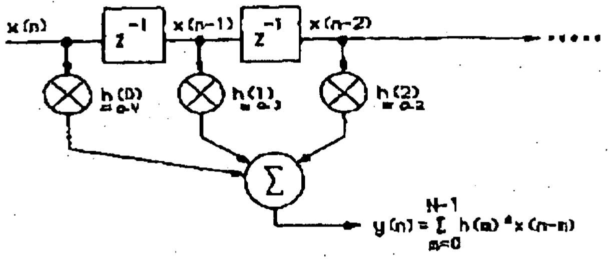


Figure 1. Typical FIR filter structure (3 taps).

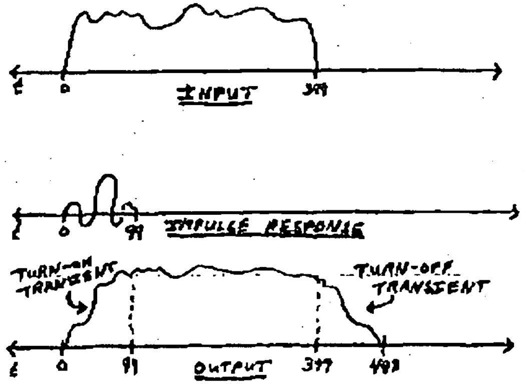


Figure 2. Linear Convolution, 400 Point Input, 100 Point Impulse Response.

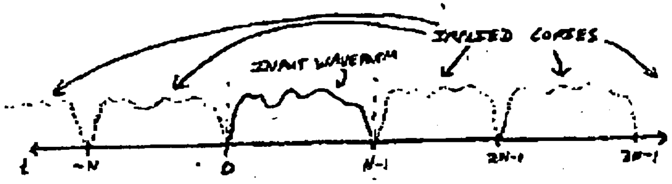


Figure 3a. Implied Periodic Waveform for Fourier Analysis

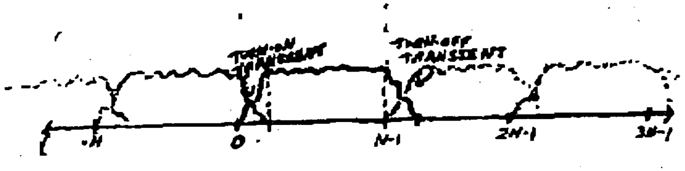


Figure 3b. Linear Convolution On Implied Periodic Waveform

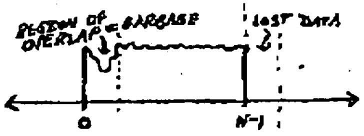


Figure 3c. Resulting Waveform

Figure 3. Circular Convolution

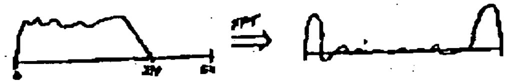


Figure 4a. Input Sequence Padded to 512 Points, and Resulting Spectrum.

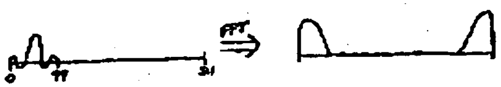


Figure 4b. Impulse Response Padded to 512 Points, and Resulting Spectrum.

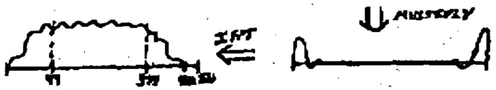


Figure 4c. Output Sequence Derived From Multiplying Above Spectra.

Figure 4. Linear Convolution Using FFTs On Padded Sequences

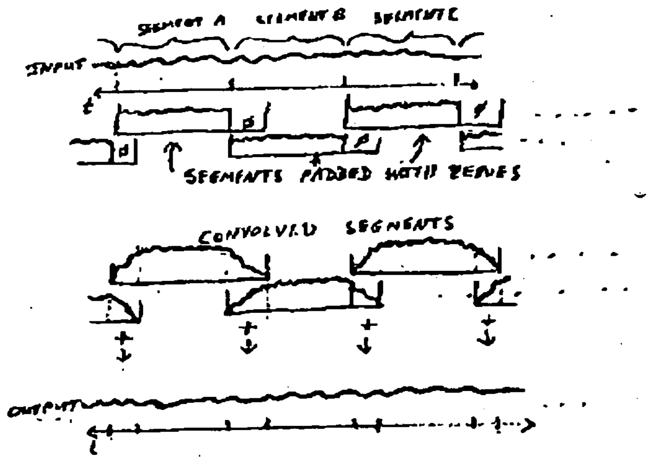


Figure 5. Overlap-And-Add Method for Segmenting Long Sequences

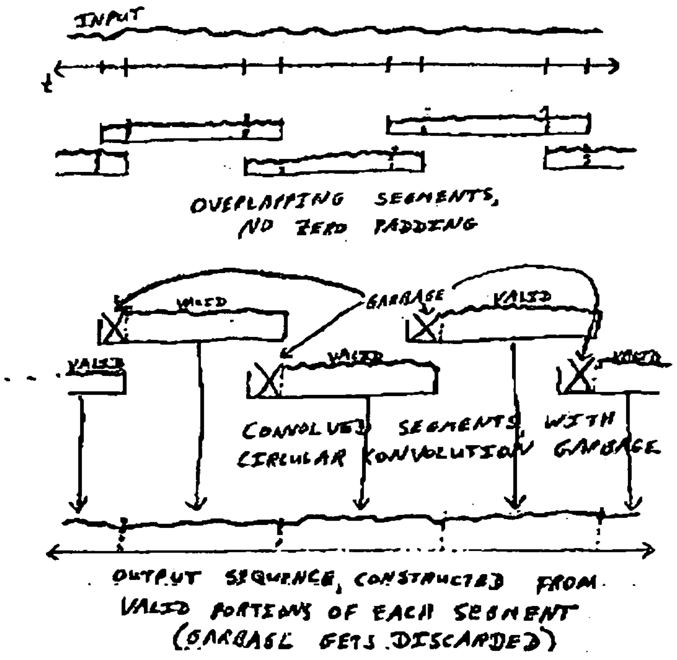


Figure 6. Overlap-And-Discard Method for Segmenting Long Sequences

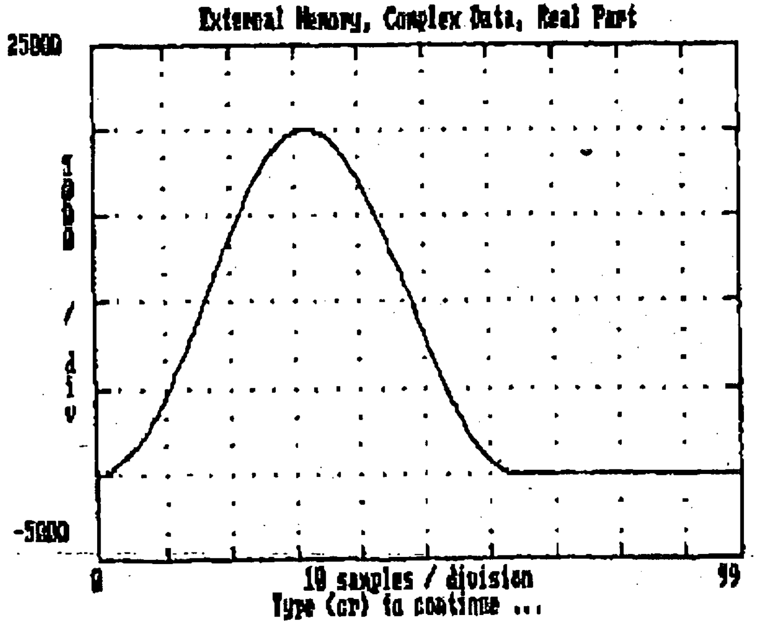


Figure 7a. 64 Point "Bump" Input (Time-scaled)

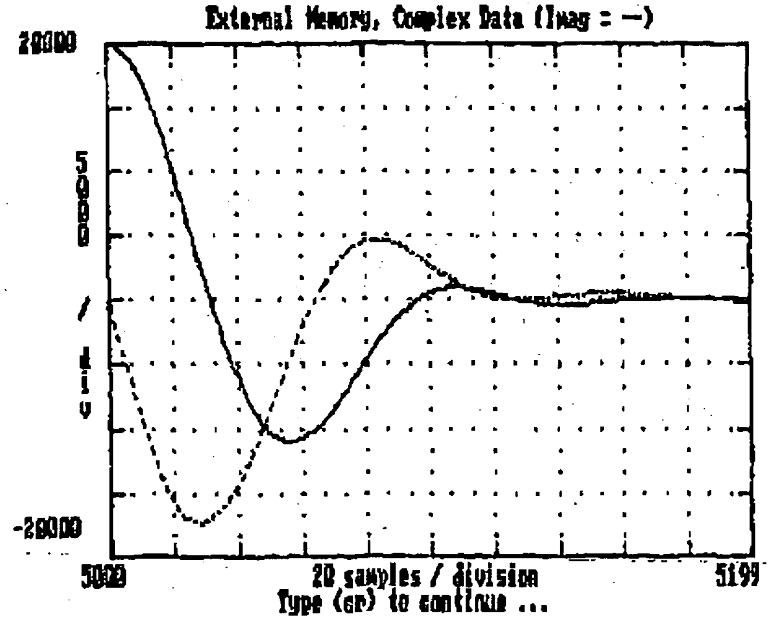


Figure 7b. Partial Display of Spectrum of Bump

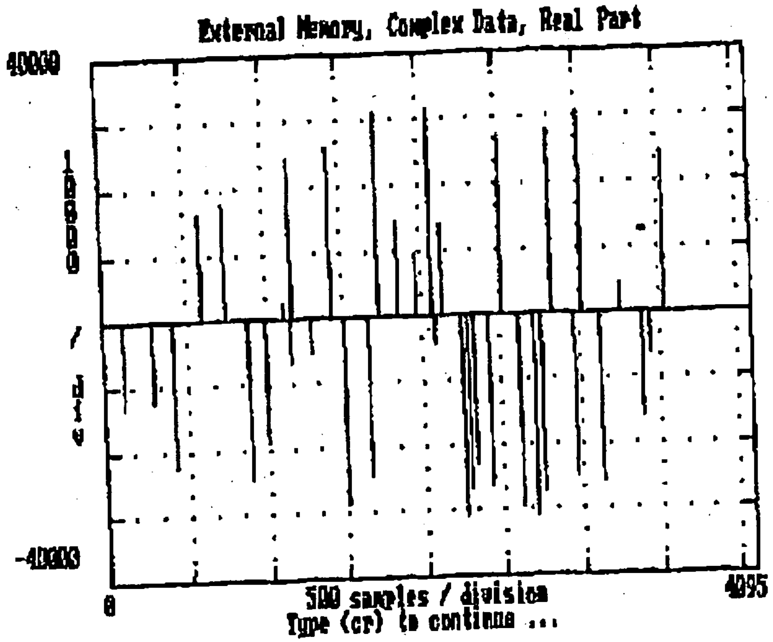


Figure 7c. Early Reflection Pattern

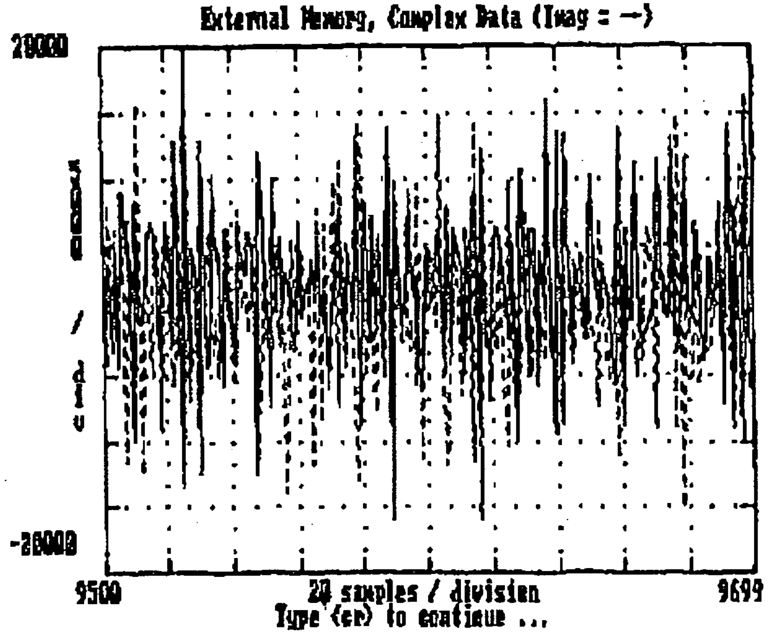


Figure 7d. Partial Display of Spectrum of Reflection Pattern

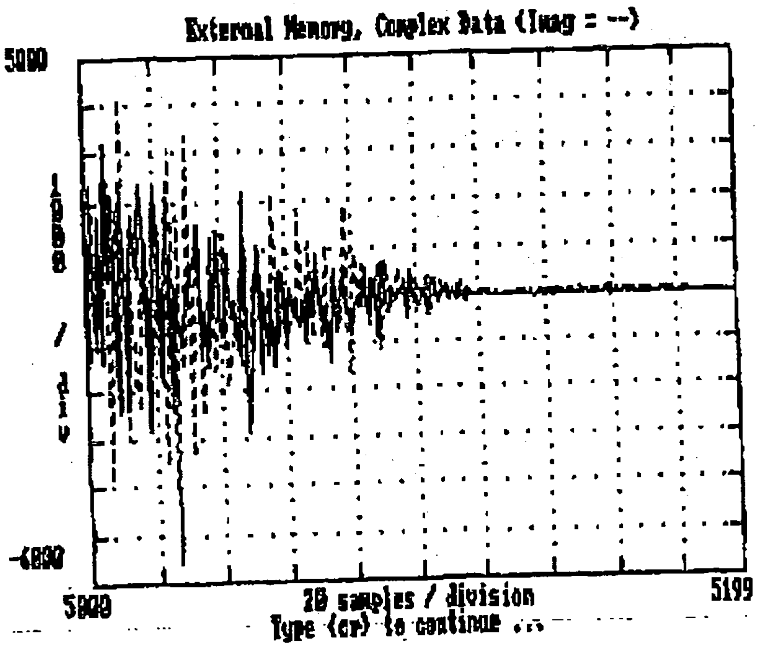


Figure 70. Partial Display of Multiplied FFTs

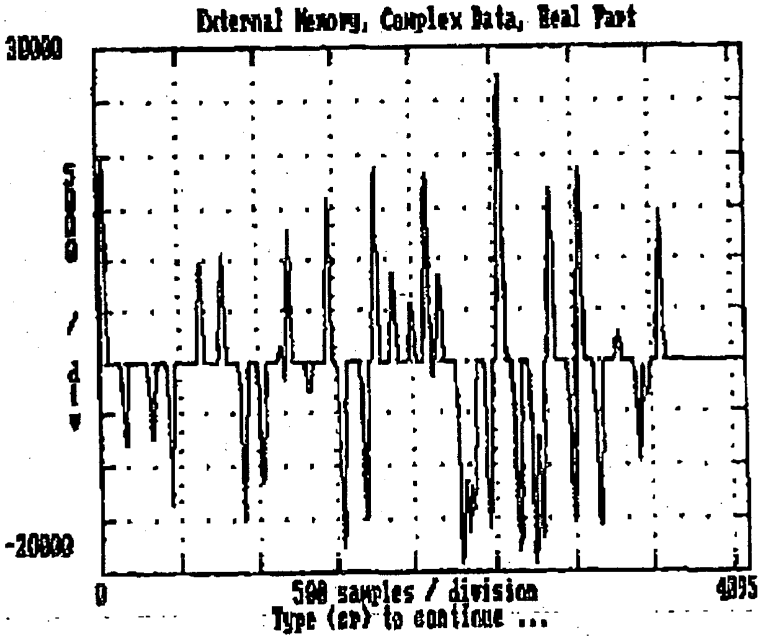


Figure 71. Result of Convolution

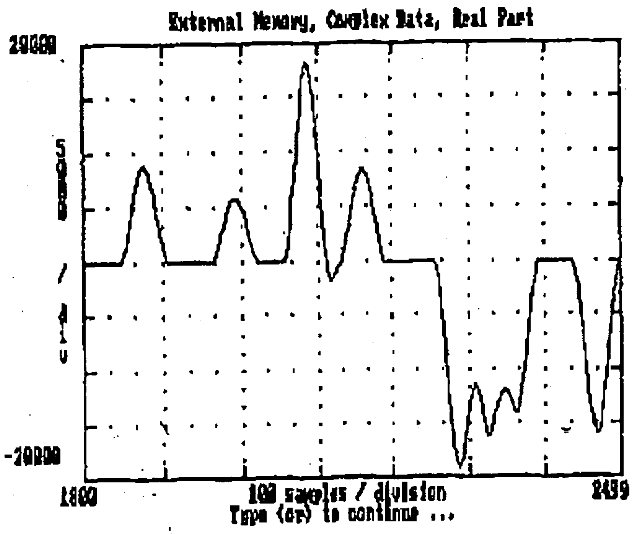


Figure 7g. Close-up Display of Portion of Result (Time-scaled)

Figure 7. Fast Convolution of "Bump" Signal with Reflection Pattern

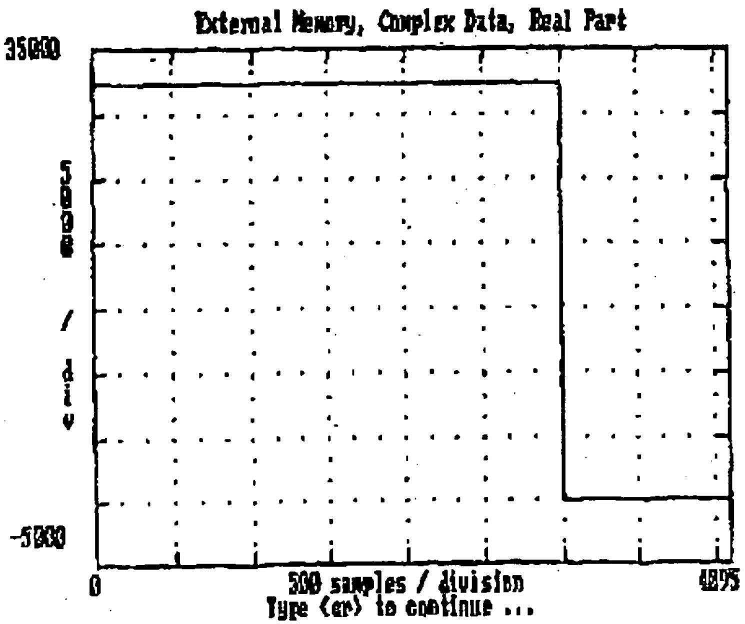


Figure 6a. Rectangular Pulse

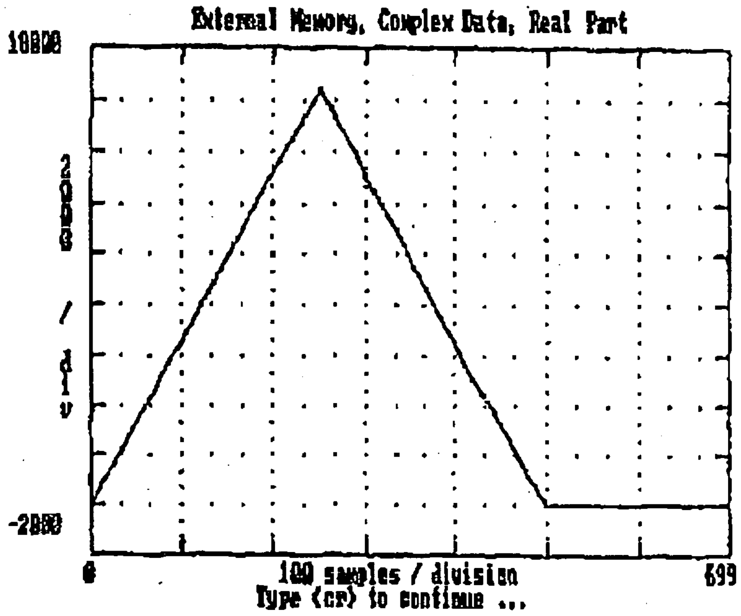


Figure 6b. Triangular Impulse Response (Time-scaled)

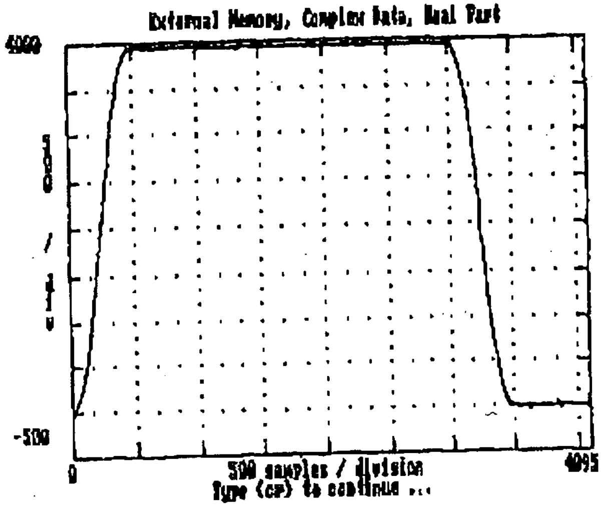


Figure 6a. Fast Convolution Result

Figure 8. Fast Convolution of Rectangular Pulse with Triangular Pulse

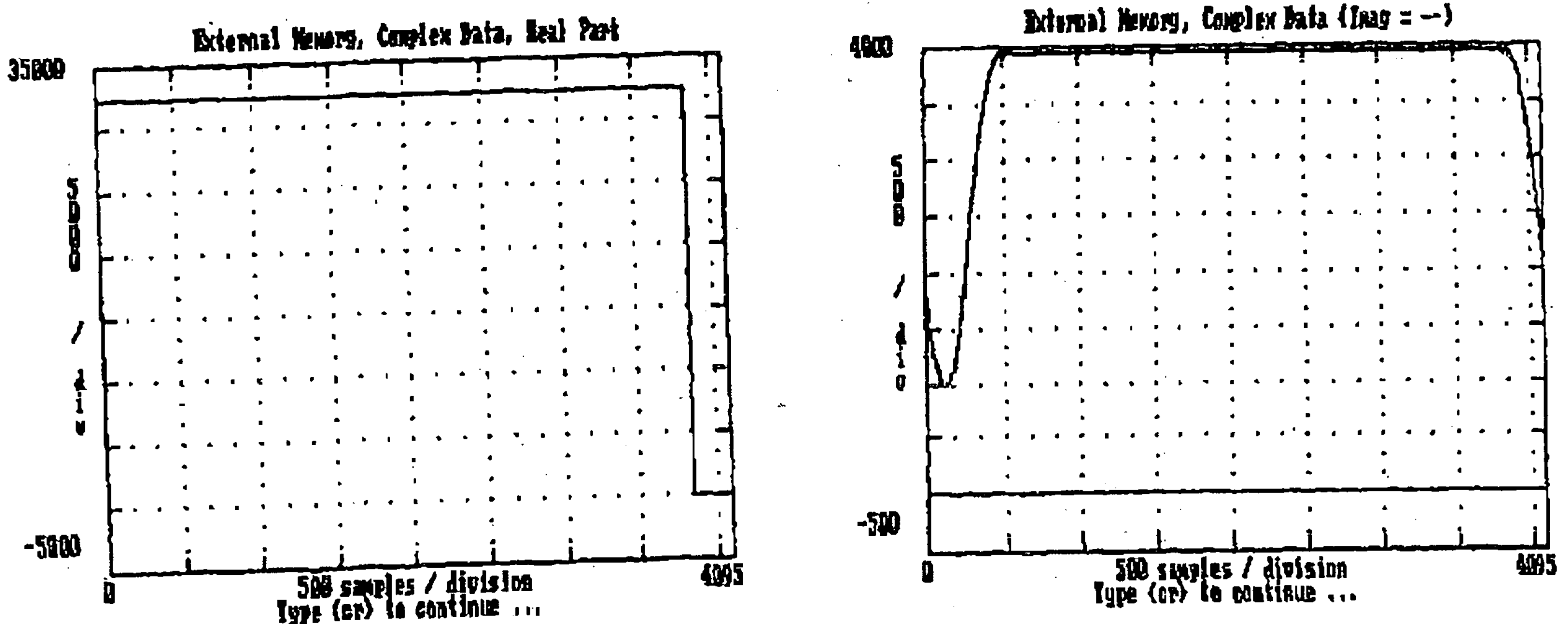


Figure 9a. Longer Rectangular Pulse

Figure 9b. Result of Circular Convolution With Triangular Pulse in Figure 9b.

Figure 9. Circular Convolution Example

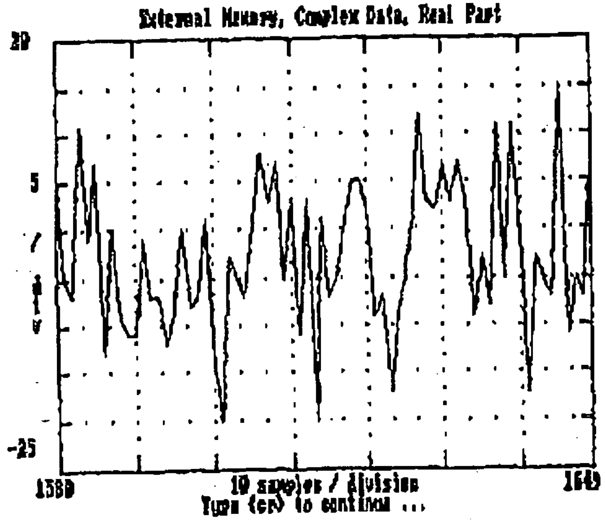


Figure 10. Computational Noise Resulting Between Bumps in Figure 7f.

A5

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) Publication number: **0 250 048 A1**

# (12) EUROPEAN PATENT APPLICATION

(21) Application number: 87201133.3

(51) Int. Cl.⁴: **H03H 21/00**

(22) Date of filing: 16.06.87

(30) Priority: 20.06.86 NL 8601604

(43) Date of publication of application:
23.12.87 Bulletin 87/52

(84) Designated Contracting States:
BE DE FR GB IT NL SE

(71) Applicant: N.V. Philips' Gloeilampenfabrieken
Groenewoudseweg 1
NL-5621 BA Eindhoven(NL)

(72) Inventor: Sommen, Petrus Christianus Wilhelmus
c/o INT. OCTROOIBUREAU B.V. Prof. Holstlaan 6
NL-5656 AA Eindhoven(NL)
Inventor: Claasen, Theodoor Antonius Carel Maria
c/o INT. OCTROOIBUREAU B.V. Prof. Holstlaan 6
NL-5656 AA Eindhoven(NL)
Inventor: Van Gerwen, Petrus Josephus
c/o INT. OCTROOIBUREAU B.V. Prof. Holstlaan 6
NL-5656 AA Eindhoven(NL)
Inventor: Kotmans, Hendrik Jan
c/o Int. Octrooibureau B.V. Prof. Holstlaan 6
NL-5656 AA Eindhoven(NL)

(74) Representative: Van den Brom, Arend Albertus et al
INTERNATIONAAL OCTROOIBUREAU B.V.
Prof. Holstlaan 6
NL-5656 AA Eindhoven(NL)

## (54) Frequency-domain block-adaptive digital filter.

(57) A frequency-domain block-adaptive digital filter (FDAF) having a finite impulse response of length N for filtering a time-domain input signal in accordance with the overlap-save method includes window means (11) for obtaining modifications (B(p;m)) of the 2N frequency-domain weighting factors (W(p;m)) from correlation products (A(p;m)). A known FDAF of this type contains five 2N-points FFT's, two of which are used in the window means (11). By utilizing a special time-domain window function which can be implemented very efficiently in the window means (11) with the aid of a frequency-domain convolution, a FDAF of this type containing only three 2N-point FFT's is obtained whose convergence properties are comparable to those of the known FDAF containing five 2N-point FFT's.

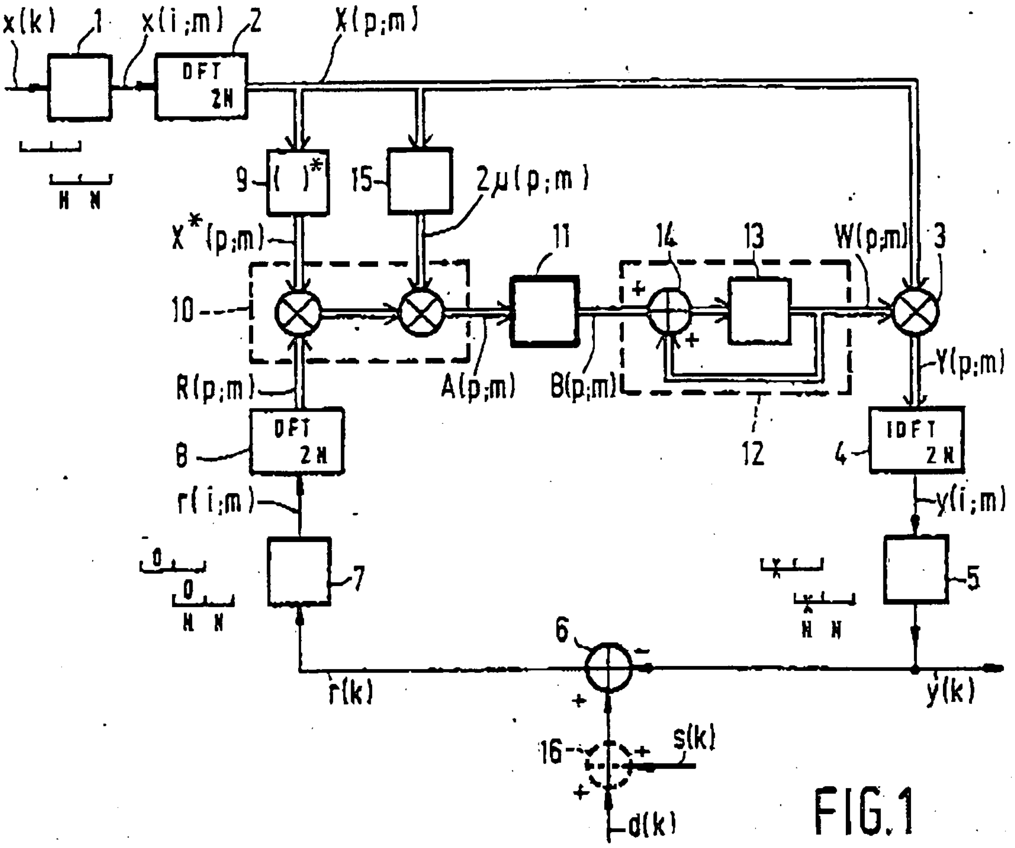

x(k)
1
x(i;m)
2
DFT 2N
X(p;m)
N N
9
( )*
15
2µ(p;m)
X*(p;m)
10
11
14
13
W(p;m)
3
+
+
R(p;m)
A(p;m)
B(p;m)
Y(p;m)
12
DFT 2N
8
IDFT 2N
4
r(i;m)
y(i;m)
0
0
N N
7
5
N N
6
−
+
r(k)
y(k)
16
+
+
s(k)
d(k)

FIG.1

## Frequency-domain block-adaptive digital filter.

### (A) Background of the invention

The invention relates to a frequency-domain block-adaptive digital filter having a finite impulse response of length N for filtering a time-domain input signal in accordance with the overlap-save method and having a structure as described in the introductory part of Claim 1.

A frequency-domain adaptive filter (FDAF) having such a structure is disclosed in the article "A Unified Approach to Time-and Frequency-Domain Realization of FIR Adaptive Digital Filters" by G.A. Clark et al. in IEEE Trans. Acoust., Speech, Signal Processing, Vol. ASSP-31, No. 5, October 1983, pages 1073-1083, more specifically Figure 2.

In the field of speech and data transmission, time-domain adaptive filters (TDAF) are used in the majority of cases and in most practical applications these TDAF's are implemented as adaptive transversal filters, in which a "least-mean-square" (LMS) algorithm is used for adapting the weights. When the length N of the impulse response assumes large values, as is the case with applications in the acoustic field, the TDAF implemented as a transversal filter has the advantage that the complexity in terms of arithmetic operations (multiplying and adding) per output sample increases linearly with the filter length N. In addition, the TDAF implemented as a transversal filter has a low convergence rate for highly correlated input signals such as speech and certain types of data, since the convergence rate decreases with an increasing ratio of the maximum to minimum eigenvalues of the correlation matrix of the input signal (see, for example, C.W.K. Gritton and D.W. Lin, "Echo Cancellation Algorithm", IEEE ASSP Magazine, April 1984, pp. 30-38, in particular pp. 32/33).

The use of frequency-domain adaptive filters (FDAF) provides the possibility to significantly improve the convergence properties for highly correlated input signals, as for any of the substantially orthogonal frequency-domain components of the input signal the gain factor in the adaptation-algorithm can be normalized in a simple way in accordance with the power of the relevant frequency component. For the most efficient implementation of a FDAF having an impulse response of length N, use is made of Discrete Fourier Transforms (DFT) of length 2N of 2N weighting factors to ensure that circular convolutions and correlations, computed with the aid of DFT's, are equivalent to the desired linear convolutions and correlations when the sectioning method is performed correctly. For large values N the computational complexity can, however, be significantly reduced in terms of arithmetic operations per output sample by utilizing efficient implementations of the DFT known as "Fast Fourier Transform" (FFT), as a result of which this complexity becomes proportional to the logarithm of the filter length N.

There where a TDAF needs only have N weighting factors for an impulse response of length N, the equivalent FDAF must utilize 2N weighting factors. After convergence, the weighting factors of an adaptive digital filter (TDAF and FDAF) will continue to fluctuate around their final values due to the presence of noise or other types of signals superimposed on the reference signal and because of the precision (that is to say word length or number of bits) with which the different signals in the digital filter are represented. With the customary, practically valid assumptions about the statistic independence of the different quantities in the filter, the weighting factors will have the same variances when no use is made of window functions in the adaptation loop of the filter. This implies that at the same convergence rate of the adaptive filter (that is to say the same gain factor in the adaption algorithm) using 2N instead of N weighting factors results in an increase of the final misalignment noise factor of the filter by 3 dB, since the final misalignment noise factor is determined by the sum of the variances of the weighting factors. In practice, the gain factor in the adaptation algorithm is chosen such that a predetermined value of the final misalignment noise factor is not exceeded. In order to compensate for the increase of the final misalignment noise factor in an FDAF, this gain factor must be halved, which causes the convergence rate also to be halved, whereas in the majority of applications a highest possible convergence rate is pursued.

Said article by Clark et al. described a solution for this problem with reference to Figures 2 and 3, the modifications of the 2N frequency-domain weighting factors not being derived directly from the second multiplier means, but by using window means for performing an operation whose time-domain equivalent is a multiplication by a rectangular window function of length 2N which constrains the last N components to be zero. An implementation of this window function in the time-domain requires the use of 2 DFT's, namely an inverse DFT for the transformation to the time-domain and a DFT to effect the transformation to the frequency-domain after multiplication by the time-domain window function. An alternative implementation is based on the consideration that a multiplication in the time-domain is equivalent to a convolution in the frequency-domain with the components of the DFT of length 2N of this time-domain window function. For high values N this alternative implementation in the frequency-domain is not attractive, as its computational complexity per component increases linearly with N, whereas in the first-mentioned time-domain implementation this complexity becomes proportional to the logarithm of N when the 2 DFT's are implemented as FFT's. Thus, a preferred implementation of the known solution results in an FDAF containing a total of 5 DFT's implemen ted as FFT's.

## (B) Summary of the Invention.

The invention has for its object to provide a frequency-domain block-adaptive digital filter of the type defined in the opening paragraph of section (A) whose computational complexity is significantly reduced compared to that of the known FDAF containing 5 DFT's, by utilizing a priori information about the global shape of the impulse response to be modelled, but whose convergence behaviour is comparable to that of the known FDAF containing 5 DFT's.

The frequency-domain block-adaptive digital filter according to the invention is characterized, in that the window means are arranged for convolving the 2N frequency-domain products with a function having one real and two mutually conjugate complex coefficients and corresponding to a time-domain window function g(k) of length 2N defined by:

$g(k) = (1/2)\,[1 + \cos((k-k_0)\,\pi/N)]$

for $k = 0, 1, \ldots, 2N-1$, where $k_0$ is a constant with $0 \leq k_0 < N$

## (C) Short description of the drawings.

The invention and its advantages will now be described in greater detail by way of example with reference to the accompanying drawings. Therein:

Figure 1 shows the general block diagram of an FDAF including window means and utilizing the overlap-save method;

Figures 2a and 2b show the block diagram of the window means for the implementation of a window function in the time-domain and the implementation equivalent thereto, in the frequency-domain, respectively;

Figures 3a and 3b show the time-domain window function according to the afore-mentioned prior art and the invention, respectively:

Figure 4 is a time diagram to illustrate the convergence behaviour of an FDAF in accordance with the afore-mentioned prior art and the invention, respectively.

## (D) Description of the embodiments.

Figure 1 shows the general block diagram of an FDAF having a finite impulse response of length N for filtering a time-domain digital input signal x(k) in accordance with the overlap-save method. Double-line signal paths in Figure 1 indicate paths in the frequency-domain and single-line signal paths indicate paths in the time-domain. Transformation from the time-domain to the frequency-domain and vice-versa is effected with the aid of the Discrete Fourier transform (DFT) and the Inverse Discrete Fourier transform (IDTF), respectively, both having a length 2N. In literature these transforms are known as 2N-point DFT's, wherein "point" may refer to both a discrete time-domain component and to a discrete frequency-domain component. To differentiate between time-domain and frequency-domain signals, time-domain signals are written in lower case letters and frequency-domain signals in upper-case letters.

The FDAF shown in Figure 1 has for its object to derive, at discrete instants k, from the input signal x(k) an output signal y(k) which equals a reference signal d(k) as well as possible. In many case, for example, when the FDAF forms part of an echo cancellor, this reference signal d(k) may be assumed to be the linear convolution of the input signal x(k) with an impulse response h(k) of length N whose shape is not accurately known. Then the FDAF has for its task to make its impulse response w(k) equal to this impulse response h(k) as well as possible.

To that end, In Figure 1 the Input signal x(k) is applied to sectioning means 1 In order to be segmented into blocks of length 2N with the aid of serial-to-parallel conversion. each block overlapping its preceding block over a length N, as is shown symbolically in Figure 1. The points of the input signal block having block number $\underline{m}$ are denoted as x(i:m) with i = 0, 1, ..., 2N-1. With the aid of transformation means 2 for performing a 2N-point DFT each input signal block is transformed to the frequency-domain, the frequency-domain points of block $\underline{m}$ thus obtained being denoted as X(p;m) with p = 0, 1 ..., 2N-1. In multiplier means 3 each frequency-domain component X(p;m) is multiplied with an associated frequency-domain weighting factor w(p;m) for forming products X(p;m) W(p;m) representing the frequency-domain components Y(p;m) of the output signal block $\underline{m}$. With the aid of transformation means 4 for performing an Inverse 2N-point DFT each output signal block Is transformed to the time-domain, the resultant time-domain points of block $\underline{m}$ being denoted down as y(i;m) with i = 0, 1, ... 2N-1. Since the weighting factors W(p;m) may be considered as being points of a 2N-point DFT performed on time-domain weighting factors w(i;m) which represent values of the Impulse response w(k) during block $\underline{m}$, the multiplication in multiplier means 3 in the time-domain corresponds to the circular convolution of the input signal x(k) during block $\underline{m}$ with the impulse response w(k) during block $\underline{m}$. The desired time-domain output signal y(k) Is, however, the linear convolution of x(k) with w(k). In accordance with the overlap-save method, this desired output signal y(k) is obtained by applying the time-domain components y(i;m) of this circular convolution for each block to sectioning means 5 In which, using parallel-to-serial conversion, the first N point y(i;m) with i = 0, 1, ..., N-1 are discarded and the last N points y(i;m) with i = N, N+1, ..., 2N-1 are passed on as the output signal y-(k), as is symbolically shown in Figure 1.

For the adaptation of the frequency-domain weighting factors W(p;m) on a block-by-block basis, use is made of a "least mean-square" (LMS) algorithm. In accordance with this algorithm, these weighting factors W(p;m) are modified as long as there is correlation between the input signal x(k) and an error signal r(k) given by the difference between the reference signal d(k) and the output signal y(k). This differential signal r(k) = d(k) - y(k) is obtained with the aid of an adder 6. The overlap-save method for determining this correlation between the signals x(k) and r(k) implies that in Figure 1 the error signal r(k) is applied to sectioning means 7 In order to be segmented into blocks of length 2N with the aid of serial-to-parallel conversion, each block overlapping its preceding block over a length N and the first portion of length N of each block being constrained to be zero, as is symbolically shown in Figure 1. The points of error signal block $\underline{m}$ are denoted as r(i;m) with i = 0, 1, ..., 2N-1, it holding for i = 0, 1, ..., N-1 that r(i;m) = 0. Using transformation means 8 for performing a 2N-point DFT each error signal block is transformed to the frequency-domain, the frequency-domain points of block $\underline{m}$ thus obtained being denoted as R(p;m) with p = 0, 1, ..., 2N-1. In addition, the frequency-domain components X(p;m) of input signal block $\underline{m}$ are applied to conjugation means 9 for producing the complex conjugate value X * (p;m) of each component X(p;m). Each conjugated component X * (p;m) is multiplied in multiplexer means 10 with the associated component R(p;m) for forming frequency domain products X* (p;m) R(p;m) which correspond to the time-domain correlation between the input signal x(k) and the error signal r(k) during block $\underline{m}$. In addition, each product X* (p;m) R(p;m) is multiplied by an amount 2 µ(p;m), where µ(p;m) is the gain factor in the adaptation algorithm, as a result of which a product A(p;m) = 2 µ(p;m) X* (p;m) R(p;m) is formed which determines the modification of the frequency-domain weighting factor W(p;m). In the FDAF shown in Figure 1, these products A(p;m) are applied to window means 11 for obtaining the ultimate modifications B(p;m) of the weighting factors W(p;m) which are formed by means of accumulator means 12. These accumulator means 12 include a memory 13 for storing the weighting factors W(p;m) of block $\underline{m}$ and an adder 14 for forming the sum of each weighting factor W(p;m) and its associated modification B(p;m), this sum being stored in memory 13 for providing the weighting factors W(p;m+1) for the sub sequent block (m+1)..

For performing its main function (effecting that its impulse response w(k) converges to the impulse response h(k) to be modelled, an FDAF utilizing the overlap-save method need not to include the window means 11 of Figure 1. In that case the modifications B(p;m) of the weighting factors W(p;m) are given by the products A(p;m) and the adaptation algorithm may be written as: W(p;m+1) = W(p;m) + 2 µ(p;m) X* (p;m) R(p;m) (1) When the input signals are not or only weakly correlated, the gain factor µ(p;m) for each weighting factor W(p;m) may have the same constant value α which is independent of the block number $\underline{m}$ (this constant α is known as the adaptation factor of the algorithm). For highly correlated input signals the convergence rate can be significantly increased by decorrelating these input signals, which can be effected by normalizing their power spectrum (see, for example, the aforementioned article by Gritton and Lin, page 36). Since the frequency-domain components X(p;m) are already available in an FDAF, the normalization can be effected in a simple way by making the gain factor µ(p;m) equal to the adaptation factor divided by the power | X(p;m) |² of the relevant component X(p;m):

µ(p;m) = α/ |X(p;m) |² (2)

This possibility is shown in Figure 1 by including means 15 in which for each applied component X(p;m) the right-hand side of formula (2) is determined which, optionally after smoothing on a block-by-block basis, with the aid of a simple recursive filter, is used as a gain factor μ(p;m) for multiplying by 2 μ(p;m) in the multiplier means 10.

As has already been mentioned in the foregoing, an FDAF must utilize 2N-point DFT's and 2N frequency-domain weighting factors W(p;m) for providing an impulse response w(k) of a length N. These weighting factors W(p;m) may be considered to be the points of a 2N-point DFT performed on 2N time-domain weighting factors w(i;m). This creates a problem because the FDAF has only N time-domain weighting factors w(i;m) which represent the N values w(k) with k = 0, 1, ..., N-1 of the impulse response of the FDAF, so that the remaining N time-domain weighting factors w(i;m) with i = N, N+1, ..., 2N-1 are actually superfluous. It is likewise described that, after convergence, the 2N weighting factors W(p;m) and consequently also the 2N weighting factors w(i;m) will continue to fluctuate around their final values as a result of the finite precision of the signal representation in the FDAF and because of the presence of noise and any other types of signals superimposed on the reference signal d(k). This superpositioning is symbolically shown in Figure 1 by a broken-line adder 16 inserted in the path of the reference signal d(k) and receiving a signal s(k) representing noise and any other type of signals. The variances of the fluctuating weighting factors will all have the same value when the FDAF does not include window means 11 and the gain factors have been chosen in the manner described in the preceding paragraph. An important parameter for the convergence behaviour of a block-adaptive filter (FDAF and TDAF) is the ratio β (m) of the variance of the residual signal d(k) - y(k) in block m to the variance of the signal s(k) superimposed on signal d(k) in block m. The final value β after convergence is known as final misalignment noise factor and this final value β is predominantly determined by the sum of the weighting factor variances (at the customary values of the adaptation factor α). The fact that N out of the 2N time-domain weighting factors w(i;m) of the FDAF are actually superfluour implies that the final value β for the FDAF is indeed unnecessarily higher by a factor of two (3 dB) than the final value β for the equivalent TDAF having only N time-domain weighting factors. For the customary values of the adaptation factor α this final value β is substantially proportional to α and the same holds for the convergence rate of β(m) as long as β(m) is much greater than β. In practice, the adaptation factor α is chosen such that a predetermined final value β is not exceeded. This implies that the adaptation factor α for the FDAF must actually be halved unnecessarily to satisfy this prescription resulting in the convergence rate also being halved, which is undesirable.

This problem can be solved by including the window means 11 in the FDAF of Figure 1. In accordance with the article by Clark et al. mentioned in the foregoing, these window means 11 can be arranged for performing an operation on the frequency-domain products A(p;m) supplied by multiplier means 10 that is equivalent to multiplying the associated time-domain products a(i;m) with a rectangular window function of length 2N which forces the last N time-domain products a(i;m) to be zero. Figures 2a and 2b show the block diagram of these window means 11 such as they are described in the article by Clark et al. (see Figure 3). In the window means 11 of Figure 2a the time-domain window function g(k) is realized with the aid of transformation means 17 for performing an inverse 2N-point DFT converting the 2N frequency-domain products A(p;m) into the 2N associated time-domain products a(i;m), a multiplier 18 multiplying each time-domain product a(i;m) with the value g(i) of window function g(k) for k = i to obtain 2N time-domain products b(i;m) = g(i) a(i;m) and transformation means 19 for performing a 2N-point DFT converting the 2N time-domain products b(i;m) into the associated 2N frequency-domain products B(p;m). These products B(p;m) constitute the modifications of the 2N frequency-domain weighting factors W(p;m) applied to the accumulator means 12 of Figure 1. The window means 11 of Figure 2b are based on the consideration that multiplying by g(i) in the time-domain as performed in Figure 2a is equivalent to a convolution in the frequency-domain and consequently consist of convolution means 20 for performing the (circular) convolution of the frequency-domain products A(p;m) with the frequency-domain components G(p) obtained by effecting a 2N-point DFT on the 2N points g(i) of the time-domain window function g(k) for k = i.

The rectangular window function g(k) of length 2N as described in the article by Clark et al. is given by the formula:

$$g(k) = \begin{cases} 1 & k = 0, 1, \ldots, N-1 \\ 0 & k = N, N+1, \ldots, 2N-1 \end{cases} \qquad (3)$$

and is shown in Figure 3a. For the sake of simplicity, the influence of this window function g(k) on the convergence behaviour of the FDAF will now be described in greater detail for the case in which the window means 11 of Figure 2a are used in Figure 1 and the memory 13 has zero content at the beginning of the adaptation. This last fact implies that for m = 0 the 2N frequency-domain weighting factors W(p;m) satisfy W(p;0) = 0, and consequently the associated time-domain weighting factors w(i;m) also satisfy w-(i;0) = 0. Each of the 2N weighting factors w(i;m) for m ≠ 0 may be considered to be the result of the block-by-block accumulation of its time-domain modifications starting from m = 0 and when the window means 11 of Figure 2a are used, these time-domain modifications are formed by the products b(i;m) = g(i)-a(i;m) at the output of multiplier 18. By using window function g(k) as defined by formula (3), the modifications b(i;m) with I = N, N+1, ..., 2N-1 and m = 0, 1, 2, ... are constrained to be zero and consequently the weighting factors w(i;m) with i = N, N+1, ..., 2N-1 for m ≠ 0 are also constrained to be zero because then w(i;m) = w(i;0) for these values i and because it holds for all values i that w(i;0) = 0. This implies that of the 2N weighting factors w(i;m) of the FDAF the N actually superfluous weighting factors w(i;m) with i = N, N+1, ..., 2N-1 will not fluctuate around the constrained value zero and consequently will not contribute to the final value $\beta$ of the parameter $\beta$(m). Halving the adaptation factor $\alpha$, necessary in the FDAF without window means 11 to prevent a prescribed final value $\beta$ from being exceeded, needs consequently not to be effected in the FDAF including window means 11 for realizing window function g(k) in accordance with formula (3) and consequently there is also no need to halve the convergence rate.

The convergence behaviour of the described FDAF's will now be illustrated with reference to simulation results for the case in which the impulse response h(k) of length N = 32 to be modelled has the global shape of an exponentially decreasing function, the input signal x(k) is a pseudo-ternary data signal in accordance with the AMI-code and is, therefore, a highly correlated signal, the value $\beta = 2^{-6}$ (= -18 dB) is chosen as the prescribed final value of the parameter $\beta$(m) and the value $\beta(0) = 2^{7}$ (= +21 dB) is chosen as the initial value. The simultation results are represented in a highly stylized form in Figure 4, which shows the parameter $\beta$(m) as a function of the number m of the iterations of the adaptation algorithm. This stylizing relates to both the convergence rate of $\beta$(m) and also to the details of the variation of $\beta$(m) for consecutive values of m. More specifically, the convergence rate of $\beta$(m) in Figure 4 has a constant value until the final value $\beta$ is reached and thereafter has zero value whereas this convergence rate is actually already noticeably smaller at values of $\beta(m) = 2^{-2}$ (= -6 dB) than at the initial value $\beta$(0) and still further decreases at still lower values of $\beta$(m); and, additionally, the details of the variation of $\beta$(m) for consecutive values of m have been omitted in Figure 4 to prevent the noisy character of these details from obscuring the image of the global shape of $\beta$(m) to too high an extent.

Curve a in Figure 4 relates to the FDAF without window means 11, the adaptation factor having a value $\alpha = 2^{-6}$ for reaching the prescribed final value $\beta$ = -18dB. Curve b in Figure 4 relates to the FDAF including window means 11 for implementing window function g(k) as defined in formula (3), the adaptation factor having a value $\alpha = 2^{-5}$ for reaching the prescribed final value $\beta$ = -18 dB; this adaptation factor $\alpha$ has a twice higher value than for the case in which the FDAF does not include window means 11. When this twice higher value $\alpha = 2^{-5}$ is also used in the FDAD without window means 11, $\beta$ (m) varies in accordance with curve c in Figure 4 which coincides with curve b until (m) has reached the value $\beta$ (m) = -15 dB, which at this value of $\alpha$ constitutes the final value $\beta$; consequently this final value $\beta$ = -15 dB exceeds the prescribed final value $\beta$ = -18 dB by 3 dB.

As regards the practical implementation of window means 11 for providing window function g(k) as defined by the formula (3), the embodiment of Figure 2a should be preferred over embodiment of Figure 2b, more specifically for large values N, as the computational complexity, expressed in the number of real multiplications and additions required for obtaining the 2N modifications B(p;m) of the weighting factors W-(p;m), is in the case of Figure 2a of a lower order than for the case illustrated in Figure 2b. Figure 2a requires two 2N-point DFT's which can efficiently be implemented as 2N-point FFT's, so that for large values N the number of arithmetic operations is of the order N log(N), the multiplication by g(k) in multiplier 18 not contributing to this number of operations since g(k) only assumes the values 1 and 0. In Figure 2b and 2N values A(p;m) are convolved with the values G(p) obtained by performing a 2N-point DFT on the 2N points g(i) of g(k) in accordance with formula (3) for k = i. As can be easily checked, G(p) has the value G-(p) = 0 for p = 2, 4, 6, ... and a value G(p) ≠ 0 for p = 0 and p = 1, 3, 5, ... So Figure 2b requires the convolution of 2N values A(p;m) with (N01) values G(p), so that for large values N the number of arithmetic operations is of the order of $N^2$ and consequently of a higher order than for the case illustrated in Figure 2a.

The FDAF's described so far do not utilize a priori information about the usually well-known global shape of the impulse response h(k) to be modelled and only utilize the a priori information about the length N of this impulse response. In contrast thereto, the FDAF according to the invention does indeed utilize a priori information about the global shape of the impulse response h(k) to be modelled, as usually it is not only known of this impulse response h(k) that the N values h(k) for k = 0, 1, ..., N-1 differ significantly from zero and are substantially zero for k ≥ N, but it is also roughly known for which values k the largest amplitudes | h(k) | occur. In addition, it is known, that the amplitudes | h(k) | have roughly a decreasing character for values k which increasingly differ from the values k with the largest amplitudes | h(k) |. The invention now utilizes this a priori information by arranging the window means 11 for providing a time-domain window function g(k) of a length 2N which is given by the formula:

$$g(k) = (1/2)\,[1 + \cos((k - k_0)\,\pi/N)] \quad (4)$$

for k = 0, 1, ..., 2N-1, where $k_0$ is a constant with $0 \leq k_0 < N$. By choosing this constant $k_0$ in the range of values k having the largest amplitudes | h(k) | it is accomplished that the window function g(k) defined in formula (4) has the same global shape as the impulse response h(k) to be modelled. In addition, the window means 11 are arranged for implementing the window function g(k) defined by formula (4) in the frequency-domain, that is to say for the convolution of the 2N frequency-domain products A(p;m) produced by multiplier means 10 in Figure 1 with the frequency-domain components G(p) obtained by effecting a 2N-point DFT on the 2N points g(l) of g(k) defined by formula (4) for k = l. As is easy to check, G(p) has the following values:

$$G(0) = N$$
$$G(1) = (N/2)\,[\cos(k_0\,\pi/N) - j\,\sin(k_0\,\pi/N)]$$
$$G(2N-1) = (N/2)\,[\cos(k_0\,\beta/N) + j\,\sin(k_0\,\pi/N)] \quad (5)$$
$$G(p) = 0,\ 2 \leq p \leq 2N-2 \quad (5)$$

Implementing time-domain window function g(k) defined in formula (4) in the frequency-domain with the aid of window means 11 as shown in Figure 2b consequently requires the convolution of 2N values A(p;m) with only 3 values G(p), namely the real value G(0) and the mutually conjugate complex values G(1) and G(2N-1), so that for high values N the number of arithmetic operations required for the 2N modifications B(p;m) of the weighting factors W(p;m) is only of the order N and consequently not of the order $N^2$, as in the case of the known window function g(k) as defined in formula (3). Using the measures according to the invention consequently accomplishes a significant reduction of the computational complexity. In this respect it should be noted that this reduction of the computational complexity is not achieved when the window function g(k) of formula (4) is implemented with the aid of window means 11 as shown in Figure 2a, because multiplying by g(k) then indeed contributes to the number of arithmetic operations, but particularly because the computational complexity of the two 2N-point DFT's implemented as FFT does not change, so that for large values N the number of computational operations remains of the order N log(N).

Now it will be explained that compared to the preferred embodiment of the known FDAF with a total of 5 2N-point DFT's implemented as FFT the significant reduction in the computational complexity of the FDAF according to the invention is not accompanied be a deterioration in the convergence behaviour. For simplicity, this explanation will be given with reference to observations in the time-domain.

In the foregoing it has already been stated that the FDAF actually has 2N time-domain weighting factors w(i;m) with i = 0, 1, ..., 2N-1. When the FDAF does not include window means 11 and the gain factors $\mu$(p;m) have been chosen in accordance with formula (2), the rate at which a weighting factor w(i;m) converges to the associated value h(i) of the impulse response h(k) to be modelled, will have for all weighting factors w(i;m) the same value which is proportional to the adaptation factor $\alpha$. Because of the use of time-domain window functions g(k) the convergence rate of a weighting factor w(i;m) becomes proportional to g(i). The convergence rate of the FDAF is a function of the convergence rate of all the weighting factors w(i;m), but will predominantly be determined by the convergence rate of those weighting factors w(i;m) which have to converge to the highest values h(i) of the impulse response h(k) to be modelled. By now choosing the constant $k_0$ in formula (4) in the range of the values k having the largest amplitudes | h(k) | it is accomplished that the value g(k) in that range is substantially equal to 1, that is to say it has substantially the same value as in the case in which no window functions g(k) are used (in fact, this latter case can also be characterized by a window function g(k) = 1 for k = 0, 1, ..., 2N-1). Chossing the window function g(k) in accordance with formula (4) then results in a convergence rate of the FDAF of the invention substantially equal to that of the FDAF without window means 11 when the adaptation factor has the same value for both cases.

As has also been mentioned in the foregoing, the variance of a weighting factor w(i;m) after convergence has the same value for all the weighting factors w(i;m) when the FDAF does not include window means and the gain factors $\mu$(p;m) have been chosen in accordance with formula (2). For the usual values of the adaptation factor $\alpha$ the value of this variance of w(i;m) is substantially proportional to $\alpha$.

However, when the window function g(k) is used, the variance of a weighting factor w(i;m) after convergence becomes proportional to $\alpha$ g(l). The final value $\beta$ of the parameter (m) is substantially determined by the sum of the variances of the weighting factors w(i;m) after convergence, so that when the window functions g(k) are used the final value $\beta$ is substantially proportional to the quantity

$$\alpha \sum_{i=0}^{2N-1} g(i) \qquad (6)$$

for the customary values of the adaptation factor $\alpha$. For the FDAF without window means 11 the quantity of formula (6) has the value $2\alpha N$ and for the FDAF with window means 11 this quantity has the value N when the window function g(k) is in accordance with formula (3), but also when the window function g(k) is in accordance with formula (4). This can be checked in a simple manner with reference to formulae (3) and (4), but also by making a comparison between Figure 3a representing g(k) of formula (3) and Figure 3b representing g(k) in accordance with formula (4) for $k_0 = 2$. Using the window functions g(k) as defined in the formulae (3) and (4) consequently results in the same final value $\beta$ of parameter $\beta$ (m) when the adaptation factor $\alpha$ has the same value for both cases.

The convergence behaviour of the FDAF according to the invention can again be illustrated on the basis of simulation results for the same case as used to illustrate the convergence behaviour of the known FDAF's. Maintaining the afore-mentioned stylizing, these simulation results are also represented in Figure 4 by curve d, which relates to the FDAF including window means 11 for implementing a window function g(k) in accordance with formula (4) with $k_0 = 0$, the adaptation factor having a value $\alpha = 2^{-5}$ for obtaining the prescribed final value $\beta$ = -18dB. This curve d substantially coincides with curve b in a large range from m = 0 onwards and has the same final value $\beta$ as curve b. Since curve b relates to the known FDAF with window means 11 implementing a window function g(k) as defined in formula (3) and also having an adaptation factor $\alpha = -2^{-5}$, Figure 4 is a clear illustration of the fact that the convergence behaviour of the FDAF according to the invention is in practice fully comparable to the convergence behaviour of this known FDAF. The deviations, shown in a stylized manner in Figure 4, between curve b and d for low values $\beta$-(m) are illustrative of the fact that in the final stage the convergence proceeds somewhat slower for curve d than for curve b. This is however of little practical importance, as the convergence in the final stage does indeed proceed slowly and the final value $\beta$ is the same for both curves b and d.

The preceding description can now be summarized as follows. For a FDAF in accordance with the overlap-save method, the article by Clark et al. describes a preferred implementation containing five 2N-points FFT's, two of which are used to constrain the last N-time-domain weighting factors to be zero by using an appropriate window function in the time-domain. By utilizing a priori information about the impulse response to be modelled, the invention provides a FDAF in accordance with the overlap-save-method having a special time-domain window function which can be implemented very efficiently in the frequency-domain, the FDAF only containing three 2N-point FFT's, whereas its convergence properties are comparable to those of the prior art implementation containing five 2N-point FFT's.

## Claims

1. A frequency-domain block-adaptive digital filter having a finite impulse response of length N for filtering a time-domain input signal in accordance with the overlap-save method, comprising:
-first sectioning means for segmenting the time-domain input signal into blocks of length 2N, each block overlapping its preceding block over a length N;
-first transformation means for obtaining 2N frequency-domain components of the Discrete Fourier Transform of length 2N of each input signal block;
-first multiplier means for multiplying each of the 2N frequency-domain components with an associated frequency-domain weighting factor for obtaining 2N weighted frequency-domain components;
-second transformation means for obtaining 2N time-domain components of the inverse Discrete Fourier Transform of length 2N of the weighted frequency-domain components;
-second sectioning means for discarding the first N time-domain components and conveying the last N time-domain components as the time-domain output signal of the filter;
-means for generating a time-domain error signal as a difference between the filter output signal and a

reference signal;
-third sectioning means for segmenting the time-domain error signal into blocks of length 2N, each block overlapping its preceding block over a length N, and for assigning the zero value to the first portion of a length N of each block;
-third transformation means for obtaining 2N frequency-domain components of the Discrete Fourier Transform of length 2N of each error signal block;
-conjugation means for forming the complex conjugate value of the 2N frequency-domain components produced by the first transformation means;
-second multiplier means for multiplying each of the 2N frequency-domain components produced by the third transformation means with the associated frequency-domain component produced by the conjugation means and with a gain factor for obtaining 2N frequency-domain products;
-window means for performing an operation on the 2N frequency-domain products, whose time-domain equivalent is a multiplication by a window function, to obtain 2N frequency-domain weighting factor modifications;
-accumulator means for accumulating on a block-by-block basis each of the 2N weighting factor modifications for obtaining the 2N frequency-domain weighting factors;
characterized in that
the window means are arranged for convolving the 2N frequency-domain products by a function having one real and two mutually conjugate complex coefficients and corresponding to a time-domain window function $g(k)$ of length 2N defined by
$g(k) = (1/2)\,[1 + \cos((k - k_0)\,\pi/N]$
for $k = 0, 1, ..., 2N-1$, where $k_0$ is a constant with $0 \leq k_0 < N$.

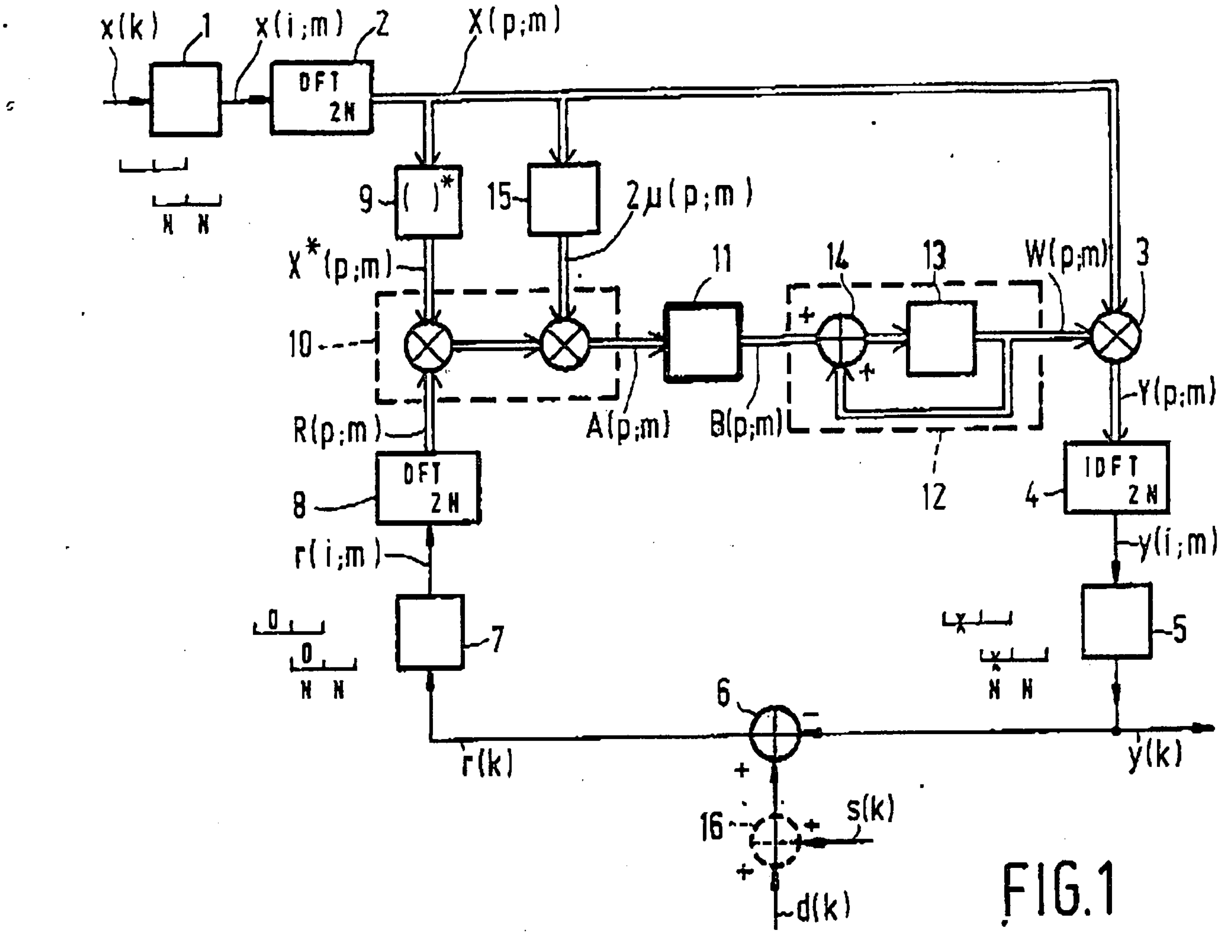


FIG.1

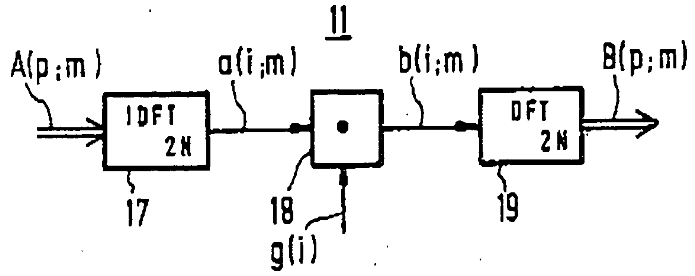


FIG.2a

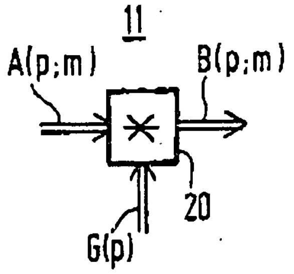


FIG.2b

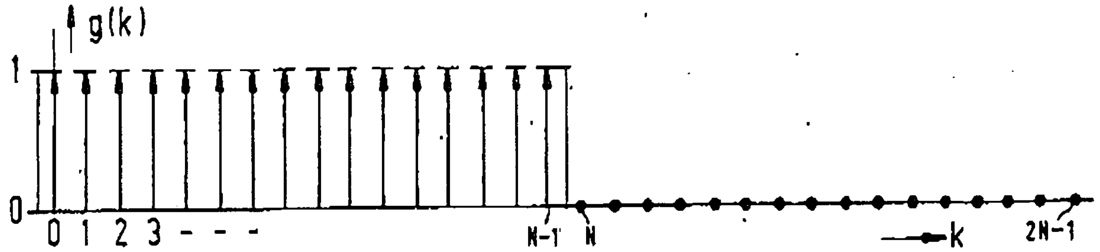


FIG.3a

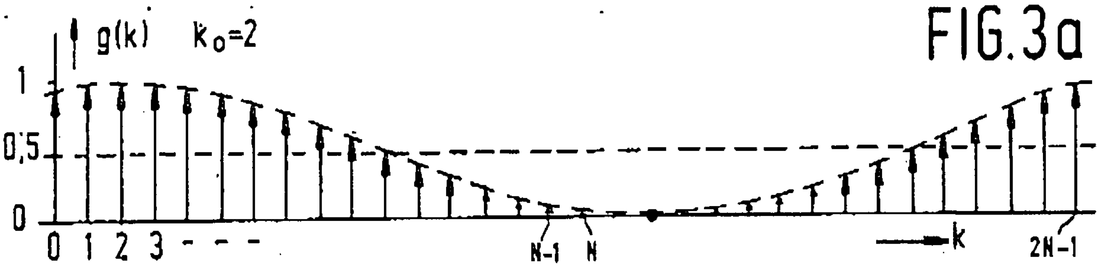


FIG.3b

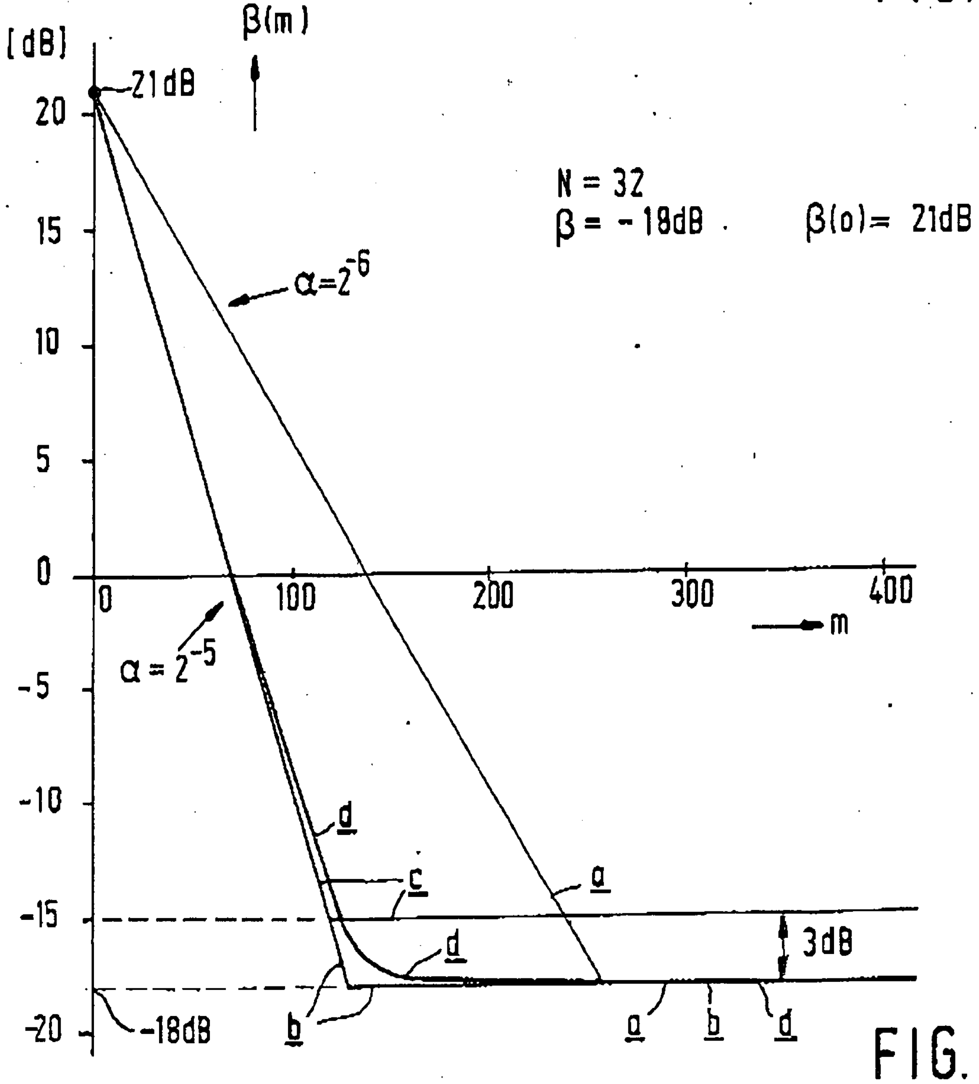


FIG.4

European Patent Office

# EUROPEAN SEARCH REPORT

Application number

EP 87 20 1133

## DOCUMENTS CONSIDERED TO BE RELEVANT

| Category | Citation of document with indication, where appropriate, of relevant passages | Relevant to claim | CLASSIFICATION OF THE APPLICATION (Int. Cl.4) |
|---|---|---|---|
| A | IEEE TRANSACTIONS ON ACOUSTICS, SPEECH AND SIGNAL PROCESSING, vol. ASSP-31, no. 5, October 1983, pages 1073-1083, IEEE, New York, US; G.A. CLARK et al.: "A unified approach to time- and frequency-domain realization of FIR adaptive digital filters"<br>* Whole document *<br>--- | 1 | H 03 H 21/00 |
| A | PROCEEDINGS OF THE IEEE, vol. 72, no. 12, December 1984, pages 1816-1818, IEEE, New York, US; J.C. LEE et al.: "A reduced structure of the frequency-domain block LMS adaptive digital filter"<br>* Whole document *<br>----- | 1 | TECHNICAL FIELDS SEARCHED (Int. Cl.4)<br>H 03 M |

The present search report has been drawn up for all claims

| Place of search | Date of completion of the search | Examiner |
|---|---|---|
| THE HAGUE | 30-09-1987 | COPPIETERS C. |

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

A6

# United States Patent [19]

Auvray

[11] Patent Number: 4,992,967

[45] Date of Patent: Feb. 12, 1991

[54] DIGITAL FILTER USING FOURIER TRANSFORMATION

[75] Inventor: Gérard Auvray, Longpont Sur Orge, France

[73] Assignee: Thomson-CSF, Paris, France

[21] Appl. No.: 258,814

[22] Filed: Oct. 17, 1988

[30] Foreign Application Priority Data

Oct. 16, 1987 [FR] France .............................. 87 14291

[51] Int. Cl.⁵ .............................................. G06F 15/31

[52] U.S. Cl. .............................................. 364/724.18

[58] Field of Search ....................... 364/724.18, 724.16

[56] References Cited

U.S. PATENT DOCUMENTS

| | | | |
|---|---|---|---|
| 4,093,994 | 6/1978 | Nussbaumer | 364/724.18 |
| 4,100,604 | 7/1978 | Perreault | 364/724.18 |
| 4,530,076 | 7/1985 | Dwyer | 364/724.18 |
| 4,807,173 | 2/1989 | Sommen et al. | 364/724.18 |

FOREIGN PATENT DOCUMENTS

3610195 10/1987 Fed. Rep. of Germany .

OTHER PUBLICATIONS

Schaller Nachrichtentechnische Zeitschrift "Verwendung der Schnellen Fourier Transformation in Digital en Filtern" vol. 27, No. 11, pp. 425–431 Nov. 1974.

IEEE Transactions on Acoustics, Speech, and Signal Processing, No. 2, pp. 109–114 Apr. 1976, "Digital Filtering by Polyphase Network".

IEEE ASSP Magazine, vol. 3, No. 3, pp. 6–11 Jul. 1986, Bellanger, "New Applications of Digital Signal Processing in Communications".

*Primary Examiner*—David H. Malzahn

*Attorney, Agent, or Firm*—Roland Plottec

[57] ABSTRACT

A digital filter having parallel paths with a plurality of transverse filters in the paths. The period of treating the digital information of N connections at the output of a circuit for calculating the discrete Fourier transform on 2N points. The output of the transverse filter is fed to a circuit for calculating the inverse Fourier transform. These two circuits permit the transverse filters to function in each path with a data period of N. The filter will find particular application in convolvers, courolaters, and for circuits for the compression of digital pulses. The filter lends itself to application, particularly in radar, sonar, telecommunications, and in sound and imaging systems.

7 Claims, 7 Drawing Sheets

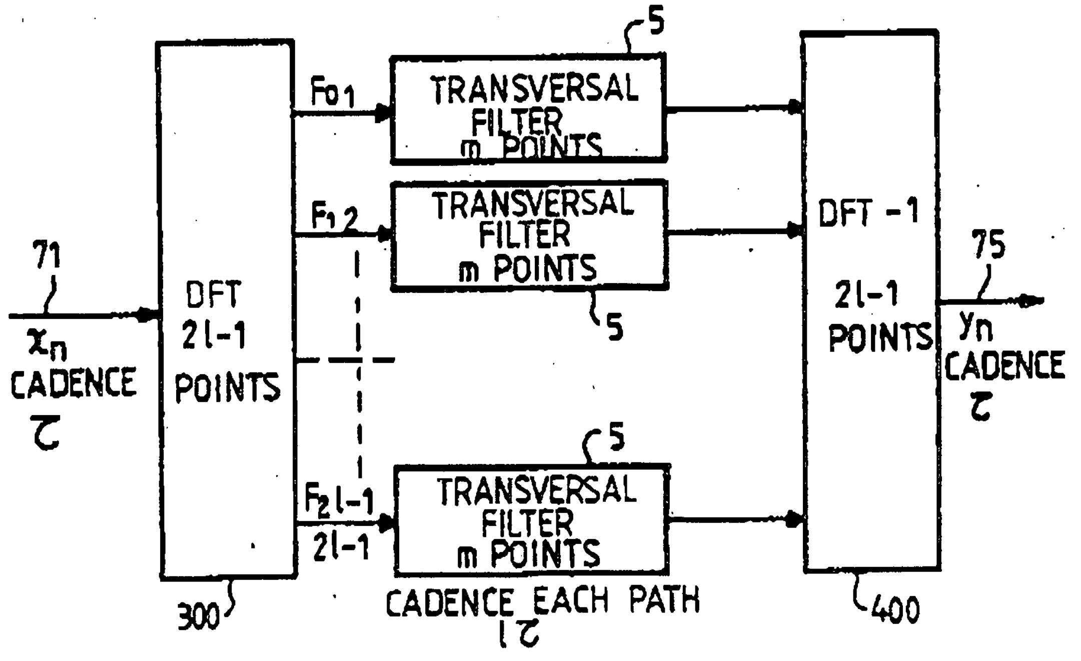


FIG.1

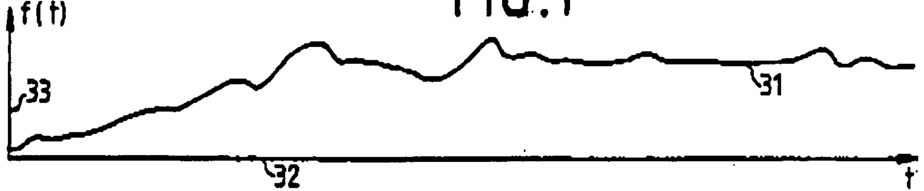


FIG.2

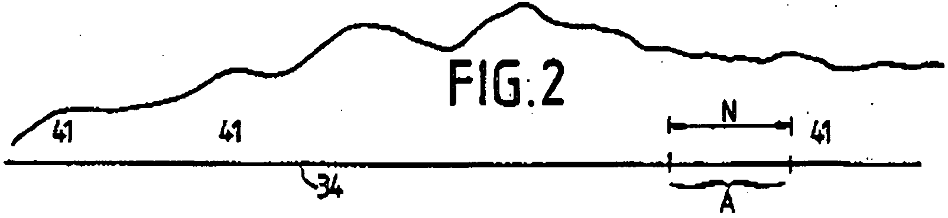


FIG.3

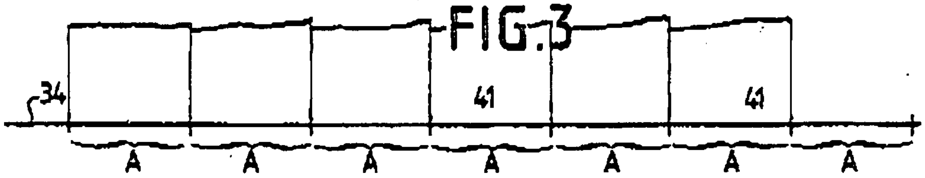


FIG.4a

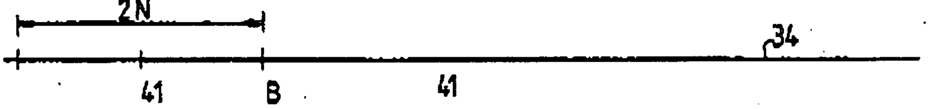


FIG.4b

FIG.4c

FIG.4d

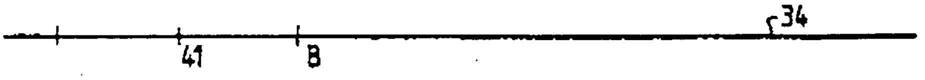


FIG.5a

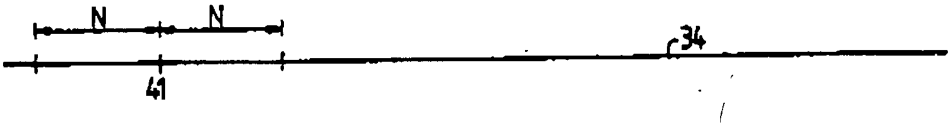


FIG.5b

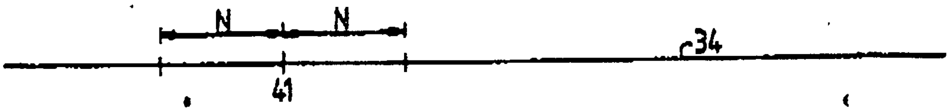


FIG.5c

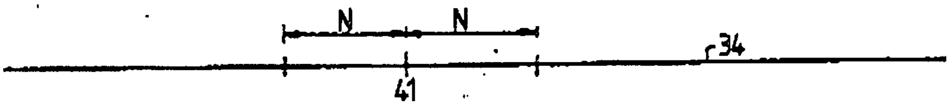


FIG.5d

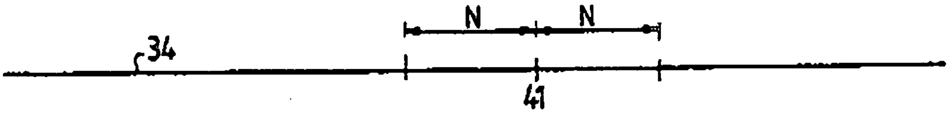


FIG.6a

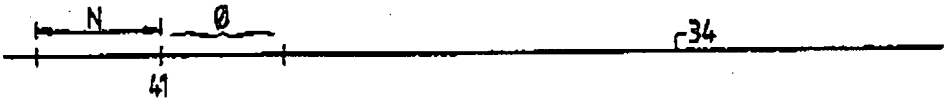


FIG.6b

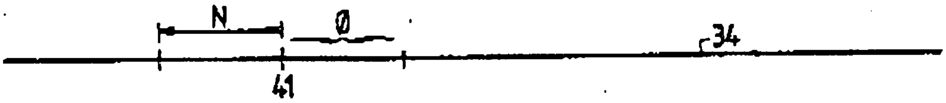


FIG.6c

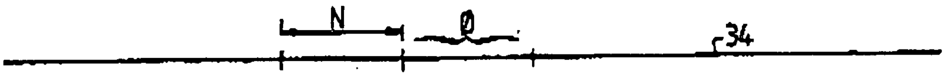


FIG.6d

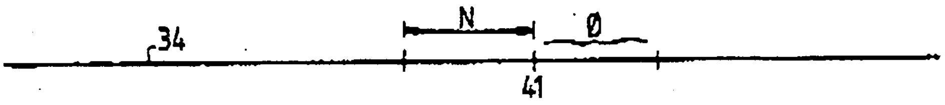


FIG. 7

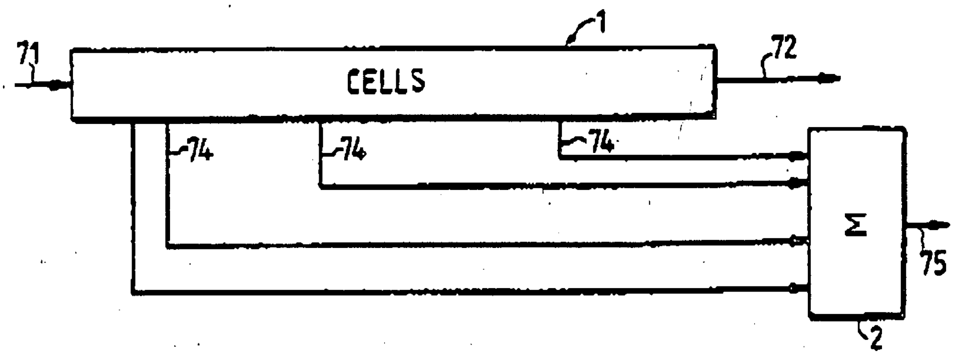

71
1
CELLS
72
74
74
74
Σ
75
2

FIG. 8

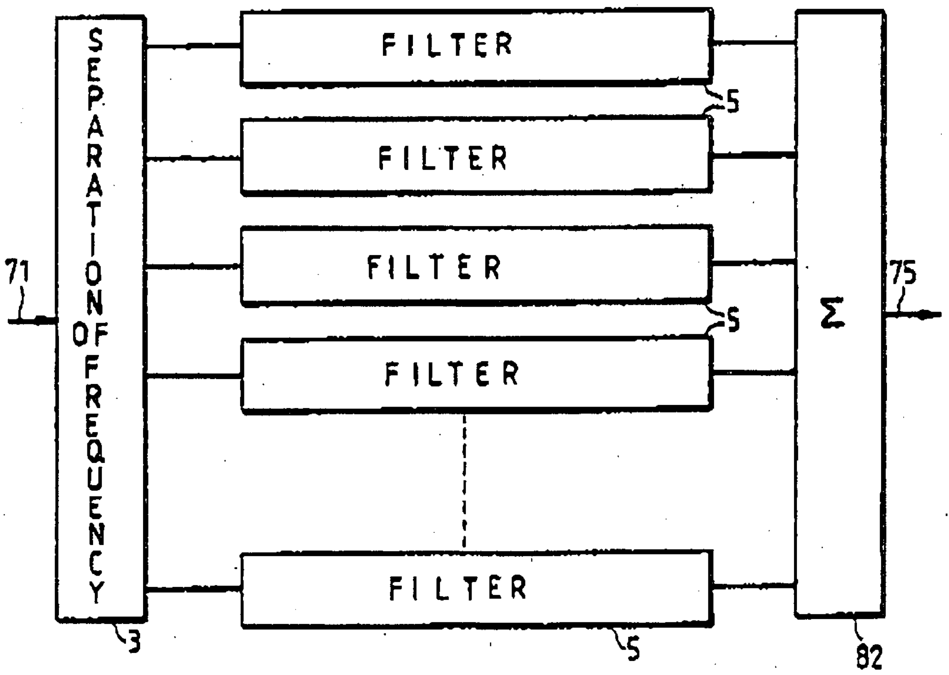
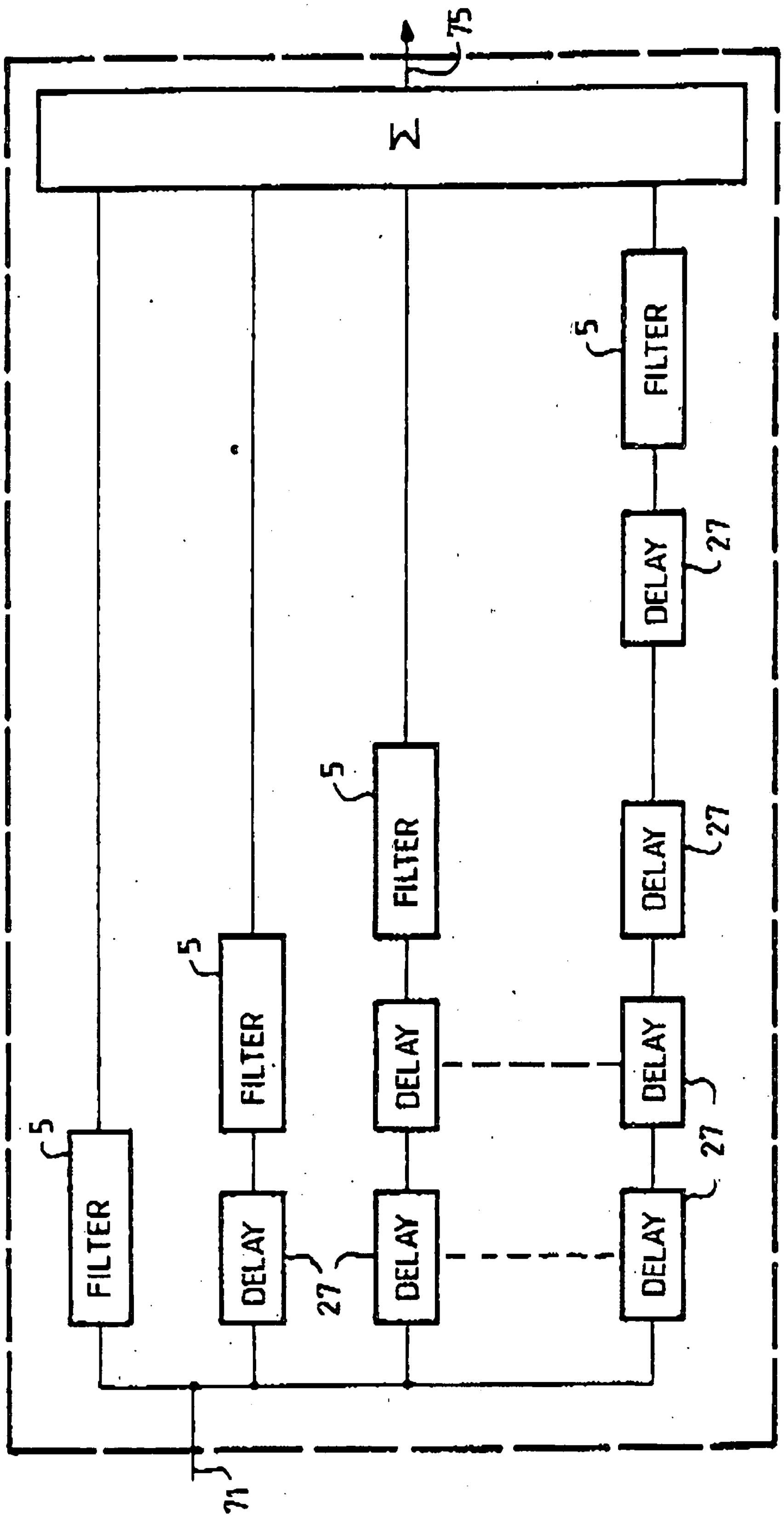

SEPARATION OF FREQUENCY
71
FILTER
FILTER
FILTER
FILTER
FILTER
5
5
5
3
Σ
75
82

FIG_9


75
Σ
FILTER
5
DELAY
27
FILTER
5
DELAY
27
FILTER
5
DELAY
27
DELAY
27
FILTER
5
DELAY
27
DELAY
27
DELAY
27
DELAY
27
71

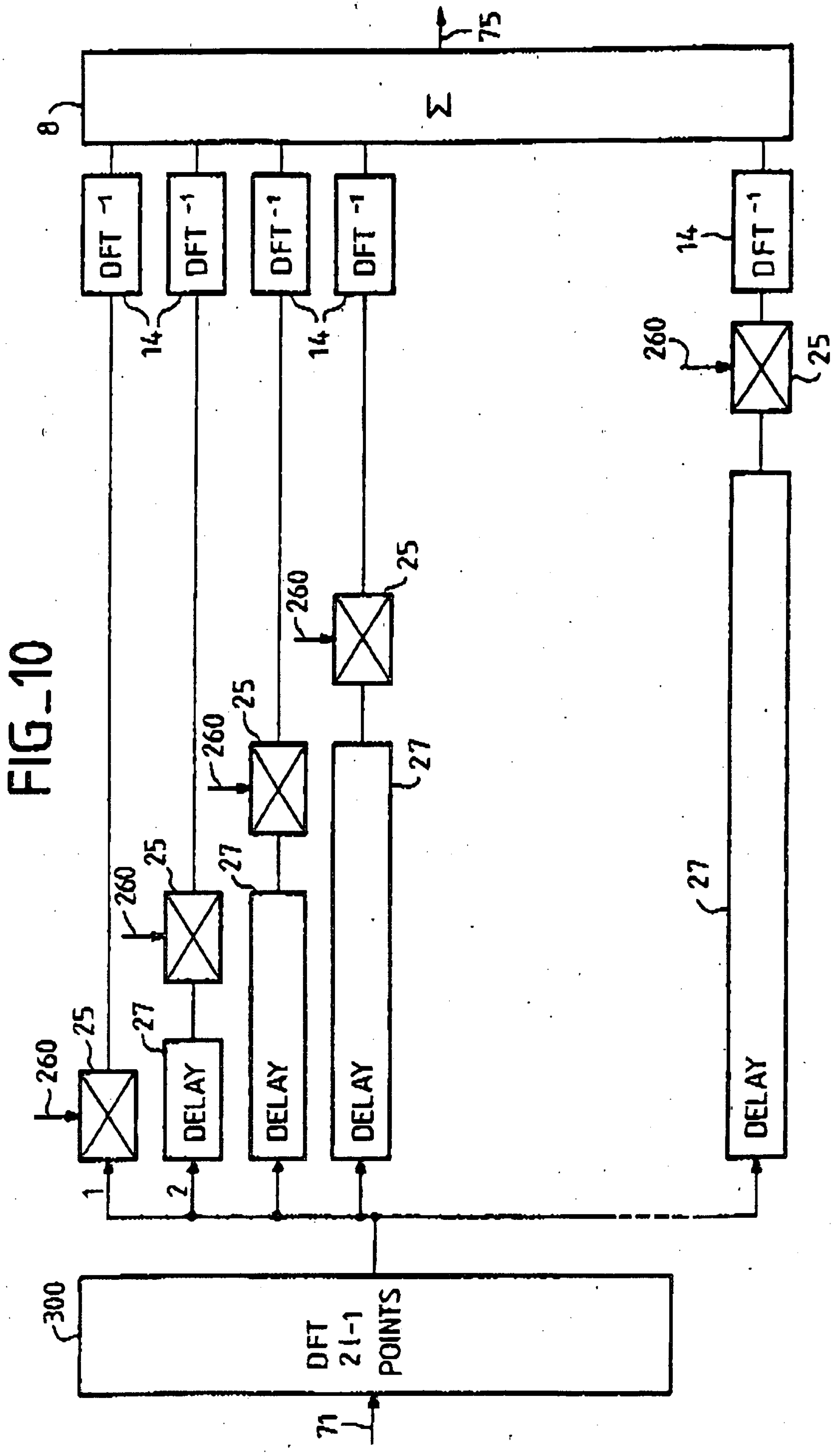

75
Σ
8
DFT⁻¹
DFT⁻¹
DFT⁻¹
DFT⁻¹
DFT⁻¹
14
14
14
260
25
260
25
260
25
260
25
260
25
27
27
27
27
DELAY
DELAY
DELAY
DELAY
1
2
300
DFT
2l-1
POINTS
71

FIG_10

FIG_11

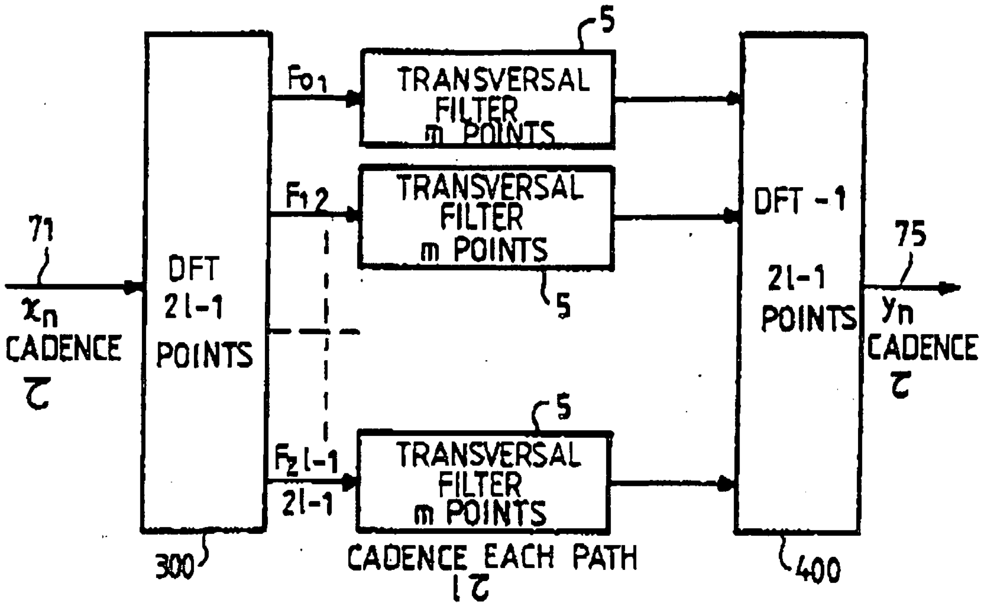

5
Fo1
TRANSVERSAL FILTER m POINTS
F1 2
TRANSVERSAL FILTER m POINTS
5
71
$x_n$
CADENCE
τ
DFT 2l-1 POINTS
5
F2l-1 2l-1
TRANSVERSAL FILTER m POINTS
CADENCE EACH PATH
lτ
300
DFT -1 2l-1 POINTS
75
$y_n$
CADENCE
τ
400

FIG_12

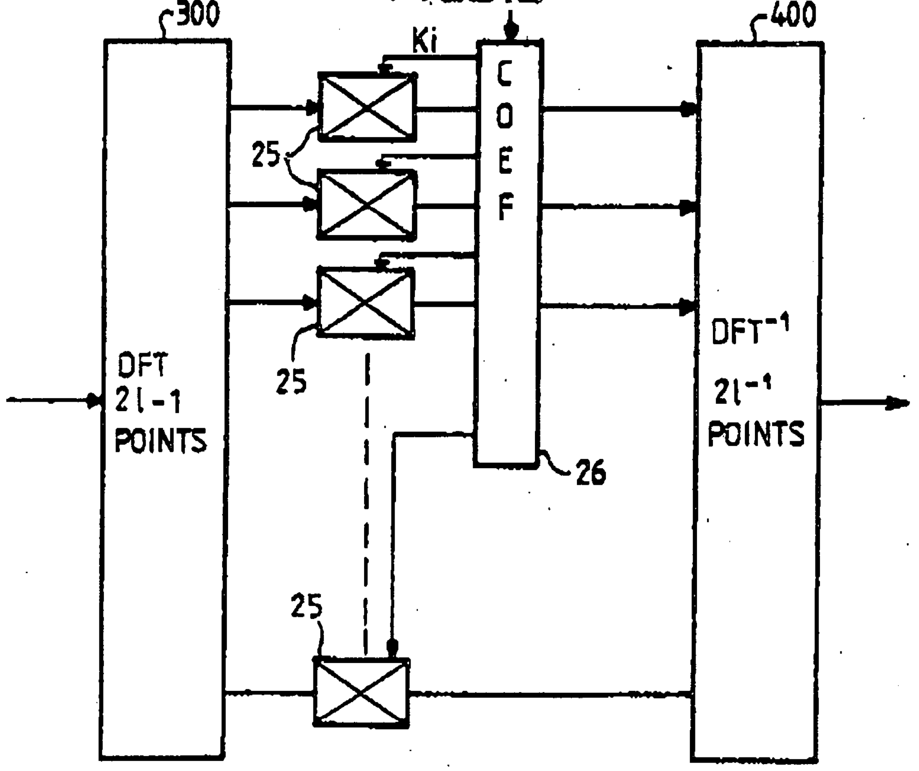
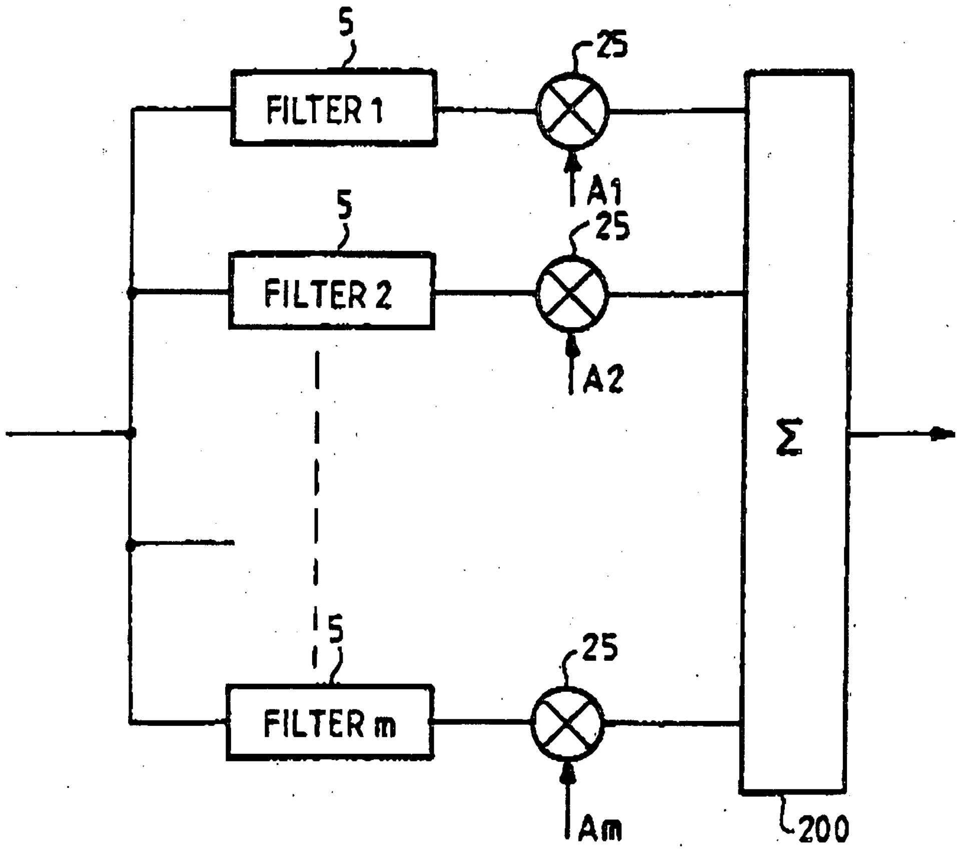

300
Ki
C O E F
25
25
25
26
DFT 2l-1 POINTS
400
DFT$^{-1}$ 2l$^{-1}$ POINTS

FIG_13



## DIGITAL FILTER USING FOURIER TRANSFORMATION

### BACKGROUND OF THE INVENTION

The invention relates generally to digital filters, and to radar having digital filters.

The invention also relates to a digital filter having a plurality of paths and the parallel treatment of data in these paths.

It is known in the prior art to filter analog data using a plurality of parallel paths. Particularly, a plurality of parallel treatment paths are used to achieve, in analog form, the compression of radar pulses, when N is the number of paths used. The rate of compression of each path will be equal to u/N$^2$ where u is the rate of compression.

In digital forms, dividing the treatment of the data between N parallel paths with a period of treatment N times slower would be an advantage. The rate of compression affected in each path would be equal to BT/N$^2$, where B is the pass band of the signal, and T the width of a pulse. The rate of compression of the circuit, having N paths in parallel, is equal to B×T.

Previously, it was believed that for N parallel paths sliding was necessary for the digital filtering for each path of the data being filtered in each path and then to sum the results. This leads to a required calculating level proportional to N$^2$. An advantage of the filter of the present invention is a reduction in the needed calculating power brought about by the separation of the paths by a discrete Fourier transform on 2N points. This gives the minimal number of points, specifically 2N−1, where the convolution of N points with the impulse response on N points of a filter can be correctly calculated by DFT (DFT)$^{-1}$ for the N points of the interval.

An object of the invention is to provide a digital filter having means for calculating the discrete Fourier transform, which is connected in parallel by a plurality of paths along which data is treated. The paths of treatment are connected to means for calculating the inverse discrete Fourier transform. The period of treatment along the parallel paths is N times less than the total input/output period of the device, N being equal to the number of samples treated simultaneously by the filter.

The invention will be better understood by a reference to the following description of different embodiments of the invention and the attached figures.

### BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a curve of a function, which is a function of time.

FIG. 2 is a representation of the curve of FIG. 1 being sampled.

FIG. 3 is a diagram of a part of the samplings of FIG. 2 which are periodic.

FIGS. 4*a*–4*d* are diagrams illustrating the calculation of a shifting Fourier transform.

FIGS. 5*a*–5*d* are diagrams showing a first example of operation of a device for calculating the Fourier transform according to the present invention.

FIGS. 6*a*–6*d* are diagrams of a second example of operation of a circuit for calculating the Fourier transform according to the present invention.

FIG. 7 is a diagram of a transverse filter adopted to be used in the circuit of the present invention.

FIG. 8 is a diagram of a digital filter having parallel paths.

FIG. 9 is a schematic drawing of an embodiment of a circuit of the present invention.

FIG. 10 is a schematic drawing of an embodiment of a circuit according to the present invention.

FIG. 11 is a schematic drawing of an embodiment of a circuit according to the present invention.

FIG. 12 is a schematic drawing of an embodiment of the present invention.

FIG. 13 is a schematic drawing of an embodiment of the circuit of the present invention.

### DETAILED DESCRIPTION

In FIG. 1, there is shown a curve 31 of a function f(t), 33 which is a function of time 32. Curve 31 represents, for example, the modulation of an electrical signal. In order to numerically/digitally manipulate this signal, a sampling is made of the signal as shown in FIG. 2.

In FIG. 2, there is shown on a coordinate 34, which corresponds to time 32 of FIG. 1, a value 41 of the function f(t) at time t. If the frequency of sampling the signal tends toward infinity, with the time between two successive samplings approaching zero, then the digitized sampled signal would have all the information contained in the original analog signal. In practice, a sampling frequency is chosen, which conforms to Shannon's Theorem, and it is understood that all increases of the sampling frequency requires the use of superior performing components.

Let us define A an interval of time with N samplings 41. The calculation of the fast Fourier transform of N points in the interval A does not correspond to the calculation of the digitized function of FIG. 2, but to a function rendered periodic as shown in FIG. 3.

In FIG. 3, there is shown a periodic function having a succession of intervals A of N samplings 41. In parallel path filters, for example the one shown in FIG. 8, a shifting Fourier transform is made, as shown in FIGS. 4*a*–4*d*, corresponding to a Fourier transform of N$^2$ points.

A cycle of calculation separates FIGS. 4*a*, 4*b*, 4*c* and 4*d*. In each cycle the Fourier transform of 2N points 41 is calculated by shifting at each cycle the interval of calculation B of a sampling.

In FIGS. 5*a*–5*d* there is illustrated a first example of a calculation of the Fourier transform made by the filter of the present invention. One cycle of calculation separates the FIGS. 5*a*–5*d*. Only the four first cycles have been shown. It should be understood that the process is continued until the last calculation has been made.

FIG. 5*a* illustrates the calculations of the fast Fourier transform on 2N points 41. FIG. 5*b* illustrates the following cycle of calculation, in which a calculation is made on 2N points 41 shifted by N in relation to the points of FIG. 5*a*. Likewise FIG. 5*c*, which illustrates a cycle of calculations after that of FIG. 5*b*, shows the calculation of the fast Fourier transform on 2N points 41 shifted by N in relation to the beginning of the interval of FIG. 5*b*. During each cycle of calculation, a calculation is made of the fast transform on an interval having 2N points 41, the interval being shifted by N points in relation to the preceding cycle of calculations.

In FIGS. 6*a*–6*d* there is illustrated a second example of calculations made in accordance with the filter of present invention. FIGS. 6*a*–6*d* are each shifted by one cycle of calculation. At the time corresponding to FIG. 6*a* a calculation of the fast Fourier transform is made on an interval having N points 41 corresponding to samplings of the signal to be treated followed by N$\phi$.

In FIG. 6b calculations are made on the N following points 41 followed by N$\phi$. In each cycle the fast Fourier transform is calculated on an interval having N points followed by N$\phi$. Between two cycles the interval is shifted by N points.

FIG. 7 shows an example of a filter according to the invention. It has a transversal filter, which is known per se. This filter has several stages which are connected to each other. The signal travels between an input 71 and an output 72 through a plurality of stages 1. The filter has several transversal outputs 74. The outputs 74 are connected to the stages of the filter, and in these filter stages there was a multiplication of the signal by a weighting coefficient. The outputs 74 are connected to an input of a summer 2. The result of these calculations is provided at an output 75 of the summer 2.

The output 72 of the several stages 1 may be connected to an input of a next stage transversal filter (not shown in FIG. 7) and which would be connected in series. The overall transversal filter thus provides a signal summing the two outputs 75.

In the case of the compression of pulses corresponding to the particular values of the multiplication coefficients of the outputs 74, the signal to be used is provided at the output 75.

In FIG. 8 there is shown a filter of a known type, which is the same as the transversal filter of FIG. 7. The filter of FIG. 8 has a frequency separation filter 3, which may be considered as a digital shifting filter, furnishing output signals on N parallel channels. The filter 3 feeds frequency the sub-bands to filters 5, which are in parallel. The calculation in FIG. 8 is brought about by a shifting calculation, that is to say, in each cycle of calculations, a new sample is fed into the filters 5. This results in the overall calculation of the circuit of FIG. 5 as proportional to N$^2$, N being the number of filters 5. The output of the frequency separation filter 3 is connected in parallel to a plurality of filters 5. These filters 5 redivide the band of the signals to be treated. The filters 5 are digital filters of a known type, which may be of any convenient or conventional type. The outputs of filters 5 are connected to a summer 4. At the output of the summer 4, there is provided a filtered signal. The circuit of FIG. 8 requires a numerical filter having a very large calculating capacity.

In FIG. 9 there is shown a schematic equivalent of the transversal filter of FIG. 7. In FIG. 9 there is shown a filter having parallel channels. Each channel has a filter corresponding to a section 5 of cells of a transversal filter, and having between 0 and l—1 elements leading to a delay increment of l$\tau$: where l is the number of channels.

In FIG. 10 there is shown a circuit equivalent to that of FIG. 9 where the response of each filter corresponds to each of the channels calculated by DFT (DFT)$^{-1}$ having 2l—1points. Each of the channels 1, 2, . . . l has 2l—1 paths corresponding to 2l—1 coefficient DFT of the input signal 71.

The circuit of FIG. 10 provides a transversal filtering function of N points. The calculations are brought about in each of the m parallel paths, each path treating l point. Thus, N=m·l.

In the following equations, the superscripts represent the number of a path (1 to m), the subscripts represent the number of sample.

Advantageously, the coefficients 260 are:

$$A^1{}_l, A^m{}_l \text{ with } l=1 \text{ to } 2l-1 \qquad \text{Equation - 1}$$

the coefficients of the DFT of the pulse response of the transversal filters 1 to m of FIG. 9.

The numerical signal 720, provided at the output of the filter, are:

$$z_{nr} = \sum_{p=1}^{m} y^p_{nr} \qquad \text{Equation-2}$$

r being the number of points in a unit of the TFD

$$y^p_{nr} = \sum_{l=0}^{l=2l-2} F_{l(r-(p-1))}A^p_l \exp(j2\pi nl/2l-1) \qquad \text{Equation-3}$$

where y is the signal presented at the output of the filters 25;

F being the DFT of the function f to be filtered. Thus:

$$z_{nr} = \sum_{l=0}^{l=2-2} \exp(j2\pi nl/2l-1)\left[\sum_{p=1}^{p=m} F_{l(r-(p-1))}A^p_l\right] \qquad \text{Equation-4}$$

It is (DFT)−1 of a group of lines DFT F'i corresponding to the terms between the brackets.

$$F_l = \sum_{p=1}^{p=m} F_{l(r-(p-1))}A^p_l \qquad \text{Equation-5}$$

F' is the linear convolution of $F_l$ of the successive blocks with the pulse response of a filter having $A^p{}_l$ with p=1 to m.

This is implemented by the filter of the invention shown in FIG. 11. The filter of FIG. 11 has a DFT 300 for calculating the DFT, for example, on 2l−1 points connected in parallel to a group of paths having transversal filters 5. The outputs of these transversal filters 5 are connected to the inputs of a inverse DFT calculator 400.

The signal at an input 71 to DFT calculator 300 has a period of $\tau$.

The transversal filters function with a period of l$\tau$.

The signal at an output 75 is provided with also having a cadence of $\tau$. It is possible to replace the number of points of the DFT, (DFT)−1 here equal to 2l −1 by an amount 2l which is a power of 2 in such a manner to able to use a calculation device for fast Fourier transforms.

The period of calculation of the circuit corresponds to the band pass of the digitized signal which is able to be treated and in B=1/$\tau$ l$\tau$ being the delay increment of the delay line 27, i.e. the difference of delay introduced between two parallel adjacent paths.

The period at which the transversal filters operate is b where b is small in comparison B.

It is necessary that:

$$\frac{1}{l\tau} = b \qquad \text{Equation-6}$$

The necessary calculating level is thus:

$$m(2l-1)b = 2Nb \qquad \text{Equation-7}$$

The calculating level is thus proportional to N in the case of a filter according to the present invention.

In contrast, for filters of the prior art, the necessary level for calculation was equal to N.B.=Nαb with α=B/b·α is the coefficient of growth of the band.

In the case of prior art filters the power of calculation required is proportional to αN and not to N. It is thought desirable to exclude the two extreme known cases from the present invention:

The case where l=1, i.e., m=N corresponds to a transversal filter of a known type of a single path;

The case where m=1, i.e., l=N corresponds to a classical filter by DFT and DFT inverse.

Thus, in the case of the present invention, $l \in ]1, N[ \cap Z$, that is to say, l pertains to the interval [1, N] with 1 and N excluded, this interval being taken within the set of integers Z and N being the number of points treated by the filter according to the present invention.

In FIG. 12 there is shown an example of a filter according to the present invention which does not have transversal filters but multipliers in each path. These perform the function of separating into small bands, the signal at the input and provides a summing of these bands as shown in FIG. 13.

The circuit of FIG. 12 has a DFT 300 for calculating the DFT on at least 2l−1 points and is connected to a group of parallel paths. Each parallel path has a multiplier 25. The outputs from the multipliers are connected to a calculating circuit of the DFT inverse 400 on at least 2l−1 points.

The multipliers 25 receive the coefficients $K_i$ i=1 to m, for example from an output of a memory 26. The memory 26 in the first example of one embodiment of the invention receives the coefficient in series and sends them out in parallel to the multipliers 25.

In an alternative embodiment, the coefficients are stored in memories and are addressed by a sequencer not shown to furnish the necessary coefficients to the function of the circuit in accordance of the present invention.

Advantageously:

$$K_l = F_{lr}\left[\sum_{p=1}^{p=l} \alpha_p A_l^p\right] \quad \text{Equation-8}$$

where:

r is the number of blocks of points of DFT;

F is the DFT of the function to be treated;

αp are the coefficients of the linear combination.

$A_l^p$ are the coefficients of the Fourier transform of the impulse responses from the filters of the parallel paths of FIG. 13.

The results of the calculation are thus:

$$z_{nr} = \sum_{i=0}^{i=2l-1} \exp(j2\pi ni/N)\, K_l \quad \text{Equation-9}$$

The period of calculation to the input and to the output of the circuit of FIG. 12 is equal to τ while in the interior of the circuit according to the invention the period is equal to lτ.

In order to use the FFT one can take 2l power of 2 in place of 2l−1.

When FFT is used and the filters dividing the band correspond to the l-point FFT filters, the Fourier transform of the impulse responses of each filter only includes one line corresponding to the central frequency of the filters.

The operation of the linear combination after separation into paths may thus be effected by FFT Linear combination of the lines (FFT)−1.

There is thus only one line per filter.

$$A_l^p = 0 \text{ if } l \neq p - 1 A^b_{p-1} = \alpha. \quad \text{Equation-10}$$

Therefore, only that lines of the FFT 2l of signals common with the FFT l are used.

In this particular case; the FFT; $(FFT)^{-1}$ are thus of a rank equal to the number of paths.

The circuit according to the present invention lends itself to be used particularly in radar, sonar, telecommunications, and for circuits and systems for treating sound and images.

I claim:

1. A digital filter using Fourier transformation for filtering sampled input data comprising samples at a predetermined rate arranged in successive blocks of l samples, said filter comprising:

first means for calculating a discrete Fourier transform on sets of 2l−1 points, each set of 2l−1 samples being shifted by l samples with respect to the preceding one;

second means for processing data including l parallel filtering paths arranged in a sequence and connected to said first means, wherein said 2l−1 points of the discrete Fourier transform are applied to each of said paths and wherein said parallel paths comprise respective delay means whereby said delay means are selected for introducing a constant relative delay between each two successive paths in said sequence, said relative delay being equal to l times a sampling interval of said input data; and

third means connected to said parallel paths for combining signals processed in said paths, said third means having an output delivering filtered data at said predetermined rate, in response to said input data.

2. A digital filter according to claim 1, wherein each of said paths further comprises a transversal filter and wherein said third means consist in summing means for summing said signals processed in said paths.

3. A digital filter according to claim 1, wherein each of said paths further comprises means for effecting a linear convolution of said 2l−1 points of the discrete Fourier transform with discrete Fourier transforms of impulse responses of l transversal filter sections and wherein said third means comprise l $DFT^{-1}$ means for calculating an inverse discrete Fourier transform respectively connected to said paths and summing means connected to said l $DFT^{-1}$ means.

4. A digital filter using Fourier transformation for filtering sampled input data comprising samples at a predetermined rate arranged in successive blocks of l samples, said filter comprising:

first means for calculating a discrete Fourier transform on sets of 2l−1 points, each set of 2l−1 samples being shifted by l samples with respect to the preceding one;

second means for processing data including 2l−1 parallel filtering paths arranged in a sequence and connected to said first means, wherein said 2l−1 points of the discrete Fourier transform are applied to said paths respectively; and third means connected to said parallel paths for combining signals processed in said paths, said third means having an output delivering filtered data at said predetermined rate, in response to said input data, said third means comprising a DFT$^{-1}$ means for calculating a 2l−1 points inverse discrete Fourier transform on said signals processed in said paths.

5. A digital filter according to claim 4, wherein each of said paths comprises a m-point transversal filter, where m is a predetermined integer, said transversal filters operating with a period equal to l times a sampling interval of said input data.

6. A digital filter according to claim 4, wherein each of said paths comprises a multiplier having a first input and an output, forming an input and an output of said path, and a second input and wherein said second means further comprise memory means for storing 2l−1 coefficients $K_i$, where $1<i<2l-1$, and for providing said coefficient $K_i$ to said second input of the multipliers of said paths respectively, said coefficients $K_i$ being selected according to the equation:

$$K_i = F_{ir}\left[\sum_{p=1}^{p=l} \alpha_p A^p_i\right]$$

where $F_{ir}$ represents the i$^{th}$ point of the discrete Fourier transform calculated by said first means on the r$^{th}$ block of samples, $A^p_i$ represents the i$^{th}$ term of the discrete Fourier transform of impulse response of a l-point transversal filter and $\alpha_p$ are coefficients of a weighted linear combination constituting said filtered data delivered by said output of said third means.

7. A digital filter according to anyone of claims 1 to 6, wherein each of said set comprises a block of samples followed by a block of zeros.

* * * * *

A7

# Multidelay Block Frequency Domain Adaptive Filter

JIA-SIEN SOO AND KHEE K. PANG

*Abstract*—A flexible multidelay block frequency domain (MDF) adaptive filter is presented. The distinct feature of the MDF adaptive filter is to allow one to choose the size of an FFT tailored to the efficient use of a hardware, rather than the requirements of a specific application. The MDF adaptive filter also requires less memory and so reduces the requirement and cost of a hardware. In performance, the MDF adaptive filter introduces smaller block delay and is faster, ideal for a time-varying system such as modelling an acoustic path in a teleconference room. This is achieved by using smaller block size, updating the weight vectors more often, and reducing the total execution time of the adaptive process. The MDF adaptive filter compares favorably to other frequency domain adaptive filters when its adaptation speed and misadjustment are tested in computer simulations.

## I. Introduction

Adaptive digital filters have become increasingly popular due to their "intelligent" nature of processing signals and the emergence of a family of powerful digital signal processors. There are many adaptive algorithms available; each has its own merits and special applications. However, for applications such as an acoustic echo canceller in teleconference systems [1], which requires filter lengths of several hundreds and sometimes thousands, the frequency domain block adaptive filter based on the least mean-square algorithm (FLMS) [2] is considered to be most suitable. This is because the FLMS adaptive filter implements the block LMS (BLMS) [3] algorithm efficiently by using the fast Fourier transform (FFT). In so doing, a significant reduction in computational load for the same adaptation performance is achieved. Many other attractive features and variations of the FLMS adaptive filter can be found in [2]–[7]. However, a few practical implementation problems of the FLMS adaptive filter have hindered its applications. These are as follows.

*1) Inefficient Use of a Hardware:* For an adaptive filter, a $2N$-point FFT is generally used for an $N$-point weight factor. Most of the available FFT or DSP chips are designed and optimized for small size FFT, typically 256 point. To implement an acoustic echo canceller of a few thousands taps, several FFT chips are cascaded together with external memory to form a larger FFT configuration, which is rather inefficient and expensive.

*2) Long Block Delay:* Since the FLMS algorithm implements block processing, if the weight size $N = 1024$, the first output $y_{k-1}$ needs to wait after the last output $y_{k-1024}$ of the same block is processed or a delay of 128 ms for an 8 kHz sampling rate. Such a long delay would make the echo more annoying.

*3) Large Quantization Error in FFT:* As the size of an FFT becomes larger, the number of multiplications and scalings increases. This causes extra quantization error.

With these limitations in mind, we present a more flexible frequency domain adaptive filter structure, called the *multidelay block frequency domain* (MDF) adaptive filter in this correspondence. The performance of the MDF adaptive filter is compared to the existing frequency domain adaptive filters. It is found that by using a small FFT size and updating the weights more often, the MDF adaptive filter has a shorter block delay, faster adaptation speed, and small memory requirement.

Before we proceed to the next section, we denote the upper and lower case symbols as frequency and time domain variables, respectively. The boldface symbols will represent vectors or matrices. All vectors are specified as column vectors with superscript $T$ to denote its transpose operation. An asterisk will denote complex conjugate transposition.

## II. The MDF Adaptive Filters

To compute the linear convolution/correlation in the FLMS [2], [5] adaptive filter, either the overlap-save or overlap-add technique [4] is normally used. It is shown in [8] that by splitting the overlap-save method into two smaller blocks of an overlap-save process, the performance of the adaptive filter improves significantly. In this section, we extend the idea further to an arbitrary number of smaller delay blocks and generalize it to the MDF adaptive filter.

Let $N$ be the total number of weights to be modeled and let $M$ be the number of delay blocks. We then choose $N'$ to be the size of the FFT, with $N'$ equal to the smallest power of two integers larger than or equal to $2N/M$. The first step of the MDF algorithm is to convert the most recent overlapped input samples to the frequency domain via the FFT as

$$\begin{aligned} X(M, j) = \operatorname{diag}\Big\{\mathrm{FFT}\big[&x_0(j-1), x_1(j-1), \\ &\cdots, x_{N'/2-1}(j-1), \\ &x_0(j), x_1(j), \cdots, x_{N'/2-1}(j)\big]^T\Big\} \end{aligned} \tag{1}$$

where $j$ is the block iteration index. The earlier delay block input vectors are obtained via block index shifting without invoking any computation as follows:

$$X(m, j) = X(m+1, j-1), \quad m = 1, 2, \cdots, M-1. \tag{2}$$

This suggests that only one FFT is needed per block iteration to transform the input vector, a significant computation saving. The output and error vectors can be expressed as

$$y(j) = \text{last } N'/2 \text{ terms of } \left\{\mathrm{FFT}^{-1}\left[\sum_{m=1}^{M} X(m, j) W(m, j)\right]\right\} \tag{3}$$

$$E(j) = \mathrm{FFT}\Big\{\big[\underbrace{0, 0, \cdots, 0}_{N'/2 \text{ zeros}}, \underbrace{[d(j) - y(j)]^T}_{N'/2 \text{ terms}}\big]^T\Big\} \tag{4}$$

where $W(m, j)$ is the $m$th weight vector and $d(j)$ is the desired vector. The weight update equations based on the LMS criteria to minimize the $|e(j)|^2$ are given as

$$\phi(m, j) = \text{first half of } \left\{\mathrm{FFT}^{-1}\left[X^*(m, j) E(j)\right]\right\} \tag{5}$$

$$\Phi(m, j) = \mathrm{FFT}\big[\phi(m, j), \underbrace{0, 0, \cdots, 0}_{N'/2 \text{ zeros}}\big]^T \tag{6}$$

$$W(m, j+1) = W(m, j) + M\mu_B \Phi(m, j) \tag{7}$$

where $m = 1, 2, \cdots, M$ and $\mu_B$ is the block step size.

The block diagram of the MDF adaptive filter depicted in Fig. 1 clearly illustrates the cascadable and efficient block structure. Note that the input/output operations of the MDF adaptive filter are identical to the FLMS, except now the FFT is $N'$-points long. The FLMS adaptive filter can, in fact, be regarded as the special case of MDF with $M = 1$. If the self-orthogonalizing algorithm is

Manuscript received February 9, 1988; revised May 19, 1989. This work was supported by the CSIRO (Australia) Collaborative Program in Information Technology.

The authors are with the Department of Electrical and Computer Systems Engineering, Monash University, Clayton, Victoria 3168, Australia.

IEEE Log Number 8932759.

0096-3518/90/0200-0373$01.00 © 1990 IEEE

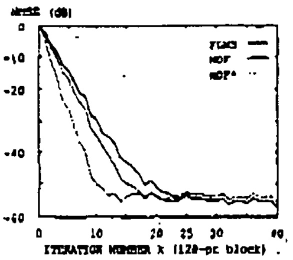


Fig. 3. Convergence characteristics of MDF ($M = 16$) and FLMS adaptive filters for uncorrelated input and step sizes: $\mu_g(\mathrm{MDF}) = 0.625E - 7$, $\mu_g(\mathrm{MDF}^*) = 0.12E - 6$, and $\mu_g(\mathrm{FLMS}) = 0.1E - 5$.

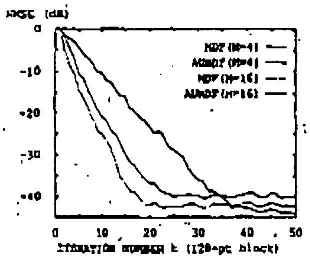


Fig. 4. Convergence characteristics of MDF and AUMDF adaptive filters. Same conditions as in Fig. 2.

TABLE I
Execution Time for the Adaptive Filters to Compute FFT

| | Execution Time (in ms) | | | | | | |
|---|---|---|---|---|---|---|---|
| | Complex FFT | | Various Adaptive Filters | | | | |
| DSP | 256-Point | 1024-Point | FLMS | UFLMS | MDF | AUMDF | UMDF |
| DSP56000 | 0.71 | 4.99 | 25.0 | 15.0 | 14.2 | 11.4 | 8.5 |
| TMS320C25 | 1.22 | 7.10 | 35.5 | 21.3 | 24.4 | 19.5 | 14.6 |

## IV. Computational Complexity

Refer to Fig. 1: it is clearly shown that the number of multiplications of the frequency domain adaptive filter is directly proportional to the number and size of the FFT used. A $2N$-point FFT can be shown to require $N/2 \log_2 N$ complex multiplications or $2N \log_2 N$ real multiplications [9]. With this assumption, the total numbers of multiplications per output sample for different adaptive filters with fixed $\mu_g$ are

$$\text{FLMS:}\quad 10 \log_2 N + 8 \tag{16}$$

$$\text{MDF:}\quad (4M + 6) \log_2 N + 8M - (4M + 6) \log_2 M \tag{17}$$

$$\text{AUMDF:}\quad 10 \log_2 N + 8M - 10 \log_2 M \tag{18}$$

$$\text{UMDF:}\quad 6 \log_2 N + 8M - 6 \log_2 M. \tag{19}$$

The memory requirement of the proposed frequency domain adaptive filter is much less than the FLMS. For the MDF adaptive filter, the memory storage is approximately $4N + 10N/M$ points compared to roughly about $14N$ points for the FLMS algorithm. Thus, if $M \geq 4$, there is more than 50 percent reduction in memory storage for the MDF adaptive filter. The above comparison is based on the number of multiplications and memory storage required separately. To make a more practical comparison, we combine both of these two factors in terms of the total execution time needed to perform the FFT operations. We first assume $N = 512$ and the DSP used is either the Motorola DSP56000 or TI TMS320C25. Since both of these DSP's are optimized for a 256-point FFT, we select $M = 4$ for the MDF adaptive filters. This means the size of the FFT required for the FLMS/UFLMS and the MDF adaptive filters are 1024 and 256 point, respectively. Table I summarizes the total execution time required for various adaptive filters. It is obvious that the MDF adaptive filters are more efficient in hardware utilization.

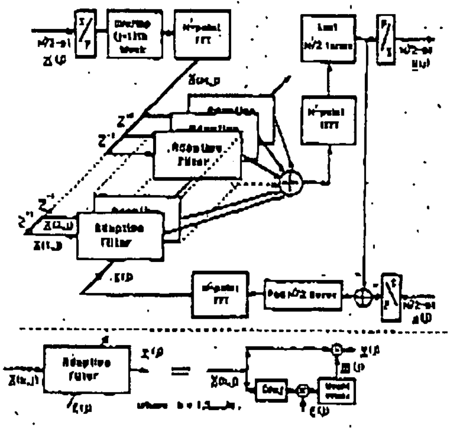


Fig. 1. A generalized MDF adaptive filter.

used, (7) becomes

$$W(m, j+1) = W(m,j) + \mu(j)\Phi(m,j) \quad (8)$$

$$\mu(j) = \mathrm{diag}[\mu_0(j), \mu_1(j), \cdots, \mu_k(j), \cdots, \mu_{N'/2-1}(j)] \quad (9)$$

$$\mu_k(j) = M\mu_0/Z_k(j), \quad k = 0, 1, \cdots, N'/2 - 1 \quad (10)$$

$$Z_k(j) = \beta Z_k(j-1) + (1-\beta)\left[\sum_{m=1}^{M} P_k(m,j)\right] \quad (11)$$

$$P_k(M,j) = X_k^*(M,j)X_k(M,j) \quad (12)$$

$$P_k(m,j) = P_k(m+1, j-1), \quad m = 1, 2, \cdots, M-1 \quad (13)$$

where $k$ is the frequency bin and $\beta$ is the smoothing constant; $\beta = 0.8$ is used here.

It should be mentioned that the self-orthogonalizing algorithm applied here is not exactly equivalent to the algorithm proposed in [5] and [6]. The power estimate in (10)–(13) makes use of the Welch methods [10] to average the periodograms of each block. This results in the reduction of variance and smoothing of the power spectrum.

## III. Simulation Results and Performance Analysis

In this section, we compare the performance of the MDF adaptive filter to other existing frequency domain adaptive filters. The simulation is based on identifying an FIR filter with the weights defined as

$$w_r^t = (-1)^r \exp[-0.04(r+1)], \quad r = 0, 1, \cdots, 127. \quad (14)$$

The input sample $x(j)$ is uniformly distributed between −100 and 100 and the quantization noise is added in the $d(j)$ by transforming the $d(j)$ from a real to an integer number. The normalized mean-square error (NMSE($k$)) averaged over ten independent runs

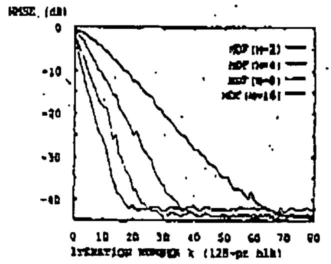


Fig. 2. Convergence characteristics of MDF adaptive filters for correlated input with eigenvalue ratio of 60 and self-orthogonalizing algorithm; $\mu_0 = 0.1$, $Z_k(0) = 5.0E+5$, with $M = 2, 4, 8,$ and 16.

in dB of the learning curve is defined as

$$\mathrm{NMSE}(k) = 10 \log \frac{\sum_{n=0}^{127}[d_n(k) - y_n(k)]^2}{\sum_{n=0}^{127}[d_n(k)]^2} \ \mathrm{dB}. \quad (15)$$

Fig. 2 illustrates that the convergence speed increases as the block size reduces when the step size $\mu_0$ is held constant. This observation agrees with the results in [3], in which the effect of block size in terms of time constant on the misadjustment is discussed. However, as the block size decreases, i.e., $M$ increases, the time constant given does not predict the convergence speed accurately. In Fig. 2, the MDF adaptive filter at $M = 16$ is not twice as fast as that at $M = 8$. This discrepancy indicates that the time domain analysis for the BLMS algorithm used in [3] is not strictly applicable to its frequency domain counterpart.

For a realistic comparison of adaptation speed, we first choose the optimal step size to give the fastest convergence for the FLMS algorithm, then obtain two sets of step size for the MDF. The first set of step size is selected to give the same steady-state error as that of FLMS, whereas the second set (denoted by the asterisk) is obtained via trial and error to give the best performance. The MDF algorithm with both sets of step size is found to be faster than the FLMS algorithm in Fig. 3. The fast convergence rate is attributed to the more frequent weight update process adopted in the MDF algorithm. The above observation is consistent with the analysis found in [11].

To make the MDF adaptive filter more efficient, we explore the possibility of not imposing the weight constraint on (5), (6) as the UFLMS [5] does, and called the *unconstrained multidelay block frequency domain* (UMDF) adaptive filter. This results in a savings of two FFT operations for each delay block. However, the UMDF adaptive filter is found to be slower and has a larger misadjustment depending on the relative magnitude of the weights to be identified and the number of delay blocks. Alternatively, we can impose a weight constraint on only one block of weights at each block time $j$. That is, (5) and (6) are only implemented for only one block, but not all blocks of weight vectors at each iteration. In so doing, we are effectively time multiplexing or alternatively applying the weight constraint on each block. Thus, the name *alternative unconstrained multidelay block frequency domain* (AUMDF) adaptive filter is used. Fig. 4 shows that the AUMDF adaptive filter has almost identical performance compared to the MDF when $M$ and $\mu_0$ are small. For $M = 16$ or $\mu_0$ large, there is some loss of performance for the AUMDF adaptive filter. Nevertheless, the choice of either an MDF, AUMDP, or UMDF adaptive filter is both application and hardware dependent.

## V. Conclusion

The MDF adaptive filter is derived and verified. It requires less memory storage, small FFT size, and allows different configurations to be chosen depending on the hardware used. In performance, the MDF adaptive filter has a smaller block delay and is faster. This is achieved by updating the weight vectors more often and reducing the total execution time in most of the DSP's. Furthermore, the total number of blocks needed can be changed dynamically without interrupting the normal operation. For example, one can add or drop one block of weight vector by checking the change in output error after some iterations to avoid the redundant operations. It is important to point out that the MDF adaptive filter is most suitable for real-time applications implemented on the DSP hardware, not on a general-purpose computer because the latter is not constrained by the small memory.

## References

[1] N. Furuya, Y. Itoh, Y. Maruyama, and Araseki, "Audio conference equipment with acoustic echo canceller," *NEC Res. Develop.*, pp. 18–23, Jan. 1983.

[2] E. R. Ferrara, "Fast implementation of LMS adaptive filters," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-28, pp. 474–475, Aug. 1980.

[3] G. A. Clark, S. K. Mitra, and S. R. Parker, "Block implementation of adaptive digital filters," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-29, pp. 744–752, June 1981.

[4] G. A. Clark, S. R. Parker, and S. K. Mitra, "A unified approach to time- and frequency-domain realization of FIR adaptive digital filters," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-31, pp. 1073–1083, Oct. 1983.

[5] D. Mansour and A. H. Gray, Jr., "Unconstrained frequency-domain adaptive filter," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-30, pp. 726–734, Oct. 1982.

[6] E. R. Ferrara, Jr., "Frequency-domain adaptive filter," in *Adaptive Filters*, C. F. N. Cowan and P. M. Grant, Eds. Englewood Cliffs, NJ: Prentice-Hall, 1985, ch. 6.

[7] J. C. Lee and C. K. Un, "Performance of transform-domain LMS adaptive digital filters," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-34, pp. 499–510, June 1986.

[8] J. S. Soo and K. K. Pang, "A new structure for block FIR adaptive digital filters," in *IREECON Int. Dig. Papers*, Sydney, Australia, Sept. 1987, pp. 364–367.

[9] E. O. Brigham, *The Fast Fourier Transform*. Englewood Cliffs, NJ: Prentice-Hall, 1974.

[10] P. D. Welch, "The use of fast Fourier transform for the estimation of power spectra," *IEEE Trans. Audio Electroacoust.*, vol. AU-15, pp. 70–73, June 1967.

[11] A. Feuer, "Performance analysis of the block least mean square algorithm," *IEEE Trans. Circuits Syst.*, vol. CAS-32, pp. 960–963, Sept. 1985.

A8

# HIGH-SPEED CONVOLUTION AND CORRELATION*

Thomas G. Stockham, Jr.
*Massachusetts Institute of Technology, Project MAC*
*Cambridge, Massachusetts*

## INTRODUCTION

Cooley and Tukey[1] have disclosed a procedure for synthesizing and analyzing Fourier series for discrete periodic complex functions.† For functions of period $N$, where $N$ is a power of 2, computation times are proportional to $N \log_2 N$ as expressed in Eq. (0).

$$T_{ct} \approx k_{ct} N \log_2 N \qquad (0)$$

where $k_{ct}$ is the constant of proportionality. For one realization for the IBM 7094, $k_{ct}$ has been measured at 60 $\mu$sec. Normally the times required are proportional to $N^2$. For $N = 1000$ speed-up factors in the order of 50 have been realized! Eq. (1b) synthesizes the Fourier series in question. The complex Fourier coefficients are given by the analysis equation, Eq. (1a).

$$F(k) = \sum_{j=0}^{N-1} f(j) w^{-jk} \qquad (1a)$$

$$f(j) = \frac{1}{N} \sum_{k=0}^{N-1} F(k) w^{jk} \qquad (1b)$$

where $w = e^{2\pi i/N}$, the principal $N$th root of unity. The functions $f$ and $F$ are said to form a discrete periodic complex transform pair. Both functions are of period $N$ since

$$F(k) = F(k + cN) \qquad (2a)$$

and

$$f(j) = f(j + cN) \qquad (2b)$$

## TRANSFORM PRODUCTS

Consider two functions $g$ and $h$ and their transforms $G$ and $H$. Let $G$ and $H$ be multiplied to form the function $C$ according to Eq. (3),

$$C(k) = G(k) \times H(k) \qquad (3)$$

and consider the inverse transform $c(j)$. $c(j)$ is given by Eq. (4)

$$\begin{aligned} c(j) &= \frac{1}{N} \sum_{J=0}^{N-1} g(J) h(j - J) \\ &= \frac{1}{N} \sum_{J=0}^{N-1} h(J) g(j - J) \qquad (4) \end{aligned}$$

as a sum of lagged products where the lags are performed circularly. Those values that are shifted from one end of the summation interval are circulated into the other.

The time required to compute $c(j)$ from either form of Eq. (4) is proportional to $N^2$. If one computes the transforms of $g$ and $h$, performs the multiplication of Eq. (3), and then computes the inverse

*Work reported herein was supported (in part) by Project MAC, an M.I.T. research program sponsored by the Advanced Research Projects Agency, Department of Defense, under Office of Naval Research Contract Number Nonr-4102(01).

†To be able to use this procedure the period must be a highly composite number.

transform of $C$, one requires a time given by Eq. (5)

$$\begin{aligned} T_{\text{circ}} &= 3\,k_{ct}\,N\log_2 N + k_m N \\ &= k_{\text{circ}}\,N(\log_2 N + \mu) \end{aligned} \tag{5}$$

where $k_{\text{circ}} = 3k_{ct}$, $\mu = k_m/k_{\text{circ}}$, and $k_m N$ is the time required to compute Eq. (3). Of course this assumes $N$ is a power of 2. Similar savings would be possible provided $N$ is a highly composite number.

## APERIODIC CONVOLUTION

The circular lagged product discussed above can be alternately regarded as a convolution of periodic functions of equal period. Through suitable modification a periodic convolution can be used to compute an aperiodic convolution when each aperiodic function has zero value everywhere outside some single finite aperture.

Let the functions be called $d(j)$ and $s(j)$. Let the larger finite aperture contain $M$ discrete points and let the smaller contain $N$ discrete points. The result of convolving these functions can be obtained from the result of circularly convolving suitable augmented functions. Let these augmented functions be periodic of period $L$, where $L$ is the smallest power of 2 greater than or equal to $M + N$. Let them be called $da(j)$ and $sa(j)$ respectively, and be formed as indicated by Eq. (6).

$$\begin{aligned} fa(j) &= f(j + j_0) && 0 \le j \le M - 1 \\ &= 0 && M \le j \le L - 1 \\ &= fa(j + nL) && \text{otherwise} \end{aligned} \tag{6}$$

where $j_0$ symbolizes the first point in the aperture of the function in question. The intervals of zero values permit the two functions to be totally non-overlapped for at least one lagged product even though the lag is a circular one. Thus, while the result is itself a periodic function, each period is an exact replica of the desired aperiodic result.

The time required to compute this result is given in Eq. (7).

$$T_{\text{apct}} = k_{\text{circ}}\,L(\log_2 L + \mu) \tag{7}$$

where $M + N \le L < 2(M + N)$. For this case, while $L$ must be adjusted to a power of 2 so that the high-speed Fourier transform can be applied, no restrictions are placed upon the values of either $M$ or $N$.

## SECTIONING

Let us assume that $M$ is the aperture of $d(j)$ and $N$ is that of $s(j)$. In situations where $M$ is considerably larger than $N$, the procedure may be further streamlined by sectioning $d(j)$ into pieces each of which contains $P$ discrete points where $P + N = L$, a power of 2. We require $K$ sections where

$$K = \text{least integer} \ge M/P \tag{8}$$

Let the $i$th section of $d(j)$ be called $d_i(j)$. Each section is convolved aperiodically with $s(j)$ according to the discussion of the previous section, through the periodic convolution of the augmented sections, $da_i(j)$ and $sa(j)$.

Each result section, $r_i(j)$, has length $L = P + N$ and must be additively overlapped with its neighbors to form the composite result $r(j)$ which will be of length

$$KP + N \ge M + N \tag{9a}$$

If $r_i(j)$ is regarded as an aperiodic function with zero value for arguments outside the range $0 \le j \le L - 1$, these overlapped additions may be expressed as

$$r(j) = \sum_{i=0}^{K-1} r_i(j - iP) \quad j = 0, 1, \ldots KP + N - 1 \tag{9b}$$

Each overlap margin has width $N$ and there are $K - 1$ of them.

The time required for this aperiodic sectioned convolution is given in Eq. (10).

$$\begin{aligned} T_{\text{sect}} &= k_{ct}(P + N)\log_2(P + N) \\ &\quad + 2Kk_{ct}(P + N)\log_2(P + N) \\ &\quad + Kk_{\text{aux}}(P + N) \\ &= k_{ct}(2K + 1)(P + N)\log_2(P + N) \\ &\quad + Kk_{\text{aux}}(P + N) \\ &\approx k_{ct}(2K + 1)(P + N)[\log_2(P + N) + \mu'] \end{aligned} \tag{10}$$

where $\mu' = k_{\text{aux}}/2k_{ct}$. $Kk_{\text{aux}}(P + N)$ is the time required to complete auxiliary processes. These processes involve the multiplications of Eq. (3), the formation of the augmented sections $da_i(j)$, and the formation of $r(j)$ from the result sections $r_i(j)$. For the author's realization in which core memory was used for the secondary storage of input and output data, $\mu'$ was measured to be 1.5, which gives $k_{\text{aux}} = 3k_{ct} \approx 300\,\mu$sec. If slower forms of auxiliary storage were employed, this figure would be enlarged slightly.

For a specific pair of values $M$ and $N$, $P$ should be chosen to minimize $T_{\text{sect}}$. Since $P + N$ must be a power of 2, it is a simple matter to evaluate Eq. (10) for a few values of $P$ that are compatible with this constraint and select the optimum choice. The size of available memory will place an additional constraint on how large $P + N$ may be allowed to become. Memory allocation considerations degrade the benefits of these methods when $N$ becomes too large. In extreme cases one is forced to split the kernel, $s(j)$, into packets, each of which is considered separately. The results corresponding to all packets are then added together after each has been shifted by a suitable number of packet widths. For the author's realization $N$ must be limited to occupy about ⅛ of the memory not used for the program or for the secondary storage of input/output data. For larger $N$, packets would be required.

## COMBINATION OF SECTIONS IN PAIRS

If both functions to be convolved are real instead of complex, further time savings over Eq. (10) can be made by combining adjacent even and odd subscripted sections $da_i(j)$ into complex composites. Let even subscripted $da_i(j)$ be used as real parts and odd subscripted $da_{i+1}(j)$ be used as imaginary parts. Such a complex composite can then be transformed through the application of Eqs. (1a), (3), and (1b) to produce a complex composite result section. The desired even and odd subscripted result sections $r_i(j)$ and $r_{i-1}(j)$ are respectively the real and imaginary parts of that complex result section.

This device reduces the time required to perform the convolution by approximately a factor of 2. More precisely it modifies $K$ by changing Eq. (8) to

$$K = \text{least integer} \geq M/2P \qquad (11)$$

For very large numbers of sections, $K$, Eq. (10) can be simplified to a form involving $M$ explicitly instead of implicitly through $K$. That form is given in Eq. (12)

$$T_{\text{fast}} \approx k_{ct} M((P + N)/P)[\log_2 (P + N) + \mu'] \qquad (12)$$

Since it makes no sense to choose $P < N$, for simple estimates of an approximate computation time we can write

$$T_{\text{fast}} \approx 2k_{ct}M[\log_2 N + \mu' + 1] \qquad (13)$$

## EMPIRICAL TIMES

The process for combined-sectioned-aperiodic convolution of real functions described above was implemented in the MAD language on the IBM 7094 Computer. Comparisons were made with a MAD language realization of a standard sum of lagged products for $N$ = 16, 24, 32, 48, 64, 96, 128, 192, and 256. In each case $M$ was selected to cause Eq. (11) to be fulfilled with the equal sign. This step favors the fast method by avoiding edge effects. However, $P$ was not selected according to the optimization method described above (under "Sectioning Convolution"), but rather by selecting $L$ as large as possible under the constraint,

$$\ln L \geq P/N \qquad (14)$$

This choice can favor the standard method.

Table 1 compares for various $N$ the actual computation times required in seconds as well as times in milliseconds per unit lag. Values of $M$, $K$, and $L$ are also given.

Relative speed factors are shown in Table 2.

## ACCURACY

The accuracy of the computational procedure described above is expected to be as good or better

Table 1. Comparative Convolution Times for Various $N$

| $N$ | 16 | 24 | 32 | 48 | 64 | 96 | 128 | 192 | 256 |
|---|---|---|---|---|---|---|---|---|---|
| $M$ | 192 | 208 | 384 | 416 | 768 | 832 | 1536 | 1664 | 3584 |
| $K$ | 2 | 1 | 2 | 1 | 2 | 1 | 2 | 1 | 1 |
| $L$ | 64 | 128 | 128 | 256 | 256 | 512 | 512 | 1024 | 2048 |
| | | | | | *Time in seconds* | | | | |
| $T_{\text{standard}}$ | 0.2 | 0.31 | 0.8 | 1.25 | 3.0 | 5.0 | 12 | 20 | 48 |
| $T_{\text{fast}}$ | 0.3 | 0.4 | 0.6 | 0.8 | 1.3 | 1.8 | 3.0 | 3.8 | 8.0 |
| | | | | | *Time in milliseconds per unit lag* | | | | |
| $T_{\text{standard}/M}$ | 1.0 | 1.4 | 2.0 | 3.0 | 3.9 | 6.0 | 7.8 | 12.0 | 13.3 |
| $T_{\text{fast}/M}$ | 1.5 | 1.9 | 1.5 | 1.9 | 1.6 | 2.1 | 1.9 | 2.2 | 2.2 |

Table 2. Speed Factors for Various $N$

| $N$ | 16 | 24 | 32 | 48 | 64 | 96 | 128 | 192 | 256 | 512 | 1024 | 2048 | 4096 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Speed factor | $\frac{2}{3}$ | $\frac{3}{4}$ | $\frac{4}{3}$ | 1.5 | 2.3 | 2.8 | 4.0 | 5.2 | 6 | 13* | 24* | 44* | 80* |

*Estimated values.

than that obtainable by summing products. Specific investigations of the accuracy of the program used to accumulate the data of Tables 1 and 2 are in process at the time of this writing. The above expectations are fostered by accuracy measurements made for floating-point data on the Cooley-Tukey procedure and a standard Fourier procedure. Since the standard Fourier procedure computes summed products, its accuracy characteristics are similar to those of a standard convolution which also computes summed products. Cases involving functions of period 64 and 256 were measured and it was discovered that two Cooley-Tukey transforms in cascade produced respectively as much, and half as much, error as a single standard Fourier transform. This data implies that the procedures disclosed here may yield more accurate results than standard methods with increasing relative accuracy for larger $N$.

## APPLICATIONS

Today the major applications for the computation of lagged products are digital signal processing and spectral analysis.

Digital signal processing, or digital filtering as it is sometimes called, is often accomplished through the use of suitable difference equation techniques. For difference equations characterized by only a few parameters, computations may be performed in times short compared to those required for a standard lagged product or the method described here. However, in some cases, the desired filter characteristics are too complex to permit realization by a sufficiently simple difference equation. The most notable cases are those requiring high frequency selectivity coupled with short-duration impulse response and those in which the impulse response is found through physical measurements. In these situations it is desirable to employ the techniques described here either alone or cascaded with difference equation filters.

The standard methods for performing spectral analysis[2] involves the computation of lagged products of the form

$$F(j) = \sum_{J=0}^{N-j-1} x(J)y(J+j) \qquad (15)$$

which, in turn, after weighting by so-called spectral windows are Fourier transformed into power spectrum estimates. Speed advantages can be gained when Eq. (15) is evaluated in a manner similar to that outlined above (under "Aperiodic Convolution") except that in this case $L$ is only required to exceed $N + \Omega$ where $\Omega$ is the number of lags to be considered. This relaxed requirement on $L$ is possible because it is not necessary to avoid the effect of performing the lags circularly for all $L$ lags but rather for only $\Omega$ of them. An additional constraint is that $\Omega$ be larger than a multiple of $\log_2 L$. The usual practice is to evaluate Eq. (15) for a number of lags equal to a substantial fraction of $N$. Since the typical situation involves values of $N$ in the hundreds and thousands, the associated savings may be appreciable for this application.

Digital spatial filtering is becoming an increasingly important subject.[3,4] The principles discussed here are easily extended to the computation of lagged products across two or more dimensions. Time savings depend on the total number of data points contained within the entire data space in question, and they depend on this number in a manner similar to that characterizing the one-dimension case.

## ACKNOWLEDGMENTS

The author is indebted to Charles M. Rader of the MIT Lincoln Laboratory for his ideas concerning the Cooley-Tukey algorithm and to Alan V. Oppenheim of the Electrical Engineering Department, MIT, for suggesting that high-speed convolutions might be realized through the utilization of that algorithm. During the preparation of this work the author became aware of the related independent efforts of Howard D. Helms, Bell Telephone Laboratories, and Gordon Sande, Jr., Princeton University.

## REFERENCES

1. J. W. Cooley and J. W. Tukey, "An Algorithm for the Machine Calculation of Complex Fourier Series," *Mathematics of Computation*, vol. 19, no. 90, pp. 297-301, (Apr. 1965).

2. R. B. Blackman and J. W. Tukey, *The Measurement of Power Spectra*, Dover Publications, New York, 1959; also *Bell System Technical Journal*, Jan. and Mar. 1958.

3. T. S. Huang and O. J. Tretiak, "Research in Picture Processing," *Optical and Electro-Optical Information Processing*, J. Tippett et al, eds., MIT Press, Cambridge, Mass., 1965, Chap. 3.

4. T. S. Huang, "PCM Picture Transmission," *IEEE Spectrum*, vol. 2, no. 12, pp. 57–63 (Dec. 1965).

A9

~~added or subtracted at local screen areas in accordance with the potential applied to the backplate. This resultant image may then be read out electrically and transmitted to a remote station over low bandwidth lines. A form of graphic communication is thus possible between the display unit and computer or directly between multiple display units without requiring storage components external to the display tubes.~~

~~ACKNOWLEDGMENT~~

~~The authors wish to thank C. X. Duncombe and J. P. Hoeksira for their assistance.~~

~~B. KAZAN~~
~~I. F. CHANG~~
~~IBM Thomas J. Watson Res. Ctr.~~
~~Yorktown Heights, N. Y. 10598~~

~~REFERENCES~~

~~[1] B. M. Gurley, "Light pen links computer to operator," *Electronics*, vol. 32, Nov. 20, 1959, pp. 85–87.~~

~~[2] R. H. Anderson, "A simplified direct viewing bistable storage tube," *IEEE Trans. Electron Devices*, vol. ED-14, Dec. 1967, pp. 838–844.~~

~~[3] A. V. Haeff, "A memory tube," *Electronics*, vol. 20, Sept. 1947, pp. 80–83.~~

~~[4] B. Kazan and M. Knoll, *Electronic Image Storage*. New York: Academic Press, 1968.~~

~~[5] R. Hofstadter, "Photoconductivity in willemite," *Phys. Rev.*, vol. 54, 1938, p. 864.~~

~~[6] R. C. Herrman and R. Hofstadter, "Photoconductivity of a natural willemite crystal," *Phys. Rev.*, vol. 59, Jan. 1, 1941, pp. 79–84.~~

## Real Signals Fast Fourier Transform: Storage Capacity and Step Number Reduction by Means of an Odd Discrete Fourier Transform

**Abstract—An odd discrete Fourier transform (ODFT) which relates in several ways to the usual discrete Fourier transform (DFT) is introduced and discussed. Its main advantage is that it can readily be applied to spectrum and correlation computations on real signals, by halving the storage capacity and greatly reducing the number of necessary steps.**

### DEFINITION

Let $X_0, X_1, \cdots, X_{N-1}$ be $N$ complex samples, and let the odd discrete Fourier transform (ODFT) be defined by the relation

$$C_K = \frac{1}{N}\sum_{n=0}^{N-1} X_n \exp\left[-2\pi j\frac{(2K+1)n}{2N}\right], \qquad K = 0, 1, \cdots, N-1. \quad (1)$$

### RECIPROCITY

The samples $X_n$ can be obtained from coefficients $C_K$ by the inverse relation, which defines the inverse ODFT:

$$X_n = \sum_{K=0}^{N-1} C_K \exp\left(2\pi j\frac{2K+1}{2N}n\right). \quad (2)$$

Manuscript received March 1, 1971.

### CONNECTION WITH THE DFT

The discrete Fourier transform (DFT) on $N$ samples is defined as[1]

$$A_K = \frac{1}{N}\sum_{n=0}^{N-1} X_n \exp\left(-2\pi j\frac{Kn}{N}\right), \qquad K = 0, 1, \cdots, N-1. \quad (3)$$

Relation (1) can be written as

$$C_N = \frac{1}{N}\sum_{n=0}^{N-1}\left[X_n \exp\left(-2\pi j\frac{n}{2N}\right)\right] \cdot \exp\left(-2\pi j\frac{Kn}{N}\right) \quad (4)$$

where the coefficients $C_K$ are the DFT of terms

$$X_n\left(\exp - 2\pi j\frac{n}{2N}\right).$$

It should also be noted that the $C_K$ are the odd lines of the $2N$ samples DFT of $X'_n$ defined by

$$X'_n = X_n, \qquad n = 0, 1, \cdots, N-1$$
$$X'_n = 0, \qquad n = N, N+1, \cdots, 2N-1.$$

This DFT can be written thus:

$$B_h = \frac{1}{N}\sum_{n=0}^{2N-1} X'_n \exp\left(-2\pi j\frac{hn}{2N}\right) = \frac{1}{N}\sum_{n=0}^{N-1} X_n \exp\left(-2\pi j\frac{hn}{2N}\right) \quad (5)$$

and

$$\text{when } h = 2K+1, B_{2K+1} = C_K$$
$$\text{when } h = 2K, B_{2K} = A_K.$$

From relation (4), one can consider coefficients $C_K$ as being the spectrum of frequency translated samples $X_n$, with $f = -1/2N$, or from relation (5), as a sampling of $X_n$ spectrum with the same rate as in the usual DFT, but shifted by half a sample.

### CONVOLUTION RELATION

Let $X_n$ and $Y_n$ be two sample series, and $C_K$ and $D_K$ the corresponding ODFT. If we compute the inverse ODFT of the $C_K \cdot D_K$ series, we obtain

$$\Gamma_q = \sum_{K=0}^{N-1} C_K D_K \exp\left(2\pi j\frac{2K+1}{2N}q\right) = \frac{1}{N^2}\sum_{n=0}^{N-1}\sum_{m=0}^{N-1}\sum_{K=0}^{N-1}\left[X_n \cdot \exp\left(-2\pi j\frac{2K+1}{2N}n\right)\right] \cdot \left[Y_m \exp\left(-2\pi j\frac{2K+1}{2N}m\right)\right] \cdot \exp\left(2\pi jq\frac{2K+1}{2N}\right)$$

$$\Gamma_q = \frac{1}{N}\left[\sum_{n=0}^{q} X_n Y_{q-n} - \sum_{n=q+1}^{N-1} X_n Y_{q-n+N}\right]. \quad (6)$$

The relation differs from the cyclic $X_n$ and $Y_n$ convolution by the sign of the second sum, but can be used when $N$ is even to calculate $N/2$ convolution values of $N/2$ samples $X_n$, $n=0, \cdots, N/2-1$ with $N$ $Y_n$ samples. This is derived from the fact that, when $q > N/2-1$, relation (6) becomes

$$\Gamma_q = \frac{1}{N}\sum_{n=0}^{N/2-1} X_n Y_{q-n}.$$

### ODFT APPLIED TO REAL SIGNALS PROCESSING

The main interest of ODFT remains in real signals processing. We have

$$C_{N-1-K} = \sum_{n=0}^{N-1} X_n \cdot \exp\left[-2\pi j\frac{2(N-1-K)+1}{2N}n\right] = \sum_{n=0}^{N-1} X_n \exp\left(2\pi j\frac{2K+1}{2N}n\right)$$

and if the $X_n$ are real,

$$C_{N-1-K} = C_K^*. \quad (7)$$

Odd and even spectrum lines are then conjugated. In fact, when $N$ is even, the computation of even lines $C_{2q}$ can be achieved separately:

$$C_{2q} = \sum_{n=0}^{N-1} X_n \exp\left(2\pi j\frac{4q+1}{2N}n\right) = \sum_{n=0}^{N/2-1} X_n \exp\left(2\pi j\frac{4q+1}{2N}n\right) + \sum_{n=N/2}^{N-1} X_n \exp\left(-2\pi j\frac{4q+1}{2N}n\right) = \sum_{n=0}^{N/2-1}(X_n - jX_{n+N/2}) \cdot \exp\left(-2\pi j\frac{4q+1}{2N}n\right)$$

$$C_{2q} = \sum_{n=0}^{N/2-1}\left[(X_n - jX_{n+N/2})\exp\left(-2\pi j\frac{n}{2N}\right)\right] \cdot \exp\left(-2\pi j\frac{nq}{N/2}\right).$$

The $C_{2q}$ series appears as the $N/2$ DFT of the complex signal:

$$U_n = (X_n - jX_{N/2+n})\exp\left(-2\pi j\frac{n}{2N}\right).$$

Then one can obtain the spectrum lines of the positive frequency range by retaining the $C_{2q}$ series for $q = 0, \cdots, N/4$, and by conjugating the $C_{2q}$ series for $q > N/4$ [relation (7)]. Using the reciprocity relation (2), the $X_n$ can be obtained from $C_{2q}$ by an $N/2$ sample DFT giving the $U_n$, and a demodulation by $\exp[2\pi j(n/2N)]$.

In the same way, it is possible to perform correlation or convolution computations on real signals, by means of an $N/2$ sample DFT after (or before) an extraction of $U_n$ from $X_n$ (or recip-

[1] J. W. Cooley, P. A. W. Lewis, and P. D. Welch, "The finite Fourier transform," *IEEE Trans. Audio Electroacoust.*, vol. AU-17, June 1969, pp. 77–85.

rocally), according to the following formula:

$$\begin{tikzcd}[column sep=small]
X_n - U_n = (X_n + jX_{n+N/2}) \exp\left(-2\pi j \frac{n}{2N}\right) \arrow[r, "\text{TFD } N/2"] & A_K \arrow[d, dash] & \\
& A_K B_K \arrow[r, "\text{TFD } N/2"'] & (\Gamma_q - j\Gamma_{q+N/2}) \exp\left(2\pi j \frac{q}{2N}\right) \rightarrow \Gamma_q - j\Gamma_{q+N/2}. \\
Y_n \rightarrow V_n = (Y_n + jY_{n+N/2}) \exp\left(-2\pi j \frac{n}{2N}\right) \arrow[r, "\text{TFD } N/2"'] & B_K \arrow[u, dash] &
\end{tikzcd}$$

## Conclusion

Spectrum analysis and correlation or convolution operations performed on real signals can be achieved by means of the ODFT which becomes, after a simple processing, a classic DFT requiring one-half the number of samples. The number of computation steps is greatly reduced, and the size of the necessary core storage is reduced by half.

J. L. Vernet
Thomson-CSF
06. Cagnes-sur-Mer, France

# A New Type of Frequency-Stabilized Gunn Oscillator

Abstract—A new type of frequency-stabilization method for a Gunn oscillator consisting of a high-Q single-tuned oscillator circuit is described. The frequency stability of this stabilized oscillator was 5×10⁻⁵ over a temperature change from −10 to 60°C at 20.970 GHz. This result satisfies the requirement for a 20-GHz radio relay system.

New microwave solid-state oscillators such as IMPATT and Gunn oscillators, have been coming into practical use; for example, local oscillators are being used for transmitters and receivers of 20-GHz radio relay systems. But these have one serious practical problem, that of frequency stability for temperature change. This problem has been studied by many researchers and engineers whose work has resulted in an injection locking to a master oscillator and a direct or indirect coupled-cavity stabilized oscillator. According to oscillator size, the latter is proper for a local oscillator, but this stabilization method has several practical problems: mode jumping during operation as well as in the instant of switching, frequency adjustability, and decline of output power. These problems are caused by the existence of two resonance circuits—the oscillator resonance circuit and the stabilization cavity. Therefore, these oscillators have two possible oscillation modes and, accordingly, frequency hysteresis over the pulling range.

To avoid such problems, we designed a high-$Q$ single-tuned oscillator circuit shown in Fig. 1. This circuit consists of a rectangular-waveguide (VBR-220) Gunn-diode mount, a half-wavelength waveguide section, a high-$Q$ external cavity coupled to the waveguide section through a slit and a matched dummy load.

A simplified equivalent representation of the circuit is shown in Fig. 2, together with a simplification of the Gunn diode as $-G_d$. The cavity, having its own resonant frequency $f_0$ and the unloaded $Q$ of $Q_0$, is coupled to the waveguide section, having characteristic admittance $Y_0$, through a slit with VSWR $S$ into the assembly with the cavity tuned.

Then the normalized admittance of the assembly for stabilization $y_L$ is given as follows:

$$y_L \triangleq Y_L/Y_0 = g + jb \tag{1}$$

$$g = 1 - S(S-1)\frac{1}{S^2 + (2Q_0\delta)^2} \tag{1a}$$

$$b = (S-1)\frac{(2Q_0\delta)}{S^2 + (2Q_0\delta)^2} \tag{1b}$$

where $\delta = (f - f_0)/f_0$ and the effect of the deviation in electrical length at the half-wavelength waveguide section is ignored since it is considered quite minor.

The susceptance $b$ of $y$ in accordance with $\omega$ is shown in Fig. 3, where the derivative of $b$ with respect to $\omega$, $\partial b/\partial\omega$ is negative at regions I and III and positive at region II. Therefore, only region II provides the possibility of oscillation, and the width $\delta_S$ of region II is given as follows from (1b):

$$\delta_S = S/Q_0. \tag{2}$$

That is, the larger $S$ gives the larger $\delta_S$ as well as the larger $|b|$ which results in a wider stabilized range. Besides, the smaller conductance $g$ and, accordingly, the larger available output power to the load are brought by the larger $S$. That is to say conclusively, in this new system, both wider stabilized region and larger output power are realized simultaneously without any trouble such as mode jumping or hysteresis.

Experimental temperature characteristics of a Gunn oscillator used in the new stabilized system are as follows.

| | |
|---|---|
| Frequency stability | 5 × 10⁻⁵ for −10 to 60°C at 20.970 GHz. |
| Power deviation | 1 dB for −10 to 60°C at 13.9 dBm. |
| Power decline | 2.5 dB relative to conventional type. |
| Pushing figure | 100 kHz/100 mV. |

These characteristics are shown in Fig. 4.

As a reference, a Gunn oscillator with the conventional arrangement ($Q_{ext} \approx 30$), using the same diode and mount with that of the abovementioned new oscillator, shows a frequency

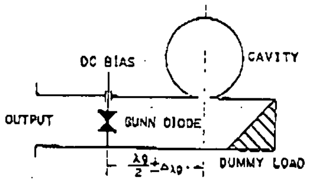


Fig. 1. Arrangement of a new type of frequency-stabilized Gunn oscillator.

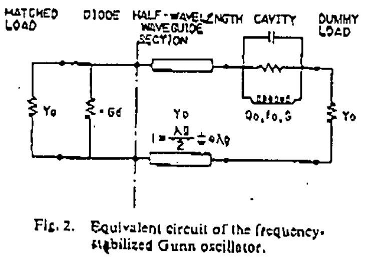

MATCHED LOAD
DIODE
HALF-WAVELENGTH WAVEGUIDE SECTION
CAVITY
DUMMY LOAD
Yo
-Gd
Yo
Qo, fo, S
Yo
$l = \frac{\lambda g}{2} \pm \Delta\lambda g$

Fig. 2. Equivalent circuit of the frequency-stabilized Gunn oscillator.

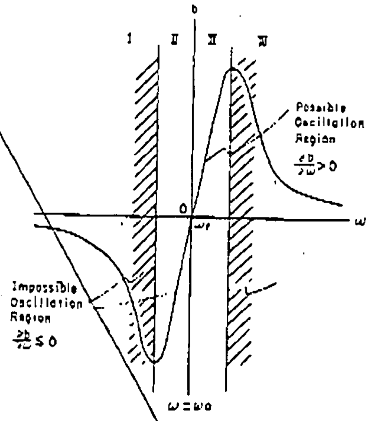


Fig. 3. Susceptance characteristics $b$ of $y_L$.

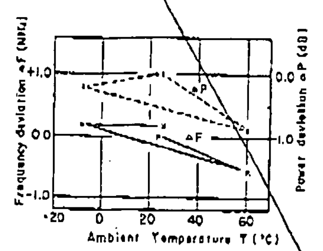


Fig. 4. Temperature characteristics. Diode (GD518B): $V_{bias} = 6.40$ V; waveguide (UBR-220); cavity ($Q_0 = 15\,000$, $TE_{011}$ mode); VSWR $S = 5.8$; output power loss 2.5 dB; pushing figure $\Delta F/\Delta V = 100$ kHz/100 mV; $F_0 = 20.97038$ GHz; $P_0 = 13.9$ dBm.

Manuscript received March 1, 1971.

A 10

### A Method of Obtaining Transfer Function

A conventional method for determining a system function is as follows. Assume that when a signal sin $(wt + \theta)$ is given at the input of the system, $A$ sin $(wt + \theta)$ is obtained at the output. The problem is to obtain values of $A$ and $\theta$ with respect to $w$. By Fourier analysis of $A$ sin $(wt + \theta)$, the following equations are obtained:

$$R(w) = \frac{w}{\pi} \int_0^{2\pi/w} A \sin(wt + \theta) \cos wt \, dt = A \sin \theta \qquad (5)$$

$$I(w) = \frac{w}{\pi} \int_0^{2\pi/w} A \sin(wt + \theta) \sin wt \, dt = A \cos \theta. \qquad (6)$$

From (5) and (6), $\theta$ can be determined as follows:

$$\theta = \arctan (R(w)/I(w)). \qquad (7)$$

From this, $A$ is obtained:

$$R(w) \sin \theta + I(w) \cos \theta = A. \qquad (8)$$

Substituting (4) into (5) and (6),

$$R(w) = \frac{1}{\pi} \sum \Delta_i \sin wt_i \qquad (9)$$

$$I(w) = \frac{1}{\pi} \sum \Delta_i \cos wt_i \qquad (10)$$

where $t_i$ is a sampling instant and $\Delta_i$ is a constant value $K$ or $-K$. From the above result, the equipment for measuring frequency characteristics has been constructed, as shown in Fig. 1. In Fig. 1, the delay is used to obtain a cos $wt$. $A$ * cos $\theta$ is calculated with ADD 1 and $A$ * sin $\theta$ with ADD 11. The output pulse series of $\Delta$-$M$ becomes the command signal of ADD or SUBTRACT. The scalar adjusts the output level of a measured signal. The multipliers are used to obtain $\theta$ and $A$ in (7) and (8), hence high speed is not required. Furthermore, since they operate sequentially, they can be replaced with ADD 1 and ADD 11. It should be noted that $\Delta$-$M$ works at the rate of 256 times frequency of a measured signal.

#### Experiment

As an example, the result obtained from measuring the amplitude and phase characteristics of a simple $RC$ circuit is shown in Fig. 2. From the experimental results, the resolution of phase is within 0.5° and the amplitude error is less than 0.1 dB.

#### Acknowledgment

The author wishes to thank Prof. J. Kawata and Prof. T. Kubota for their useful suggestions in the course of this work.

#### References

[1] F. de Jager, "Delta modulation, a method of PCM transmission using a one-unit code," *Philips Res. Rep.*, vol. 7, 1952.
[2] K. Nitadori, "Statistical analysis of delta PCM," *J. Inst. Electron. Commun. Eng. Jap.*, vol. 48, Feb. 1965.

## Odd-Time Odd-Frequency Discrete Fourier Transform for Symmetric Real-Valued Series

G. BONNEROT AND M. BELLANGER

*Abstract*—An efficient algorithm is obtained for the DFT of $N$ point symmetric real-valued series if time samples are taken as odd multiples of half the sampling period $T/2$ and frequency samples are taken as odd multiples of $1/2NT$.

Manuscript received August 14, 1975.

The authors are with Télécommunications Radioélectriques et Téléphoniques, Plessis Robinson, France.

### I. Introduction

Fourier transforms of symmetric real-valued input series are of interest in several areas. A first application concerns correlation techniques, as the autocorrelation function is an even real function; another important application in communications has recently been pointed out [1], dealing with the digital frequency multiplexing of real signals.

An algorithm adapted to this specific problem has been presented in [2]; eliminating all unnecessary operations for a symmetric real input series permits the total computation time to be approximately half that required for the general case of arbitrary real input. This technique is attractive when programmed on a general purpose computer, but it is not easily implemented in specific hardware. The purpose of this letter is to introduce a new algorithm that features similar computational advantages and leads to a quite simple design as it carries out the transform of an $N$ point symmetric real valued series basically with a $N/4$ complex-input discrete Fourier transform (DFT) computer (provided $N$ is an integer multiple of 4).

### II. Derivation of the Algorithm

It has been shown in [3] that considering sampling points on the frequency axis located at odd multiples of $1/2NT$, $T$ being the sampling period of the time series and $N$ the transform size, leads to an odd-frequency DFT well adapted to the transform of real signals. In this letter, sampling points on the time axis located at odd multiples of $T/2$ are also considered, and then an odd-time odd-frequency DFT (O$^2$DFT) is obtained.

*Definition:* Let the O$^2$DFT be defined by the following relation in which $X_n (n = 0, 1, \cdots, N-1)$ are $N$ time samples and $C_k (k = 0, 1, \cdots, N-1)$ are $N$ frequency samples.

$$C_k = \frac{1}{N} \sum_{n=0}^{N-1} X_n \exp [-2\pi j (2k+1)(2n+1)/4N]. \qquad (1)$$

The time samples $X_n$ can be obtained from the frequency samples $C_k$ by the inverse relation which defines the inverse O$^2$DFT.

$$X_n = \sum_{n=0}^{N-1} C_k \exp [2\pi j (2k+1)(2n+1)/4N]. \qquad (2)$$

#### *A. Properties of O$^2$DFT with Real Input and Output*

It is easily shown from (1) that, if $\{X_n\}$ are real numbers, then the following relation holds, $C_k^*$ being the complex conjugate of $C_k$,

$$C_{N-1-k} = -C_k^*. \qquad (3)$$

Similarly, if the $C_k$ are real numbers,

$$X_{N-1-k} = -X_k^*. \qquad (4)$$

It is worth pointing out that, with $N$ even, the numbers $X_0, X_{N/2}$ and $C_0$, $C_{N/2}$ are no longer specific cases, as they are with standard DFT, and only $C_{2k} (k = 0, 1, \cdots, N/2 - 1)$ and $X_{2n} (n = 0, 1, \cdots, N/2 - 1)$ need to be computed. If the additional hypothesis is made that input is real and odd $(X_{N-1-k} = -X_k)$, and $N$ is an integer multiple of 4, the transform establishes a relation between two sets of $N/2$ real numbers and can be expressed as a product of three matrix factors: the conventional matrix of the $N/4$ DFT and two diagonal matrices.

The same result holds for an arbitrary real input $\{X_n\}$, when only the real part of $\{C_k\}$ needs to be computed. The input set $\{X_n\}$ can be replaced by the odd set $\{Y_n\} = \{(X_n - X_{N-1-n})/2\}$.

To prove this let us first consider the $\{C_{2k}\}$:

$$C_{2k} = \frac{1}{N} \sum_{n=0}^{N-1} X_n \exp [-2\pi j (4k+1)(2n+1)/4N].$$

Considering the complex multiplying factor of $X_{N/2+n}$.

$$\exp [-2\pi j (4k+1)(2n+N+1)/4N]$$
$$= -j \exp [-2\pi j (4k+1)(2n+1)/4N]$$

the summation reduces to

$$C_{2k} = \frac{1}{N} \sum_{n=0}^{N/2-1} (X_n - j X_{N/2+n}) \exp [-2\pi j (4k+1)(2n+1)/4N].$$

The term $C_{2k+N/2}$, which is one term of the array $\{C_{2k}\}$ if $N$ is a multiple of 4, is given by

$$C_{2k+N/2} = \frac{1}{N} \sum_{n=0}^{N/2-1} (X_n - j X_{N/2+n}) \cdot \exp\left[-2\pi j(4k+1+N)(2n+1)/4N\right].$$

The multiplying factor of the term with the index $n$ is

$$\exp\left[-2\pi j(4k+1+N)(2n+1)/4N\right] = (-j)^{2n+1} \exp\left[-2\pi j(4k+1)(2n+1)/4N\right].$$

The combination of $C_{2k}$ and $jC_{2k+N/2}$ gives the second reduction of the summation, by the elimination of the odd-indexed $X_n$ points.

$$C_{2k} + jC_{2k+N/2} = \frac{2}{N} \sum_{n=0}^{N/4-1} (X_{2n} - jX_{N/2+2n}) \cdot \exp\left[-2\pi j(4k+1)(2n+1)/4N\right].$$

By developing the product $(4k+1)(4n+1)$ we get

$$C_{2k} + jC_{2k+N/2} = \frac{2}{N} \sum_{n=0}^{N/4-1} (X_{2n} - jX_{N/2+2n}) \cdot \exp\left[-2\pi j\left(\frac{nk}{N/4} + \frac{k+1/8}{N} + \frac{n+1/8}{N}\right)\right].$$

The constant term $1/4N$ has been split into two equal parts. After proper factorization, the following is produced:

$$C_{2k} + j\,C_{2k+N/2} = \frac{2}{N} \exp\left[-2\pi j \frac{k+1/8}{N}\right] \sum_{n=0}^{N/4-1} \left[X_{2n} - jX_{N/2+2n}\right] \cdot \exp\left[-2\pi j \frac{n+1/8}{N}\right] \exp\left[-2\pi j \frac{nk}{N/4}\right].$$

Let $D_0$ be the diagonal matrix having the dimension $N/4$ and the diagonal element $\exp\left[-2\pi j(p+1/8)/N\right]$ with $p = 0, 1, \cdots, N/4 - 1$, and let $T_{N/4}$ be the $N/4$ DFT matrix as defined in [4] and [5]; then the transform is expressed in terms of matrix factors by

$$\frac{N}{2}\left[C_{2k} + j\,C_{2k+N/2}\right] = (D_0)(T_{N/4})(D_0)\left[X_{2n} - jX_{N/2+2n}\right]$$

where [ ] are vectors comprising $N/4$ elements.

### B. Inverse Transform

As far as the inverse transform is concerned, due to the equality $C_{N-1-k} = -C_k$, equation (2) can be rewritten as

$$X_n = \sum_{k=0}^{N-1} C_{N-1-k} \exp\left[+2\pi j(2N-2-2k+1)(2n+1)/4N\right]$$

$$X_n = \sum_{k=0}^{N-1} C_k \exp\left[-2\pi j(2k+1)(2n+1)/4N\right].$$

Then a device performing the operations expressed by the matrix product $(D_0)\,(T_{N/4})\,(D_0)$ can be used for direct and inverse transform without any modification.

In summary, for the calculation of the DFT of a set of $N$ real numbers with an odd symmetry

$$X_{N-1-n} = -X_n$$

the procedure can be described as follows.

1) Consider the complex vector comprising $N/4$ elements:

$$[X_{2n} - jX_{N/2+2n}].$$

2) Multiply each term by a complex weighting number equal to $\exp\left[-2\pi j(n+1/8)/N\right]$ which provides an $N/4$ element complex vector $[U_n]$.

3) Compute the standard $N/4$ DFT of the series $[U_n]$; the result will be a complex vector $[V_k]$ of dimension $N/4$.

4) Multiply each of the $N/4$ complex elements of the vector $[V_k]$ obtained by a complex weighting number similar to those used in phase two:

$$\exp\left[-2\pi j(k+1/8)/N\right].$$

The real and imaginary parts of the obtained vector $[Z_k]$ are the desired results:

$$Z_k = \frac{N}{2}(C_{2k} + j\,C_{2k+N/2}).$$

Similar procedures can be used and lead to the same reductions of computation amount and storage in the following applications: The input sample set is real with even symmetry, and the input sample set is pure imaginary with odd or even symmetry.

## III. Conclusion

A procedure based on the O$^2$DFT has been shown to bring significant reductions of computation amount and storage in the calculation of cosine or sine series for real data. In terms of hardware, any equipment performing a standard complex DFT can be used with minor modifications.

## References

[1] J. L. Daguet, M. Bellanger, and G. Bonnerot, "Méthode simplifiée de multiplexage en fréquence de signaux numériques réels," *Câbles Transmissions*, July 1975.

[2] H. Ziegler, "A fast Fourier transform algorithm for symmetric real-valued series," *IEEE Trans. Audio Electroacoust.*, vol. AU-20, pp. 353-356, Dec. 1972.

[3] J. L. Vernet, "Real signals fast Fourier transform: Storage capacity and step number reduction by means of an odd discrete Fourier transform," *Proc. IEEE* (Lett.), vol. 59, pp. 1531-1532, Oct. 1971.

[4] M. C. Pease, "An adaptation of the fast Fourier transform for parallel processing," *J. Ass. Comput. Mach.*, vol. 15, pp. 252-264, Apr. 1968.

[5] H. Sloate, "Matrix representations for sorting and the fast Fourier transform," *IEEE Trans. Circuits Syst.*, vol. CAS-21, pp. 109-116, Jan. 1974.

# Laser Scanning of MOS IC's Reveals Internal Logic States Nondestructively

DAVID E. SAWYER AND DAVID W. BERNING

*Abstract*—A laser scanning system has been used to observe the internal logic pattern in a MOS LSI device in a nondestructive manner. The laser scanner has also been used to selectively change logic states deep within the device. Pictures of the logic patterns revealed by the scanner are discussed.

A laser flying-spot scanner has been used to observe logic information passing through a MOS shift register. Previously, attempts were made to use the scanning electron microscope, but MOS devices tend to become nonoperational when examined in this manner [1]. The shift register was a static dual 128-bit p-MOS ion-implanted device, and the operation of the circuit was easily observed without altering the device characteristics. In order to be able to observe the logic flow in the device with the laser scanner, the package leads are connected as appropriate for normal circuit operation. The information which describes the circuit operation is extracted by monitoring variations in power supply current to the device.

The scanning system will be described in detail elsewhere [2]. The system conceptually is quite simple: the beam from a 0.633-$\mu$m continuous-wave He–Ne laser is deflected sequentially from two mirrors oscillating in orthogonal directions to generate a scanning raster, and this raster is focused on the specimen to be investigated. The electron beam on a cathode-ray-tube display screen is deflected in synchronism with the laser scan. A display of the specimen's small-signal photo-

Manuscript received August 25, 1975; revised October 1, 1975. This research was conducted as part of the NBS program on Semiconductor Measurement Technology with principal funding from the Defense Research Projects Agency through ARPA Order 2397.

The authors are with the Electronic Technology Division, National Bureau of Standards, Washington, DC 20234.

A 11

# DIGITAL PROCESSING OF SIGNALS

## THEORY AND PRACTICE

**Second Edition**

MAURICE BELLANGER
Scientific Director
T.R.T. Le Plessis-Robinson, France

*Foreword by*
Professor Pierre Aigrain
Secretary of State for Research, France

*A Wiley–Interscience Publication*

JOHN WILEY & SONS
Chichester · New York · Brisbane · Toronto · Singapore

Reprinted April 1990

This edition is a translation of Traitement
Numérique Du Signal--Théorie Et Pratique (3rd Edition),
published by Masson, Paris.
© Masson, Editeur et CNEST-ENST Paris (3rd Edition).

*Library of Congress Cataloging in Publication Data:*

Bellanger, Maurice.
[Traitement numérique du signal. English]
Digital processing of signals: theory and practice/Maurice
Bellanger; foreword by Pierre Aigrain; translated by Joan
McMullan.—2nd ed.
p. cm.
Translation of: Traitement numérique du signal.
"A Wiley-Interscience publication."
Includes bibliographies and index.
ISBN 0471 92101 7 (pbk.)
1. Signal processing—Digital techniques. I. Title.
TK5102.5.B3613 1988 88-20732
621.48'043—dc 19 CIP

*British Library Cataloguing in Publication Data:*

Bellanger, Maurice
Digital processing of signals.—2nd ed.
1. Digital signals. Processing
I. Title II. Traitement numerique du
signal. *English*
621.38'043

ISBN 0471 92101 7

Phototypesetting by Thomson Press (India) Limited; New Delhi
Printed and bound in Great Britain by Courier International Ltd,
Tiptree, Essex

# CONTENTS

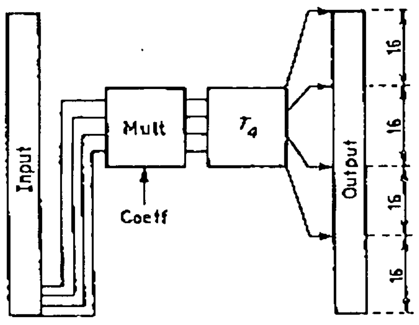


*Fig. 3.4 Transposed stage of radix-4 transform of order 64*

16 words apart. In principle, the first stage does not contain the part which performs the multiplications, but in practice, tailoring one stage runs the risk of complicating the timing unit, and it is often preferable to preserve the same structure of all the stages by introducing unit multiplications.

The search for identical stages may not be the most important objective, as would be the case in software, for example. In this case we only want to eliminate having to make the global permutation $Q_N^r$ over an ensemble of numbers either before or after transformation, because this permutation is difficult to program and is time-consuming, especially if access to machine language is not possible. The global permutation must, therefore, be incorporated into the various stages.

Let us take the recurrence equation which relates $T_N$ to $T_{N/r}$ and which was given in the preceding section as

$$T_N = P_N^r(T_{N/r} \times I_r)\Delta_N(I_{N/r} \times T_r)$$

By applying the commutation formula (3.13) this becomes:

$$T_N = (I_r \times T_{N/r})P_N^r\Delta_N(I_{N/r} \times T_r)$$

and the following expression is obtained using iteration:

$$T_N = \prod_{i=1}^{\log_r(N)} (I_{N/r^i} \times P_{r^i})(I_{N/r^i} \times \Delta_{r^i})(I_{N/r} \times T_r) \qquad (3.21)$$

This transform is calculated in $\log_r(N)$ stages, each comprising three operations:

(1) A combination of data with additions and subtractions corresponding to the matrix $(I_{N/r} \times T_r)$;
(2) A complex multiplication following the matrix $(I_{N/r^i} \times \Delta_{r^i})$;
(3) A permutation operation which is different at each stage and which is defined by $(I_{N/r^i} \times P_{r^i})$.

The data are both supplied and recovered in their natural order.

A program using this algorithm can result in an improvement of more than 30% in the calculation time when compared to a program with bit reversal. However, the calculation cannot be made with a single-data memory because the pairs of numbers change position at each stage.

The methods which have been given allow a unified representation of the algorithms for the fast Fourier transform and offer the user the possibility of determining and realizing the most suitable and effective algorithm in each case. They are equally useful for important particular cases.

## 3.5 PARTIAL TRANSFORMS

The transforms which have been studied in the above sections relate to sets of $N$ numbers which may be complex. In a fine-spectrum analysis it can happen that the order of the transform $N$ becomes very large while we are interested in knowing only a reduced number of points in the spectrum. The limitation of the calculation to useful single points can then permit a large saving.

Let us calculate the partial transform defined by the following equation, where $r$ is a factor of $N$:

$$\begin{bmatrix} X_p \\ X_{p+1} \\ X_{p+2} \\ \vdots \\ X_{p+r-1} \end{bmatrix} = \begin{bmatrix} 1 & W^P & W^{2P} & \cdots & W^{(N-1)P} \\ 1 & W^{P+1} & W^{2(P+1)} & \cdots & W^{(N-1)(P+1)} \\ 1 & W^{P+2} & W^{2(P+2)} & \cdots & W^{(N-1)(P+2)} \\ \vdots & \vdots & & & \vdots \\ 1 & W^{(P+r-1)} & \cdots & \cdots & W^{(N-1)(P+r-1)} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_{N-1} \end{bmatrix}$$

From the whole set of data one can form $N/r$ subsets, each containing $r$ terms:

$$\begin{array}{l} (x_0, x_{N/r}, x_{2N/r}, \quad , \quad x_{(r-1)N/r}) \\ (x_1, x_{(N/r)+1}, \quad , \quad x_{(r-1)N/r+1}) \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ (x_{(N/r)-1}, x_{(2N/r)-1}, \quad , \quad x_{N-1}) \end{array}$$

Assume $D_r$ is the diagonal matrix of dimension $r$, whose elements are the powers of $W$, $W^k$ with $0 \leqslant k \leqslant r-1$.

The matrix of the partial transform can be separated into $N/r$ submatrices which can each be applied to one of the sets which were defined earlier and the matrix equation of the transform is written:

$$[X]_{p,r} = \sum_{i=0}^{(N/r)-1} D_r^i T_r (W^P)^i D_r^{(N/r)}[x]_{i,r}$$

where $[X]_{p,r}$ denotes the set of $r$ numbers $X_k$ with $p \leqslant k \leqslant p+r-1$ and $[x]_{i,r}$ the set of data $x_k$ with $k = nN/r + i$ and $n = 0, 1, \ldots, r-1$. The transform $T_r$ is that of order $r$.

Consequently, if $r$ is a factor of $N$, a partial transform relating to $r$ points is calculated using $N/r$ transforms of order $r$ with which the appropriate diagonal matrices are associated.

If $N$ and $r$ are powers of 2, the number $M_p$ of complex multiplications to be made is given by

$$M_P=\frac{N}{r}\left(\frac{r}{2}\log_2\frac{r}{2}+2r\right)=N\left[\tfrac{1}{2}\log_2\left(\frac{r}{2}\right)+2\right] \qquad (3.22)$$

This result is equally valid when it is the number of points to be transformed which is limited, as is often the case in spectrum analysis. A common example of a partial transform is that applied to real data.

### 3.5.1 Transform of real data and odd DFT

If the data to be transformed are real the properties listed in Chapter 2 show that the transformed numbers $X(k)$ and $X(N-k)$ are complex conjugates, that is, $X(k)=X(N-k)$. Thus, it is only necessary to calculate the set of the $X_k$ with $0 \leqslant k \leqslant N/2-1$ and the above result can be applied:

$$[X]_{0,N/2}=T_{N/2}[x]_{0,N/2}+D_{N/2}T_{N/2}[x]_{1,N/2}$$

In this particular case, the transform $T_{N/2}$ has only to be calculated once taking advantage of the following property of the discrete Fourier transform: if the set to be transformed $x_k$ is purely imaginary, and the transformed set is such that:

$$X(k)=-\bar{X}(N-k)$$

Under these conditions the procedure for calculating the transform of a real set is as follows:

(1) Using the $x(k)$, form a complex set of $N/2$ terms $y(k)=x(2k)+jx(2k+1)$ with $0 \leqslant k \leqslant N/2-1$.
(2) Calculate the transform $Y(k)$ of the set $y(k)$ with $0 \leqslant k \leqslant N/2-1$.
(3) Calculate the required numbers using the expression

$$X(k)=\frac{1}{2}\left[Y(k)+\bar{Y}\left(\frac{N}{2}-k\right)\right]+\tfrac{1}{2}je^{-j2\pi(k/N)}\left[\bar{Y}\left(\frac{N}{2}-k\right)-Y(k)\right]$$

with $0 \leqslant k \leqslant N/2-1$.

If $N$ is a power of 2, the number of complex multiplications $Mc$ to be made is

$$Mc=\frac{N}{4}\log_2\left(\frac{N}{4}\right)+\frac{N}{2}=\frac{N}{4}\log_2 N \qquad (3.23)$$

Memory locations are required for $N$ real numbers. An algorithm for real data is described in detail in Ref. [4]. Another method of calculating the transforms of real numbers is to use odd transforms [5].

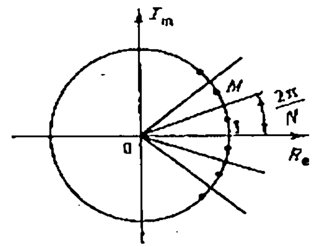


*Fig. 3.5 Coefficients of the odd discrete Fourier transform*

The odd discrete Fourier transform establishes by definition the following relations between two sets of $N$ complex numbers $x(n)$ and $X(k)$:

$$X(k)=\frac{1}{N}\sum_{n=0}^{N-1}x(n)e^{-j2\pi(2k+1)n/(2N)} \qquad (3.24)$$

$$x(n)=\sum_{k=0}^{N-1}X(k)e^{j2\pi(2k+1)n/(2N)} \qquad (3.25)$$

The coefficients of this transform have as their co-ordinates the points M of a unit circle such that the vector OM makes an angle with the abscissa which is an odd multiple of $2\pi/2N$, as shown in Figure 3.5.

By setting $W=e^{-j(\pi/N)}$ the matrix of this transform is written:

$$T_N^{\prime}=\begin{bmatrix} 1 & W & W^2 & \cdots & W^{N-1} \\ 1 & W^3 & W^6 & \cdots & W^{3(N-1)} \\ 1 & W^5 & W^{10} & \cdots & W^{5(N-1)} \\ \vdots & \vdots & & & \vdots \\ 1 & W^{(2N-1)} & \cdots & \cdots & W^{(2N-1)(N-1)} \end{bmatrix}$$

If the $x(n)$ are real numbers, one can write

$$X(N-1-k)=\frac{1}{N}\sum_{n=0}^{N-1}x(n)e^{-j2\pi[2(N-1-k)+1]n/(2N)}$$

$$=\frac{1}{N}\sum_{n=0}^{N-1}x(n)e^{j2\pi(2k+1)n/(2N)}$$

Thus,

$$X(N-1-k)=\overline{X(k)}$$

Consequently, since the $X(k)$ with even and odd indices are complex conjugates it is sufficient to calculate the $X(k)$ with even index in order to perform a transform on real numbers. Such a transform is the matrix $T_R$ given by

$$T_R = \begin{bmatrix} 1 & W & W^2 & \cdots & W^{N/2} & \cdots & W^{N-1} \\ 1 & W^5 & W^{10} & \cdots & W^{5N/2} & \cdots & W^{5(N-1)} \\ \vdots & \vdots & & & \vdots & & \vdots \\ 1 & W^{2N-3} & \cdots & \cdots & W^{(2N-3)N/2} & \cdots & W^{(2N-3)(N-1)} \end{bmatrix}$$

Let $D_{N/2}$ be the diagonal matrix whose elements are $W^k$ with $0 \leqslant k \leqslant N/2 - 1$, and let $T_{N/2}$ be the matrix of the transform of order $N/2$. Allowing for the fact that $W^{2N} = 1$ and $W^{N/2} = -\mathrm{j}$, this becomes:

$$T_R = [T_{N/2}D, -\mathrm{j}T_{N/2}D] = (T_{N/2}D) \times [1, -\mathrm{j}]$$

The odd transform of the real data is then calculated by carrying out a transform of order $N/2$ on the set of complex numbers:

$$y(n) = \left[x(n) - \mathrm{j}x\left(\frac{N}{2} + n\right)\right]W^n \quad \text{with} \quad 0 \leqslant n \leqslant \frac{N}{2} - 1$$

The number of calculations is the same as in the method illustrated at the beginning of this section, but the structure is simpler. It should be noted that the transformed numbers give a frequency sampling of the signal spectrum represented by the $x(n)$, displaced by a half-step on the frequency axis.

An important case where significant simplifications are introduced is that of real symmetrical sets. Reductions in the calculations are illustrated by using the doubly odd transform [6].

### 3.5.2 The odd-time odd-frequency DFT

The odd-time odd-frequency discrete Fourier transform establishes by definition the following relations between two sets of $N$ complex numbers $x(n)$ and $X(k)$:

$$X(k) = \frac{1}{N}\sum_{n=0}^{N-1} x(n)\mathrm{e}^{-\mathrm{j}2\pi(2k+1)(2n+1)/(4N)} \tag{3.26}$$

$$x(n) = \sum_{k=0}^{N-1} X(k)\mathrm{e}^{\mathrm{j}2\pi(2k+1)(2n+1)/(4N)} \tag{3.27}$$

The coefficients of this transform are based on the points M of a unit circle such that the vector $\overrightarrow{OM}$ forms an angle with the abscissa which is an odd multiple of $2\pi/4N$ as shown in Figure 3.6.

If the $x(n)$ are real numbers, expression (3.26) leads to:

$$X(N - 1 - k) = -\bar{X}(k)$$

Similarly, if the $X(k)$ are real numbers, then:

$$x(N - 1 - n) = -\bar{x}(n)$$

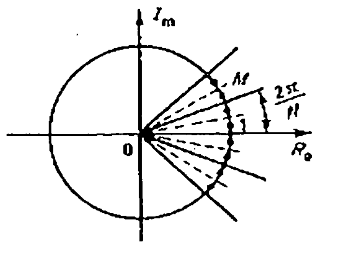


*Fig. 3.6 Coefficients of the doubly odd discrete Fourier transform*

By assuming as before that $W = \mathrm{e}^{-\mathrm{j}(\pi/N)}$, the matrix of the transform is written as:

$$T_N^{\mathrm{II}} = \begin{bmatrix} W^{1/2} & W^{3/2} & W^{5/2} & \cdots & W^{N-1/2} \\ W^{3/2} & W^{9/2} & W^{15/2} & \cdots & W^{3(N-1/2)} \\ W^{5/2} & W^{15/2} & W^{25/2} & \cdots & W^{5(N-1/2)} \\ \vdots & \vdots & \vdots & & \vdots \\ W^{N-1/2} & W^{3(N-1/2)} & \cdots & \cdots & W^{(2N-1)(N-1/2)} \end{bmatrix}$$

This transform is factorized as follows:

$$T_N^{\mathrm{II}} = W^{1/2}\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & W & 0 & \cdots & 0 \\ \vdots & \vdots & & & \vdots \\ 0 & 0 & & \cdots & W^{N-1} \end{bmatrix}\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & W^2 & W^4 & \cdots & W^{2(N-1)} \\ 1 & W^4 & W^8 & \cdots & W^{4(N-1)} \\ \vdots & \vdots & & & \vdots \\ 1 & W^{2(N-1)} & & \cdots & W^{2(N-1)(N-1)} \end{bmatrix}$$

$$\times\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 1 & W & 0 & \cdots & 0 \\ \vdots & \vdots & & & \vdots \\ 0 & 0 & & \cdots & W^{N-1} \end{bmatrix}$$

That is,

$$T_N^{\mathrm{II}} = W^{1/2}D_N T_N D_N$$

Let us consider the case where the set of data $x(n)$ is real and antisymmetric, i.e. $x(n) = -x(N - 1 - n)$. Then the same applies to the set $X(k)$. The set of the $x(n)$ for even $n$ is equal to the set of the $x(n)$ for odd $n$, except for the sign. The situation is the same for the set of $X(k)$.

In order to calculate the transform it is sufficient in this case to carry out the calculations for the $x(2n)$ with $0 \leqslant n \leqslant N/2 - 1$ since the $X(k)$ are real numbers. Alternatively, it is sufficient to make the calculations on the $X(2k)$ with $0 \leqslant k \leqslant N/2 - 1$.

The corresponding matrix $T_{RR}$ is written:

$$T_{RR} = W^{1/2}\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & W^2 & 0 & \cdots & 0 \\ \vdots & \vdots & & & \vdots \\ 0 & 0 & \cdots & \cdots & W^{2(N/2-1)} \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 & \cdots & 1 & 1 \\ 1 & W^2 & \cdots & W^{9[(N/2)-1]} & 0 \\ 1 & W^{16} & \cdots & \vdots & \vdots \\ \vdots & \vdots & & \vdots & \vdots \\ 1 & W^{8[(N/2)-1]} & W^{8[(N/2)-1](N/2)-1]} & & 0 \end{bmatrix}\begin{bmatrix} 0 & \cdots & 0 \\ W^2 & \cdots & 0 \\ \vdots & & \vdots \\ 0 & & W^{2[(N/2)-1]} \end{bmatrix}$$

Allowing for $W^{2N} = 1$, this becomes:

$$T_{RR} = W^{1/2}D_{N/2}\begin{bmatrix} T_{N/4} & T_{N/4} \\ T_{N/4} & T_{N/4} \end{bmatrix}D_{N/2}$$

and, as $W^{2N/4} = -j$, this calculation can be made with one performance of the operations represented by the matrix $T_{N/4}$ on the set of numbers $x(2n) - jx(2n + N/2)$ with $0 \leqslant n \leqslant N/4 - 1$. The $N/4$ numbers obtained are complex ones whose real parts form the set of the desired $X(2k)$ with $0 \leqslant k \leqslant N/4 - 1$. By carrying out the operation defined by $T_{RR}$ for the transformed numbers of rank $2k + N/2$ with $0 \leqslant k \leqslant N/4 - 1$ it can be verified that we have obtained the earlier numbers multiplied by $-j$. That is, the imaginary part of the numbers obtained previously furnishes the set of the $X(2k + N/2)$. It follows that, if the doubly odd transform is applied to a real and antisymmetric set of $N$ terms or to a symmetric set which becomes antisymmetric through a suitable change of sign, it is reduced to the equation

$$\left[X(2k) + jX\left(2k + \frac{N}{2}\right)\right] = W^{1/2}D_{N/4}T_{N/4}D_{N/4}\left[x(2n) - jx\left(2n + \frac{N}{2}\right)\right]$$

with $0 \leqslant k \leqslant N/4 - 1$, $0 \leqslant n \leqslant N/4 - 1$, and where $D_{N/4}$ is a diagonal matrix whose elements are $W^{2i}$ with $0 \leqslant i \leqslant N/4 - 1$.

The number of complex multiplications $Mc$ which are necessary is

$$Mc = \frac{N}{8}\log_2\left(\frac{N}{8}\right) + 2\frac{N}{4} = \frac{N}{8}\log_2(2N) \tag{3.28}$$

Comparisons using the different transforms are given in Table 3.1 to illustrate the amount of calculation for each type.

The importance of the odd transforms can be easily seen. It should, however, be noted that other algorithms allow greater reductions to be made for real data and for symmetric real data [7], but these are not as simple to use, especially for practical implementations.

One feature of the doubly odd transform when applied to a real antisymmetric set is that it is identical to the inverse transform. Apart from the scale factor $1/N$, there is no distinction, in this case, between the direct and the inverse transforms.

Table 3.1. ARITHMETIC COMPLEXITIES OF THE VARIOUS FAST FOURIER TRANSFORMS

| | Complex multiplications | Complex additions | Memory positions |
|---|---|---|---|
| Complex DFT | $\frac{N}{2}\log_2\left(\frac{N}{2}\right)$ | $N\log_2 N$ | $2N$ |
| Odd DFT—real data | $\frac{N}{4}\log_2(N)$ | $\frac{N}{2}\log_2\left(\frac{N}{2}\right)$ | $N$ |
| Doubly odd DFT—symmetrical real data | $\frac{N}{8}\log_2(2N)$ | $\frac{N}{4}\log_2\left(\frac{N}{4}\right)$ | $\frac{N}{2}$ |

The Fourier transform of a real symmetric set is introduced, for example, when deriving the power spectrum density of a signal from its autocorrelation function.

### 3.5.3 Sine and cosine transforms

The transforms considered so far have complex coefficients. Discrete transforms of the same family can be obtained using the real and imaginary parts of the complex coefficients [8].

The following transforms can be defined:

(1) The cosine DFT (cos-DFT):

$$X_{CF}(k) = \frac{1}{N}\sum_{n=0}^{N-1} x(n)\cos\frac{2\pi nk}{N} \tag{3.29}$$

(2) The sine DFT (sin-DFT):

$$X_{SF}(k) = \frac{1}{N}\sum_{n=0}^{N-1} x(n)\sin\frac{2\pi nk}{N} \tag{3.30}$$

(3) The Discrete cosine Transform (DCT):

$$X_{DC}(0) = \frac{\sqrt{2}}{N}\sum_{n=0}^{N-1} x(n)$$

$$X_{DC}(k) = \frac{2}{N}\sum_{n=0}^{N-1} x(n)\cos\left(\frac{2\pi(2n+1)k}{4N}\right). \tag{3.31}$$

The inverse transform is given by:

$$x(n) = \frac{1}{\sqrt{2}}X_{DC}(0) + \sum_{n=1}^{N-1} X_{DC}(k)\cos\frac{2\pi(2n+1)k}{4N}$$

(4) The Discrete sine Transform (DST):

$$X_{DS}(k) = \sqrt{\left(\frac{2}{N+1}\right)} \sum_{n=0}^{N-1} x(n) \sin\left[\frac{2\pi(n+1)(k+1)}{2N+2}\right] \tag{3.32}$$

Through algebraic manipulations, as in the previous sections, it is possible to establish relationships between the standard DFT and these transforms as well as among these transforms themselves.

For example, from the definitions we have:

$$DFT(N) = \cos\text{-}DFT(N) - j\sin\text{-}DFT(N)$$

Now, considering the cosine DFT:

$$X_{CF}(k) = \sum_{n=0}^{N/2-1} x(2n)\cos\frac{2\pi nk}{N/2}$$

$$+ \sum_{n=0}^{N/4-1} [x(2n+1) + x(N-2n-1)]\cos\left[\frac{2\pi(2n+1)k}{4N/4}\right] \tag{3.33}$$

it is clear that the cosine DFT of order $N$ can be completed with the help of a cosine $DFT$ of order $N/2$ and a DCT of order $N/4$; in concise form:

$$\cos DFT(N) = \cos DFT(N/2) + DCT(N/4)$$

Similarly, the DCT is expressed by:

$$X_{DC}(k) = \frac{2}{N}\sum_{n=0}^{N/2-1}\left[x(2n)\cos\frac{2\pi(4n+1)k}{4N} + x(2n+1)\cos\frac{2\pi[4(N-n-1)+1]k}{4N}\right]$$

and now, taking

$$y(n) = x(2n); \quad 0 \leqslant n \leqslant \frac{N}{2} - 1$$

$$y(N-n-1) = x(2n+1)$$

we have:

$$X_{DC}(k) = \frac{2}{N}\sum_{n=0}^{N-1} y(n)\cos\frac{2\pi(4n+1)k}{4N}$$

Expanding the cosine function yields, in concise form:

$$DCT(N) = \cos\frac{2\pi k}{4N}\cos DFT(N) - \sin\frac{2\pi k}{4N}\sin\text{-}DFT(N) \tag{3.34}$$

Finally, the DCT of order $N$ can be completed with the help of a DFT of the same order.

Among the transforms based on real coefficients, the Discrete Hartley transform (DHT) is worth mentioning. It is defined by [9]:

$$X_{DH}(k) = \frac{1}{N}\sum_{n=0}^{N-1} x(n)\left[\cos 2\pi\frac{nk}{N} + \sin 2\pi\frac{nk}{N}\right] \tag{3.35}$$

and the inverse transform:

$$x(n) = \sum_{k=0}^{N-1} X_{DH}(k)\left[\cos 2\pi\frac{nk}{N} + \sin 2\pi\frac{nk}{N}\right]$$

The connection with the DFT is given by:

$$X(k) = \tfrac{1}{2}[X_{DH}(k) + X_{DH}(N-1-k) - j(X_{DH}(k) - X_{DH}(N-1-k))] \tag{3.36}$$

The discrete sine and cosine transforms have been introduced for information compression, particularly in image processing. It is worth pointing out that, for images, they provide reasonably accurate approximations of the eigen-transform, which yields a signal representation with the minimum number of parameters.

### 3.5.4 The two-dimensional DCT

For a set of $(N \times N)$ real data the two-dimensional DCT (2D-DCT) is defined by:

$$X(k_1,k_2) = \frac{4e(k_1)e(k_2)}{N^2}\sum_{n_1=0}^{N-1}\sum_{n_2=0}^{N-1} x(n_1,n_2)\cos\frac{2\pi(2n_1+1)k_1}{4N}\cos\frac{2\pi(2n_2+1)k_2}{4N} \tag{3.37}$$

and:

$$x(n_1,n_2) = \sum_{k_1=0}^{N-1}\sum_{k_2=0}^{N-1} e(k_1)e(k_2)X(k_1,k_2)\cos\frac{2\pi(2n_1+1)k_1}{4N}\cos\frac{2\pi(2n_2+1)k_2}{4N}$$

with:

$$e(k) = \frac{1}{\sqrt{2}}; \quad k = 0$$

$$e(k) = 1; \quad k \neq 0$$

That transform is separable and it can be computed as follows:

$$X(k_1,k_2) = \frac{2}{N}e(k_2)\sum_{n_2=0}^{N-1}\cos\frac{2\pi(2n_2+1)k_2}{4N}$$

$$\times\left[\frac{2}{N}e(k_1)\sum_{n_1=0}^{N-1} x(n_1,n_2)\cos\frac{2\pi(2n_1+1)k_1}{4N}\right] \tag{3.38}$$

Thus, the 2D-transform can be computed using $2N$ times the 1D-DCT and the number of real multiplications is of the order of $N^2\log_2(N)$. In fact, that amount can be reached, with the help of an algorithm based on the decomposition of a DCT of order $N$ into two DCT of order $N/2$ [8]. It is even possible to reach the value $3/4\,N^2\log_2 N$, through extension of the decimation technique and splitting the set of $(N \times N)$ data into subsets of $(N/2 \times N/2)$ data [10].

## 3.6 OTHER FAST ALGORITHMS

Fast Fourier transform algorithms form a technique for calculating a discrete Fourier transform of order $N$ using a number of multiplications of the order of $N \log_2 N$. It has been shown in the previous sections that these algorithms have a relatively simple structure and offer sufficient flexibility for good adaptation to the operating constraints and to technological characteristics to be achieved. This explains why they are of great interest for practical applications.

Nevertheless, they are not the only method of fast calculation of a DFT, and algorithms can be elaborated which involve, at least in certain cases, a smaller calculation time or a lower number of multiplications, or which are applicable for values of the order $N$ which are not necessarily powers of two.

A first approach consists of replacing the complex multiplications, which are costly in circuitry or time, by a set of operations which are simpler to put into operation. Reference [11] describes a technique which uses one property of the DFT mentioned in Section 2.4.1, namely the fact that multiplications by the coefficients $W^k$ correspond to phase rotations.

One method which is particularly interesting is used to obtain a multiplication volume of order $N$, instead of $N \log_2 N$, for a DFT of order $N$. This method depends on factorizing the matrix $T_N$ in a particular way. It is decomposed into a product of three factors:

$$T_N = B_N C_N A_N$$

where $A_N$ is a $J$ by $N$ matrix, $J$ is a whole number, $C_N$ is a diagonal matrix of dimension $J$ and $B_N$ is an $N$ by $J$ matrix. The special feature of this factorization is that the elements of the matrices $A_N$ and $B_N$ are 0, 1, or $-1$. Under these conditions, the calculation requires only $J$ multiplications. This decomposition is obvious for $J = N^2$; for example for $N = 3$, we obtain:

$$T_3 = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

$$\times \begin{bmatrix} 1 & & & & & & & & \\ & 1 & & & & & & O & \\ & & 1 & & & & & & \\ & & & 1 & & & & & \\ & & & & W & & & & \\ & & & & & W^2 & & & \\ & & & & & & 1 & & \\ & O & & & & & & W^2 & \\ & & & & & & & & W \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

With some low values of $N$, certain factorizations are available in which $J$ is

of the order of $N$, in which case there are the same number of multiplications. In order to generalize this property and to illustrate a suitable factorization of $T_N$ it is necessary to perform a permutation of the data before and after transformation. For example, for $N = 12$, by assuming

$$X' = \begin{bmatrix} X_0 \\ X_3 \\ X_6 \\ X_9 \\ X_4 \\ X_7 \\ X_{10} \\ X_1 \\ X_8 \\ X_{11} \\ X_2 \\ X_5 \end{bmatrix} \quad \text{and} \quad x' = \begin{bmatrix} x_0 \\ x_9 \\ x_6 \\ x_3 \\ x_4 \\ x_1 \\ x_{10} \\ x_7 \\ x_8 \\ x_5 \\ x_2 \\ x_{11} \end{bmatrix}$$

and by using the Kronecker products of the matrices, it can be shown that:

$$X' = (T_3 \times T_4)x'$$

Similarly, if $N$ has $L$ factors such that

$$N = N_L N_{L-1} \cdots N_1$$

It can be shown that:

$$X' = (T_{N_L} \times T_{N_{L-1}} \times \cdots T_{N_1})x' \tag{3.39}$$

By using the factorization defined earlier for the matrices $T_{N_L}$ and the algebraic properties of the Kronecker products, this becomes

$$X' = (B_{N_L} \times B_{N_{L-1}} \times \cdots \times B_{N_1}(C_{N_L} \times C_{N_{L-1}} \times \cdots \times C_{N_1})$$
$$\times (A_{N_L} \times A_{N_{L-1}} \times \cdots \times A_{N_1})x'$$

This result defines a types of algorithm called the Winograd algorithm.

It can be clearly seen that the algorithm of order $N$ is deduced from algorithms of order $N_i$ with $1 \leqslant i \leqslant L$. Herein lies the importance of algorithms with small numbers of multiplications for small values of $N$. Reference [12] gives algorithms for $N = 2, 3, 4, 5, 7, 8, 9$, and 16, where the number of multiplications is of the order of $N$, as shown in Table 3.2. In the multiplication column, the figures in parentheses give the number of multiplications by coefficients different from 1. Further, these are complex multiplications which correspond to two real multiplications. The number of additions is comparable to that for FFT algorithms.

*Table 3.2.* ARITHMETIC COMPLEXITIES OF LOW ORDER WINOGRAD ALGORITHMS

| Order of the DFT | Multiplications | Additions |
|---|---|---|
| 2 | 2 (0) | 2 |
| 3 | 3 (2) | 6 |
| 4 | 4 (0) | 8 |
| 5 | 6 (5) | 17 |
| 7 | 9 (8) | 36 |
| 8 | 8 (2) | 26 |
| 9 | 11 (10) | 44 |
| 16 | 18 (10) | 74 |

The algorithms for low values of $N$ are obtained by calculating the Fourier transform as a set of correlations:

$$X_k = \sum_{n=1}^{N-1} (x_n - x_0)W^{nk}; \quad k = 1, \dots, N-1$$

and by using the algebraic properties of the set of exponents of $W$ which are defined modulo $N$.

For example, for $N = 4$, the sequence of operations is as follows:

$$t_1 = x_0 + x_2, \qquad t_2 = x_1 + x_3$$
$$m_0 = 1(t_1 + t_2), \quad m_1 = 1(t_1 - t_2)$$
$$m_2 = 1(x_0 - x_2), \quad m_3 = j(x_1 - x_3)$$
$$X_0 = m_0$$
$$X_1 = m_2 + m_3$$
$$X_2 = m_1$$
$$X_3 = m_2 - m_3$$

For $N = 8$:

$$t_1 = x_0 + x_4, \quad t_2 = x_2 + x_6, \quad t_3 = x_1 + x_5,$$
$$t_4 = x_1 - x_5, \quad t_5 = x_3 + x_7, \quad t_6 = x_3 - x_7,$$
$$t_7 = t_1 + t_2, \quad t_8 = t_3 + t_5,$$
$$m_0 = 1(t_7 + t_8), \qquad m_1 = 1(t_7 - t_8),$$
$$m_2 = 1(t_1 - t_2), \qquad m_3 = 1(x_0 - x_4),$$
$$m_4 = \cos\left(\frac{\pi}{4}\right)(t_4 - t_6), \quad m_5 = j(t_3 - t_5),$$
$$m_6 = j(x_2 - x_6), \qquad m_7 = j\sin\left(\frac{\pi}{4}\right)(t_4 + t_6),$$

$$s_1 = m_3 + m_4, \quad s_2 = m_3 - m_4, \quad s_3 = m_6 + m_7, \quad s_4 = m_6 - m_7$$
$$X_0 = m_0, \quad X_1 = s_1 + s_3, \quad X_2 = m_2 + m_5, \quad X_3 = s_2 - s_4,$$
$$X_4 = m_1, \quad X_5 = s_2 + s_4, \quad X_6 = m_2 - m_5, \quad X_7 = s_1 - s_3.$$

Finally, Winograd algorithms generally introduce a reduction in the amount of computation. This can be large when compared to FFT algorithms. The situation is similar for other algorithms, such as those using the polynomial transforms [13].

These techniques have a large application potential and are of considerable importance in certain cases. However, it should be noted that they may require a larger memory capacity and a more complicated sequence of operations, which results in an increase in the size of the system's control unit or in the volume of the program memory.

Another attractive path towards the optimization of processing and machines is to use transforms operating in finite fields, called number theoretic transforms.

## 3.7 NUMBER-THEORETIC TRANSFORMS

Fourier transformation involves making arithmetic operations on the field of complex numbers. The machines which carry out these operations generally use binary representations which are approximations to the data and the coefficients. The precision of the calculations is a function of the number of bits available in the machine.

In practice, a machine with $B$ bits carries out its operations on the set with $2^B$ integers: $0, 1, \dots, 2^B - 1$. In this set the usual laws of addition and multiplication cannot be applied, as shifting and truncation must be introduced, which lead to approximations in the calculations, as was shown in Section 2.3.

The first condition to be fulfilled to ensure exact calculations in a set $E$ is that the product or the sum of two elements of the set $E$ also belongs to this set. This condition is satisfied in the set of integers $0, 1, \dots, M-1$ if the calculations are made modulo $M$. By appropriate selection of the modulus $M$ it is possible to define transformations with properties which are comparable to those of the DFT and which allow error-free calculation of convolutions with fast calculation algorithms.

The definition of such transformations rests on the algebraic properties of integers modulo $M$, for certain values of $M$. They are called number-theoretic transforms.

The choice of the modulus $M$ is governed by the following considerations:

(1) Simplicity of the calculations in the modular arithmetic. In principle, modular arithmetic implies a division by the modulus $M$. This division is trivial for $M = 2^m$. It is very simple for $M = 2^m \pm 1$, because the result is obtained by adding a carry-bit (1s complement arithmetic) or subtracting it.

(2) The modulus must be sufficiently large. The result of the convolution must be capable of representation without ambiguity in the modulo $M$ arithmetic. For example, a convolution with 32 terms with 12-bit data and 8-bit coefficients requires $M > 2^{25}$.

(3) Suitable algebraic properties. The set of modulo $M$ integers should have algebraic properties allowing the definition of transformations comparable to the DFT.

First, there should be periodic elements in order that the fast algorithms can be elaborated; it must have an element $\alpha$ such that:

$$\alpha^N = 1$$

A transformation can then be defined by the expression:

$$X(k) = \sum_{n=0}^{N-1} x(n)\alpha^{nk} \qquad (3.40)$$

For the existence of the inverse transformation which is defined by the expression:

$$x(n) = N^{-1} \sum_{k=0}^{N-1} X(k)\alpha^{-nk} \qquad (3.41)$$

it is first necessary for $N$ and the powers of $\alpha$ to have inverses.

It can be shown that $N$ has an inverse modulo $M$ if $N$ and $M$ are prime relative to each other. The element $\alpha$ should be prime relative to the $M$ and of order $N$, i.e. $\alpha^N = 1$.

The existence of the inverse transformation implies a further condition for the $\alpha^i$, that

$$\sum_{k=0}^{N-1} \alpha^{ik} = N\delta(i) \quad \text{with} \quad \begin{array}{l} \delta(i) = 1 \text{ if } i = 0 \text{ modulo } N \\ \delta(i) = 0 \text{ if } i \neq 0 \text{ modulo } N \end{array}$$

This condition reflects the fact that each element $(1 - \alpha^i)$ must have an inverse. It can be shown that all of the conditions for the existence of a transformation and its inverse reduce to the following one. For each prime factor $P$ of $M, N$ must be a factor of $P - 1$. Thus, if $M$ is prime, $N$ must divide $M - 1$.

Fast algorithms can be elaborated if $N$ is a composite number, in particular, if $N$ is a power of 2. These algorithms are similar to those of the FFT:

$$N = 2^m$$

The calculations to be performed in the transformation are considerably simplified in the particular case when $\alpha = 2$.

Finally, an interesting choice for the modulus $M$ is

$$M = 2^{2^m} + 1$$

where $M$ is prime. These are the Fermat numbers.

A transform based on the Fermat numbers is defined as follows:

(1) Modulus $M = 2^{2^m} + 1$;

(2) Order of the transform: $N = 2^{m+1}$;

(3) Direct transform:

$$X(k) \doteq \sum_{n=0}^{N-1} x(n)2^{nk}$$

(4) Inverse transform:

$$x(n) = (2^t) \sum_{k=0}^{N-1} X(k)2^{-nk} \quad \text{with} \quad t = 2^{m+1} - m - 1$$

*Example*: $m = 3$, $2^m = 8$, $M = 257$, $N = 16$, $t = 12$.

This transform allows the calculation of convolutions of real numbers, as with the discrete Fourier transform, but with the following advantages.

(a) The result is obtained without approximation;

(b) The operations relate to real numbers;

(c) The calculation of the transform and its inverse does not require any multiplication. The only multiplications which remain are in the transformed space.

This technique does, nevertheless, have some significant limitations. As the calculations are exact, the modulus $M$ should be sufficiently large, resulting in numbers which are long.

The relations between the parameters $M$ and $N$ given above require that the calculations are carried out with a number of bits $B$ of the order of $N/2$. That is, the number of terms in the convolution is approximately twice the number of bits of data in the calculation. The application is consequently restricted to convolutions which contain a small number of terms.

The field for the application of number theoretic transforms can be widened by employing numbers other than the Fermat numbers, or by treating the convolutions with many terms as two dimensional convolutions [14]. Reference [15] describes a practical example.

## REFERENCES

[1] M. C. Pease, *Methods of Matrix Algebra*, Academic Press, New York, 1965.

[2] M. Kunt, Generalized fast transforms. *AGEN*, Zurich, No. 13, June 1972, pp. 1–49.

[3] H. Sloate, Matrix representations for sorting and the fast Fourier transform. *IEEE Trans.*, CAS-21(1), 1974.

[4] G. Bergland, A fast Fourier transform algorithm for real valued series. *Comm. ACM*, 11(10), October 1968.

[5] J. L. Vernet, Real signals FFT by means of an odd discrete Fourier transform. *Proc. IEEE*, October 1971.

[6] G. Bonnerot and M. Bellanger, Odd-time odd-frequency DFT for symmetric real-valued series. *Proc. IEEE*, March 1976.

[7] H. V. Sorensen, D. L. Jones, M. T. Heideman and C. S. Burrus, Real valued Fast Fourier Transform algorithms. *IEEE Transactions*, ASSP-35, No. 6, pp. 849–63, June 1987.

[8] M. Vetterli and H. Nussbaumer, Simple FFT and DCT algorithms with reduced number of operations. *Signal Processing*, 6, No. 4, August, 267–78, 1984.

[9] R. N. Bracewell, The fast Hartley transform. *Proc. IEEE*, 22, August 1984, 1010–18.

[10] M. A. Haque, A two-dimensional fast cosine transform. *IEEE Trans.*, ASSP-33, No. 6, December 1532–9, 1985.

[11] A. Despain, Very fast Fourier transform algorithm hardware for implementation. *IEEE Transactions on Computers*, C28(5), May 1979.

[12] H. Silverman, An introduction to programming the Winograd Fourier transform algorithm. *IEEE Trans.*, ASSP-25(2), 1977.

[13] H. Nussbaumer, *Fast Fourier Transform and Convolution Algorithms*, Springer Verlag, Berlin, 1981.

[14] R. C. Agrawal and C. S. Burrus, Number theoretic transforms to implement fast digital convolution. *Proc. IEEE*, 63, 1975.

[15] J. H. MacClellan, Hardware realization of a Fermat number transform. *IEEE Trans.*, ASSP-24, 1976.

## EXERCISES

1 Find the Kronecker product $A \times I_3$ for the matrix $A$ such that:

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$$

and the unit matrix $I_3$ of dimension 3.

Find the product $I_3 \times A$ and compare it with the above.

2 By taking square matrices of dimension 2, verify the properties of the Kronecker products given in Section 3.1.

3 Give the factorizations of the matrix of a DFT of order 64 on base 2, base 4, and base 8, following the procedure given in Section 3.2.

Calculate the required number of multiplications for each of these three cases and compare the results.

4 Using the decimation-in-time approach, factorize the matrix of the DFT of order 12. What is the minimum number of multiplication and additions required? Write a computer program for the calculation.

5 Factorize the matrix of a DFT of order 16 on base 2 for the following two cases:

(a) When the data appear at both input and output in natural order,
(b) When the stages in the calculation are identical.

For the latter case, devise an implementation scheme involving the use of shift registers as memories.

6 Calculate a discrete Fourier transform of order 16 relating to real data.
Give the algorithm which uses a DFT of order 8 for this calculation.
Give the algorithm based on the odd DFT.
Compare these algorithms and the numbers of operations.

7 Calculate a DFT of order 12 by using a factorization of the type given in Section 3.6 and with the given permutations for the data.

Evaluate the number of operations and compare it with the values found in Exercise 4.

8 To perform a circular convolution of the two sets $x = (2, -2, 1, 0)$ and $h = (1, 2, 0, 0)$ a number-theoretic transform of modulus $M = 17$ and coefficient $\alpha = 4$ is used.

As $N = 4$, verify that $\alpha^N = 1$. Give the matrices of the transformation and the inverse transformation. Prove that the desired result is the set $y = (2, 2, -3, 2)$.

A 12

t.c. electronic
ULTIMATE SOUND MACHINES

# Personal testimony regarding EP 0 649 578

In the time from January 1st 1988 to March 31st 1997 I was employed at Bang & Olufsen, Struer, Denmark as a Research Engineer.

One day in the beginning of this period we had a visit from Peter Single, then Technical Director of Austek Microsystems Pty. Inc. I still have his business card.

Mr. Single demonstrated a Fast Convolution box based on the newly developed Austek A41102 FDP Frequency Domain Processor Chip. The box was capable of real-time FIR filtering of digital audio from a CD player (44.1 kHz samplingrate) with very long impulse responses, 32K (i.e. 32768) taps or more.

I think he had visited Aalborg University in the morning and came to see us after lunch. Associate Professor Sofus B. Nielsen at Aalborg University has confirmed having received a visitor from Austek giving the same demonstration. This was in 1989, as the visitor - probably Mr. Single - was on his way back from the 87th AES Convention in New York, where a paper (AES Preprint 2830) by Peter Single and David McGrath was indeed presented. The paper's title was "*Implementation of a 32768-Tap FIR Filter Using Real-Time Fast Convolution*". I seem to remember that we had a laugh at B&O about Australia being so far away, that Denmark was "on the way home" from New York.

Present at the meeting at B&O (probably in October 1989) were my colleagues Erik Sørensen and Henrik Fløe Mikkelsen. Today Henrik is Senior DSP Technology Manager at B&O and Erik is IT Manager at Struer Erhvervsskole (Technical School) in Struer, Denmark.

The Austek demonstration went well, and we were all duly impressed, although the filtering algorithm suffered from a substantial latency (input-output delay) of more than a second. We couldn't help thinking about how to reduce this latency problem.

As the meeting was drawing to an end (I think we had actually left the meeting room, waiting for a taxi to come and pick up Mr. Single), Erik Sørensen had got an idea: If you start a number of different-sized fast convolution filters in parallel and add up their outputs, the small ones can reproduce the start of the composite impulse response with low latency giving the larger ones time to produce their later parts of the composite impulse response. **I distinctly remember Erik explaining the idea to Mr. Single.** The meeting was not under NDA, it was a normal product demonstration visit (I believe the convolution box was sold for USD 10,000).

Austek became Lake DSP which became Lake Technology Ltd. all with David McGrath as a primary mover on the technology side. Mr. McGrath filed the patent application EP 0 649 578 for the Low-Latency Fast Convolution Algorithm in 1993. **I have often been thinking - with regret - that it is really Erik's old idea that they are now trying to make the world pay them for.**

Risskov Denmark February 16th 2004

Knud Bank

Knud Bank Christensen
Research Engineer
TC Electronic A/S