(12) PATENT APPLICATION **(11)** Application No. **AU 200072371 A1**
(19) AUSTRALIAN PATENT OFFICE

(54) Title
Method and apparatus for shortening the critical path of reduced complexity sequence estimation techniques

(51)[7] International Patent Classification(s)
H03M 013/00

(21) Application No: **200072371** (22) Application Date: **2000.12.19**

(30) Priority Data

(31) Number	(32) Date	(33) Country
09471920	**1999.12.23**	**US**

(43) Publication Date : **2001.06.28**
(43) Publication Journal Date : **2001.06.28**

(71) Applicant(s)
Lucent Technologies Inc.

(72) Inventor(s)
Kameran Azadet; Erich Franz Haratsch

(74) Agent/Attorney
SPRUSON and FERGUSON,GPO Box 3898,SYDNEY NSW 2001

METHOD AND APPARATUS FOR SHORTENING THE CRITICAL PATH OF REDUCED COMPLEXITY SEQUENCE ESTIMATION TECHNIQUES

Abstract

A method and apparatus are disclosed for improving the processing time of reduced complexity sequence estimation techniques, such as reduced state sequence estimation (RSSE) (400). The possible values for the branch metrics in the RSSE are precomputed to permit pipelining and the shortening of the critical path. Precomputing the branch metrics for all possible symbol combinations in the channel memory makes it possible to remove the branch metrics unit (BMU) (210) and decision-feedback unit (DFU) (340) from the feedback loop, thereby reducing the critical path. A look-ahead branch metrics unit (LABMU) and an intersymbol interference canceller (ISIC) (420) precompute the branch metrics for all possible values for the channel memory. At the beginning of each decoding cycle, a set of multiplexers (MUXs) select the appropriate branch metrics based on the survivor symbols in the corresponding survivor path cells (SPCs), which are then sent to an add-compare-select unit (ACSU) (220). The computational load of the precomputations is reduced for multi-dimensional trellis codes by precomputing each dimension of the multi-dimensional trellis code separately. Prefiltering techniques are used to reduce the computational complexity by shortening the channel memory. A hybrid survivor memory architecture is also disclosed for a RSSE for a channel having a channel memory of length L, where the survivors corresponding to the L past decoding cycles are stored in a register exchange architecture (REA), and survivors corresponding to later decoding cycles are stored in a trace-back architecture (TBA) or register exchange architecture (REA). Symbols are mapped to information bits to reduce the word size before being moved from the first register exchange architecture (REA) to the trace-back architecture (TBA) or the second register exchange architecture (REA).

FIG. 4

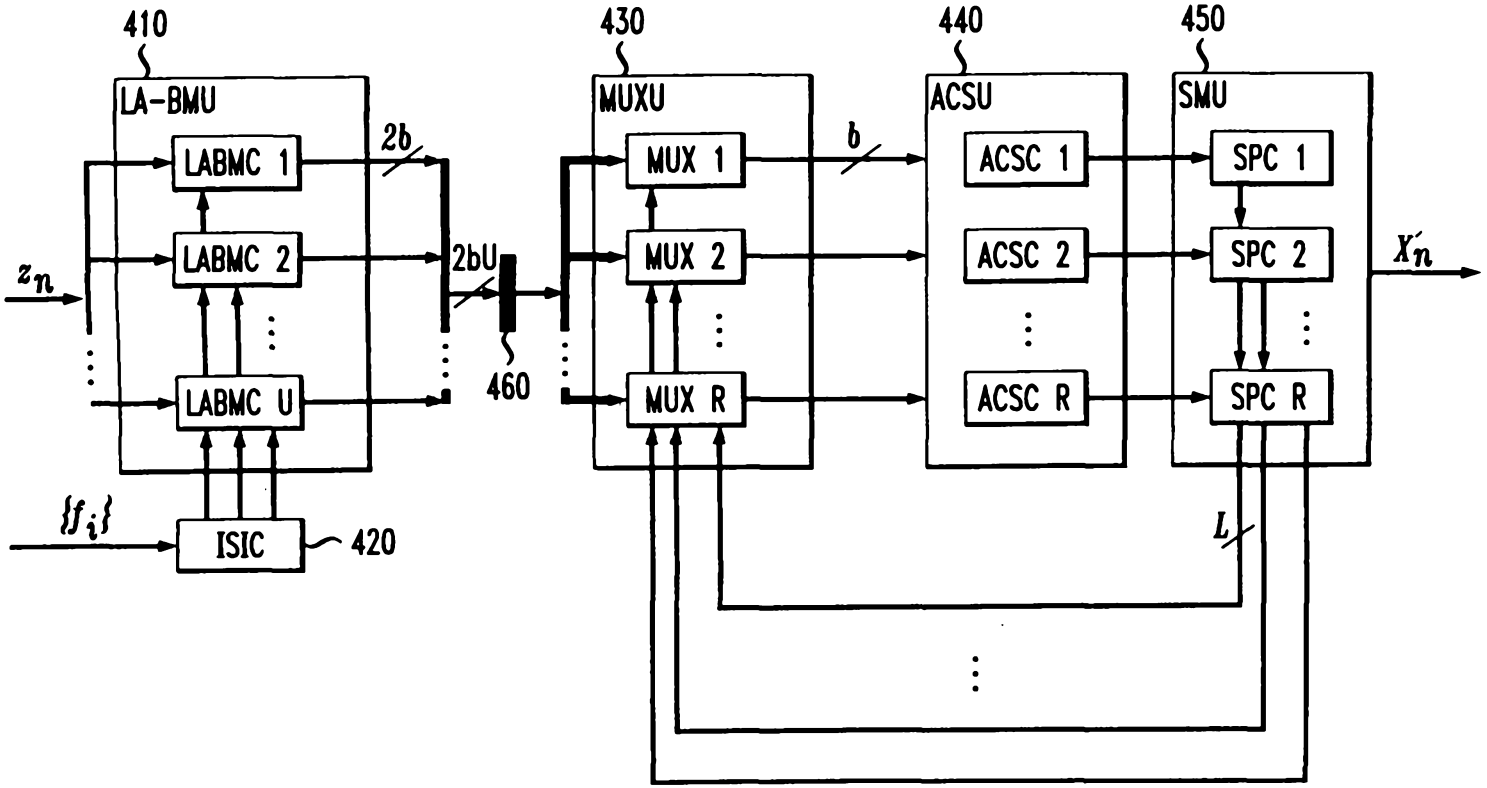
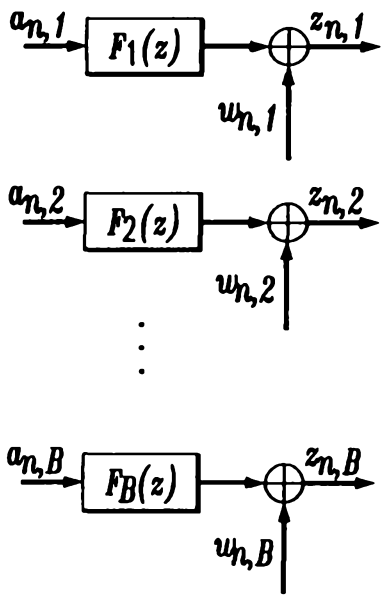

410
LA-BMU
LABMC 1
LABMC 2
LABMC U
2b
2bU
460
430
MUXU
MUX 1
MUX 2
MUX R
b
440
ACSU
ACSC 1
ACSC 2
ACSC R
450
SMU
SPC 1
SPC 2
SPC R
z_n
X'_n
$\{f_i\}$
ISIC
420
L

FIG. 5


$a_{n,1}$
$F_1(z)$
$z_{n,1}$
$u_{n,1}$
$a_{n,2}$
$F_2(z)$
$z_{n,2}$
$u_{n,2}$
$a_{n,B}$
$F_B(z)$
$z_{n,B}$
$u_{n,B}$

S&F Ref: 534157

AUSTRALIA

PATENTS ACT 1990

COMPLETE SPECIFICATION

FOR A STANDARD PATENT

ORIGINAL

Name and Address of Applicant :	Lucent Technologies Inc. 600 Mountain Avenue Murray Hill New Jersey 07974-0636 United States of America
Actual Inventor(s):	Kameran Azadet and Erich Franz Haratsch
Address for Service:	Spruson & Ferguson St Martins Tower 31 Market Street Sydney NSW 2000
Invention Title:	Method and Apparatus for Shortening the Critical Path of Reduced Complexity Sequence Estimation Techniques

The following statement is a full description of this invention, including the best method of performing it known to me/us:-

IP Australia
Documents received on:
19 DEC 2000
Sydney
Job No: 5845c

METHOD AND APPARATUS FOR SHORTENING THE CRITICAL PATH OF REDUCED COMPLEXITY SEQUENCE ESTIMATION TECHNIQUES

Field Of The Invention

The present invention relates generally to channel equalization and decoding techniques, and more particularly, to sequence estimation techniques with shorter critical paths.

Background Of The Invention

The transmission rates for local area networks (LANs) that use twisted pair conductors have progressively increased from 10 Megabits-per-second (Mbps) to 1 Gigabit-per-second (Gbps). The Gigabit Ethernet 1000 Base-T standard, for example, operates at a clock rate of 125 MHz and uses category 5 cabling with four copper pairs to transmit 1 Gbps. Trellis-coded modulation (TCM) is employed by the transmitter, in a known manner, to achieve coding gain. The signals arriving at the receiver are typically corrupted by intersymbol interference (ISI), crosstalk, echo, and noise. A major challenge for 1000 Base-T receivers is to jointly equalize the channel and decode the corrupted trellis-coded signals at the demanded clock rate of 125 MHz, as the algorithms for joint equalization and decoding incorporate non-linear feedback loops that cannot be pipelined.

Data detection is often performed using maximum likelihood sequence estimation (MLSE), to produce the output symbols or bits. A maximum likelihood sequence estimator (MLSE) considers all possible sequences and determines which sequence was actually transmitted, in a known manner. The maximum likelihood sequence estimator (MLSE) is the optimum decoder and applies the well-known Viterbi algorithm to perform joint equalization and decoding. For a more detailed discussion of a Viterbi implementation of a maximum likelihood sequence estimator (MLSE), see Gerhard Fettweis and Heinrich Meyr, "High-Speed Parallel Viterbi Decoding Algorithm and VLSI-Architecture," IEEE Communication Magazine (May 1991), incorporated by reference herein.

In order to reduce the hardware complexity for the maximum likelihood sequence estimator (MLSE) that applies the Viterbi algorithm, a number of sub-optimal approaches, such as "reduced state sequence estimation (RSSE)" algorithms, have been proposed or suggested. For a discussion of reduced state sequence estimation (RSSE) techniques, as well as the special cases of decision-feedback sequence estimation (DFSE) and parallel decision-feedback equalization (PDFE) techniques, see, for example, P. R. Chevillat and E. Eleftheriou, "Decoding of Trellis-Encoded Signals in the Presence of Intersymbol Interference and Noise", IEEE Trans. Commun., vol. 37, 669-76, (July 1989), M. V. Eyuboglu and S. U. H. Qureshi, "Reduced-State Sequence Estimation For Coded Modulation On Intersymbol Interference Channels", IEEE JSAC, vol. 7, 989-95 (Aug. 1989), or A. Duel-Hallen and C. Heegard, "Delayed decision-feedback sequence estimation," IEEE Trans. Commun., vol. 37, pp. 428-436, May 1989, each incorporated by reference herein. For a discussion of the M algorithm, see, for example, E. F. Haratsch, "High-Speed VLSI Implementation of Reduced Complexity Sequence Estimation Algorithms With Application to Gigabit Ethernet 1000 Base-T," Int'l Symposium on VLSI Technology, Systems, and Applications, Taipei (Jun. 1999), incorporated by reference herein.

Generally, reduced state sequence estimation (RSSE) techniques reduce the complexity of the maximum likelihood sequence estimators (MLSE) by merging several states. The RSSE technique incorporates non-linear feedback loops that cannot be pipelined. The critical path associated with these feedback loops is the limiting factor for high-speed implementations.

United States Patent Application Serial Number 09/326,785, filed June 4, 1999 and entitled "Method and Apparatus for Reducing the Computational Complexity and Relaxing the Critical Path of Reduced State Sequence Estimation (RSSE) Techniques," incorporated by reference herein, discloses a reduced state sequence estimation (RSSE) algorithm that reduces the hardware complexity of RSSE techniques for a given number of states and also relaxes the critical path problem. While the disclosed RSSE algorithm exhibits significantly improved processing time, additional processing gains are needed

for many high-speed applications. A need therefore exists for a reduced state sequence estimation (RSSE) algorithm with improved processing time. Yet another need exists for a reduced state sequence estimation (RSSE) algorithm that is better suited for a high-speed implementation using very large scale integration (VLSI) techniques.

Summary Of The Invention

Generally, a method and apparatus are disclosed for improving the processing time of the reduced complexity sequence estimation techniques, such as the RSSE technique, for a given number of states. According to one feature of the invention, the possible values for the branch metrics in the reduced state sequence estimation (RSSE) technique are precomputed in a look-ahead fashion to permit pipelining and the shortening of the critical path. Thus, the present invention provides a delay that is similar to a traditional optimum Viterbi decoder. Precomputing the branch metrics for all possible symbol combinations in the channel memory in accordance with the present invention makes it possible to remove the branch metrics unit (BMU) and decision-feedback unit (DFU) from the feedback loop, thereby reducing the critical path. In the illustrative implementation, the functions of the branch metrics unit (BMU) and decision-feedback unit (DFU) are performed by a look-ahead branch metrics unit (LABMU) and an intersymbol interference canceller (ISIC) that are removed from the critical path.

A reduced state sequence estimator (RSSE) is disclosed that provides a look-ahead branch metrics unit (LABMU) to precompute the branch metrics for all possible values for the channel memory. At the beginning of each decoding cycle, a set of multiplexers (MUXs) select the appropriate branch metrics based on the survivor symbols in the corresponding survivor path cells (SPCs), which are then sent to an add-compare-select unit (ACSU). The critical path now comprises one MUX, ACSC and SPC. The disclosed RSSE can be utilized for both one-dimensional and multi-dimensional trellis codes.

For multi-dimensional trellis codes where the precomputation of multi-dimensional branch metrics becomes computationaly too expensive, a modified RSSE is

disclosed to reduce the computational load. The metrics for each dimension of the multi-dimensional trellis code are precomputed separately. The appropriate one-dimensional branch metrics are then selected based on the corresponding survivor symbols in the corresponding survivor path cell (SPC) for that dimension. A multi-dimensional branch metrics unit then combines the selected one-dimensional branch metrics to form the multi-dimensional branch metrics. According to another aspect of the invention, prefiltering techniques are used to reduce the computational complexity by shortening the channel memory. An example is provided of a specific implementation for a 1000 Base-T Gigabit Ethernet implementation that truncates the postcursor channel memory length to one.

A novel memory-partitioned survivor memory architecture for the survivor memory units in the survivor path cell is also disclosed. In order to prevent latency for the storage of the survivor symbols, which are required in the decision feedback unit (DFU) or the multiplexer unit (MUXU) with zero latency, a hybrid survivor memory arrangement is disclosed for reduced state sequence estimation (RSSE). In an RSSE implementation for a channel memory of length L, the survivor symbols corresponding to the L past decoding cycles are utilized (i) for intersymbol interference cancellation in the decision-feedback units (DFU) of a conventional RSSE, and (ii) for the selection of branch metrics in the multiplexers (MUXU) in an RSSE according to the present invention. The present invention stores the survivors corresponding to the L past decoding cycles in a register exchange architecture (REA), and survivors corresponding to later decoding cycles are stored in a trace-back architecture (TBA) or register exchange architecture (REA). Before symbols are moved from the register exchange architecture (REA) to the trace-back architecture (TBA), they are mapped to information bits to reduce the word size. In a 1000 Base-T implementation, the register exchange architecture (REA) is used for the entire survivor memory, as the latency introduced by the trace-back architecture (TBA) in the second memory partition would lead to a violation of the tight latency budget specified for the receiver in the 1000 Base-T standard.

Brief Description Of The Drawings

FIG. 1 illustrates an equivalent discrete time model of a conventional trellis coded communications system;

FIG. 2 illustrates a conventional implementation of the Viterbi algorithm;

FIG. 3 illustrates the architecture for conventional implementation of an reduced state sequence estimator (RSSE);

FIG. 4 illustrates the architecture of a reduced state sequence estimator (RSSE) with precomputation of branch metrics in accordance with the present invention;

FIG. 5 illustrates the use of multi-dimensional trellis coded modulation for a multidimensional channel;

FIG. 6 illustrates the architecture for a one-dimensional precomputation for a multi-dimensional reduced state sequence estimator (RSSE) in accordance with the present invention;

FIG. 7 illustrates the architecture of a reduced state sequence estimator (RSSE) that utilizes prefiltering techniques in accordance with the present invention to shorten the channel memory;

FIG. 8 illustrates a decision-feedback prefilter for a 1000 Base-T Gigabit Ethernet implementation that truncates the postcursor channel memory length from fourteen to one;

FIG. 9 illustrates the look-ahead computation of 1D branch metrics by one of the 1D-LABMU units of FIG. 6 for the 1000 Base-T Gigabit Ethernet implementation;

FIG. 10 illustrates the selection of the 1D branch metrics by the multiplexer of FIG. 6 for the 1000 Base-T Gigabit Ethernet implementation; and

FIG. 11 illustrates a novel memory-partitioned register exchange network (SPC-n) for state one for the 1000 Base-T Gigabit Ethernet implementation.

Detailed Description

As previously indicated, the processing speed for reduced complexity sequence estimation techniques, such as reduced state sequence estimation (RSSE), is limited by a recursive feedback loop. According to one feature of the present invention, the processing speed for such reduced state sequence estimation (RSSE) techniques is improved by precomputing the branch metrics in a look-ahead fashion. The precomputation of the branch metrics shortens the critical path, such that the delay is of the same order as in a traditional Viterbi decoder. According to another feature of the present invention, the computational load of the precomputations is significantly reduced for multi-dimensional trellis codes. Prefiltering can reduce the computational complexity by shortening the channel memory. The RSSE techniques of the present invention allow the implementation of RSSE for high-speed communications systems, such as the Gigabit Ethernet 1000 Base-T standard.

TRELLIS-CODED MODULATION

As previously indicated, RSSE techniques reduce the computational complexity of the Viterbi algorithm, when the RSSE techniques are used to equalize uncoded signals or jointly decode and equalize signals, which have been coded, using trellis-coded modulation (TCM). While the present invention is illustrated herein using decoding and equalization of trellis coded signals, the present invention also applies to the equalization of uncoded signals, as would be apparent to a person of ordinary skill in the art. TCM is a combined coding and modulation scheme for band-limited channels. For a more detailed discussion of TCM, see, for example, G. Ungerboeck, "Trellis-Coded Modulation With Redundant Signal Sets," IEEE Comm., Vol. 25, No. 2, 5-21 (Feb. 1987), incorporated by reference herein. FIG. 1 illustrates the equivalent discrete time model of a trellis coded communications system.

As shown in FIG. 1, information symbols X_n consisting of m bits are fed into a TCM encoder 110. The rate $m'/(m'+1)$ encoder 110 operates on m' input bits and produces $m'+1$ encoded bits, which are used to select one of the $2^{m'+1}$ subsets (each of size $2^{m-m'}$) from the employed signal constellation of size 2^{m+1}, while the uncoded bits are used to select one symbol a_n within the chosen subset. In the illustrative implementation, Z-level pulse amplitude modulation (Z-PAM) is used as the modulation scheme for the symbols a_n. The techniques of the present invention, however, can be applied to other modulation schemes such as PSK or QAM, as would be apparent to a person of ordinary skill in the art. The selected symbol a_n is sent over the equivalent discrete-time channel. Assuming a one-dimensional channel, the channel output z_n at time instant n is given by:

$$z_n = q_n + w_n = \sum_{i=0}^{L} f_i \cdot a_{n-i} + w_n , \qquad (1)$$

where q_n is the signal corrupted by ISI, $\{f_i\}$, $i \in [0,..,L]$ are the coefficients of the equivalent discrete-time channel impulse response ($f_0 = 1$ is assumed without loss of

generality), L is the length of the channel memory, and $\{w_n\}$ represents white Gaussian noise with zero mean and variance σ^2.

The concatenation of the trellis coder and channel defines a combined code and channel state, which is given by

$$\xi_n = (\mu_n; a_{n-L}, \ldots, a_{n-1}), \qquad (2)$$

where μ_n is the code state and $\alpha_n = (a_{n-L}, \ldots, a_{n-1})$ is the channel state at time n. The optimum decoder for the received signal is the maximum likelihood sequence estimator (MLSE) that applies the Viterbi algorithm to the super trellis defined by the combined code and channel state. The computation and storage requirements of the Viterbi algorithm are proportional to the number of states. The number of states of the super trellis is given by:

$$T = S \times 2^{mL}, \qquad (3)$$

where S is the number of code states.

The Viterbi algorithm searches for the most likely data sequence by efficiently accumulating the path metrics for all states. The branch metric for a transition from state ξ_n under input a_n is given by:

$$\lambda(z_n, a_n, \xi_n) = \left(z_n - a_n - \sum_{i=1}^{L} f_i a_{n-i}\right)^2. \qquad (4)$$

Among all paths entering state ξ_{n+1} from predecessor states $\{\xi_n\}$, the most likely path is chosen according to the following path metric calculation, which is commonly referred to as add-compare-select (ACS) calculation:

$$\Gamma(\xi_{n+1}) = \min_{\{\xi_n\} \to \xi_{n+1}} (\Gamma(\xi_n) + \lambda(z_n, a_n, \xi_n)). \qquad (5)$$

An implementation of the Viterbi algorithm is shown in FIG. 2. The Viterbi implementation 200 shown in FIG. 2 comprises of a main components branch metric unit

(BMU) 210, an add-compare-select unit (ACSU) 220 and a survivor memory unit (SMU) 230. The branch metric unit (BMU) 210 calculates the metrics for the state transitions according to equation (4). The ACS unit (ACSU) 220 evaluates equation (5) for each state, and the survivor memory unit (SMU) 230 keeps track of the surviving paths. The data flow in the BMU 210 and SMU 230 is strictly feed-forward and can be pipelined at any level to increase throughput. The bottleneck for high-speed processing is the ACSU 220, as the recursion in the ACS operation in equation (5) demands that a decision is made before the next step of the trellis is decoded.

RSSE techniques reduce the complexity of the maximum likelihood sequence estimator (MLSE) by truncating the channel memory such that only the first K of the L channel coefficients $\{f_i\}$, $i \in [1,..,L]$, are taken into account for the trellis. See, A. Duel-Hallen and C. Heegard, "Delayed decision-feedback sequence estimation," IEEE Trans. Commun., vol. 37, pp. 428-436, May 1989, incorporated by reference herein. In addition, the set partitioning principles described in P. R. Chevillat and E. Eleftheriou, "Decoding of Trellis-Encoded Signals in the Presence of Intersymbol Interference and Noise," IEEE Trans. Comm., Vol. 37, 669-676 (Jul. 1989) and M.V. Eyuboglu and S. U. Qureshi, "Reduced-State Sequence Estimation for Coded Modulation on Intersymbol Interference Channels," IEEE JSAC, Vol. 7, 989-995 (Aug. 1989), each incorporated by reference herein, are applied to the signal alphabet. The reduced combined channel and code state is given in RSSE by

$$\rho_n = (\mu_n; J_{n-K},..,J_{n-1}), \qquad (6)$$

where J_{n-i} is the subset the data symbol a_{n-i} belongs to. The number of different subsets J_{n-i} is given by 2^{m_i}, where m_i defines the depth of subset partitioning at time instant $n-i$. It is required that

$$m' \le m_K \le m_{K-1} \le ... \le m_1 \le m. \qquad (7)$$

The number of states in the reduced super trellis is given as follows:

$$R = S \times 2^{m_K + ... + m_1}. \qquad (8)$$

In RSSE, the branch metric for reduced state ρ_n under input a_n takes the modified form:

$$\lambda_n(z_n, a_n, \rho_n) = (z_n - a_n + u_n(\rho_n))^2, \quad (9)$$

where:

$$u_n(\rho_n) = -\sum_{i=1}^{L} f_i \hat{a}_{n-i}(\rho_n) \quad (10)$$

$\hat{\alpha}_n(\rho_n) = (\hat{a}_{n-L}(\rho_n), \ldots, \hat{a}_{n-1}(\rho_n))$ is the survivor sequence leading to the reduced state ρ_n and $\hat{a}_{n-i}(\rho_n)$ is the associated survivor symbol at time instant $n-i$. In equation (10), an ISI estimate $u(\rho_n)$ is calculated for state ρ_n by taking the data symbols associated with the path history of state ρ_n as tentative decisions. The best path metric for state ρ_{n+1} is obtained by evaluating

$$\Gamma(\rho_{n+1}) = \min_{\{\rho_n\} \to \rho_{n+1}} (\Gamma(\rho_n) + \lambda_n(z_n, a_n, \rho_n)). \quad (11)$$

RSSE can be viewed as a sub-optimum trellis decoding algorithm where each state uses decision-feedback from its own survivor path to account for the ISI not considered in the reduced trellis.

FIG. 3 illustrates the architecture for the implementation of RSSE. As shown in FIG. 3, the decision-feedback cells (DFC) in the decision-feedback unit (DFU) 340 calculate R ISI estimates by considering the survivors in the corresponding survivor path cell (SPC) of the SMU 330 according to equation (10). Each branch metric cell (BMC) in the BMU 310 computes the metrics for the $b = 2^{m'}$ transitions leaving one state. For each state, the best path selection is performed in the ACS cell (ACSC) according to equation (11). In contrast to Viterbi decoding, the DFC, BMC, and SPC cells are in the critical loop in addition to the ACSC cell. The techniques for parallel processing of the Viterbi algorithm exploit the fact that the branch metric computation in equation (4) does not depend on the decision of the ACS function in equation (5). Thus, branch metrics

can be calculated for k trellis steps in a look-ahead fashion to obtain a k-fold increase of the throughput. Sees G. Fettweis and H. Meyr, "High-Speed Viterbi Processor: A Systolic Array Solution," IEEE JSAC, Vol. 8, 1520-1534 (Oct. 1990) or United States Patent Number 5,042,036, incorporated by reference herein. However, for RSSE techniques, the branch metric computation in equation (9) depends on the decision of the ACSC in the ACSU 320, which evaluates equation (11), in the previous symbol period, as the surviving symbols in the SPC of the SMU 330 are needed for the decision-feedback computations in equation (10). Thus, the block processing techniques described in G. Fettweis and H. Meyr, referenced above, cannot be applied to speed up the processing of RSSE.

PRECOMPUTATION OF BRANCH METRICS

The critical path in RSSE involves more operations than in the Viterbi algorithm. In particular, the branch metric computations in the BMC can be very expensive in terms of processing time, as euclidean distances have to be obtained by either squaring or performing a table-lookup to achieve good coding gain performance. Also, the evaluation of equation (10) in the DFC 340-n may have a significant contribution to the critical path. Precomputing all branch metrics for all possible symbol combinations in the channel memory in accordance with the present invention makes it possible to remove the BMU 310 and DFU 340 from the feedback loop. This potentially allows for a significant reduction of the critical path in RSSE.

In principle, the channel state $\alpha_n = (a_{n-L},..,a_{n-1})$ can take $U = \left(2^{m+1}\right)^L$ different values. The ISI estimates for a particular channel assignment $\tilde{\alpha} = (\tilde{a}_{n-L},..,\tilde{a}_{n-1})$ can be obtained by evaluating the following equation:

$$\tilde{u}(\tilde{\alpha}) = -\sum_{i=1}^{L} f_i \tilde{a}_{n-i} . \qquad (12)$$

It is noted that equation (12) does not depend on the time n and is thus a constant for a particular channel assignment $\tilde{\alpha}$. The speculative branch metric for a transition from channel assignment $\tilde{\alpha}$ under input a_n is then given by

$$\tilde{\lambda}_n(z_n, a_n, \tilde{\alpha}) = (z_n - a_n + \tilde{u}(\tilde{\alpha}))^2 . \qquad (13)$$

The trellis coder 100 in FIG. 1 defines $2b = 2^{m'+1}$ different subsets. Assuming that in the case of parallel transitions the best representative in a subset is obtained by slicing, a maximum of $M = 2b \times U = 2^{m'+1} \times 2^{(m+1)L}$ different branch metrics $\tilde{\lambda}_n(z_n, a_n, \tilde{\alpha})$ are possible and have to be precomputed. The trellis coder shown in FIG. 1 may not allow all symbol combinations in the channel memory α_n. Therefore, the number of branch metrics which have to be precomputed might be less than M. The actual number of branch metrics which have to be precomputed should be determined from the reduced super trellis.

For the add-compare-select cell (ACSC) 320-n, the appropriate branch metrics $\lambda_n(z_n, a_n, \rho_n)$ among all precomputed branch metrics $\tilde{\lambda}_n(z_n, a_n, \tilde{\alpha})$ are selected by using the survivor path $\hat{\alpha}_n(\rho_n)$:

$$\lambda_n(z_n, a_n, \rho_n) = sel\{\Lambda_n(z_n, a_n, \rho_n), \hat{\alpha}_n(\rho_n)\}. \qquad (14)$$

In equation (14), $\Lambda_n(z_n, a_n, \rho_n)$ is a vector containing the 2^{mL} branch metrics $\tilde{\lambda}_n(z_n, a_n, \tilde{\alpha})$, which can occur for a transition from state ρ_n under input a_n for different channel assignments $\tilde{\alpha}$. The selector function in equation (14) can be implemented with a 2^{mL} to 1 multiplexer.

It is noted that equations (12) and (13) are both independent from the decision in the recursive ACS function in equation (11). Thus, the precomputations in equations (12) and (13) are strictly feed-forward and can be pipelined at any level. Only the selection function in equation (14) lies in the critical path in addition to the add-compare-select cell (ACSC) and survivor path cell (SPC).

The architecture of an RSSE 400 with precomputation of branch metrics in accordance with the present invention is shown in FIG. 4. The ISI canceller (ISIC) 420 calculates all U values which can occur for $\tilde{u}(\tilde{\alpha})$. Each of these U values is used by a corresponding look-ahead BMC (LABMC) 410-n to calculate $2b$ speculative branch metrics $\tilde{\lambda}_n(z_n, a_n, \tilde{\alpha})$. All the M = $2bU$ branch metrics precomputed in the look-ahead

BMU (LABMU) 410 are then sent to the MUX unit (MUXU) 430. Then, at the beginning of each decoding cycle, each multiplexer (MUX) 430-n in the MUX unit (MUXU) 430 selects the appropriate branch metrics based on the survivor symbols in the corresponding SPC 450-n, which are then sent to the ACSU 440. Each multiplexer (MUX) 430-n in the MUX unit (MUXU) 430 takes L past symbols from the corresponding survivor path cell (SPC) 450-n. The ACSU 440 and SMU 450 may be embodied as in the conventional RSSE 300 of FIG. 3. The output of the LABMU 410 is placed in a pipeline register 460. The critical path now comprises of just the MUX 430, ACSC 440-n, and SPC 450-n. The MUX 430 selects a branch metric in accordance with equation (14) dependent on the symbols in the SPC 450-n. Although the number of precomputed branch metrics increases exponentially with the channel memory L and the number of information bits m, this technique is feasible for small m (corresponding to small symbol constellation sizes) and short L.

PRECOMPUTATION FOR MULTIDIMENSIONAL TRELLIS CODES

Significant coding gains for large signal constellations can be achieved with multidimensional TCM. FIG. 5 illustrates the use of multi-dimensional trellis coded modulation for a multidimensional channel. The B-dimensional symbol $\bar{a}_n = (a_{n,1},..,a_{n,B})$, where $\bar{a}_n$ is a vector, is sent over the B-dimensional channel with the channel coefficients $\{f_{i,j}\}$, $i \in [0,..,L]$, $j \in [1..,B]$ such that the channel output $\bar{z}_n = (z_{n,1},..,z_{n,B})$, is a vector given as

$$z_{n,j} = \sum_{i=0}^{L} f_{i,j} \cdot a_{n-i,j} + w_{n,j}, \quad j \in [1,..,B], \qquad (15)$$

where $\{w_{n,j}\}$, $j \in [1,..,B]$ are B uncorrelated independent white Gaussian noise sources. Z-PAM is considered as the transmission scheme for each channel. The following results are valid for other modulation schemes as well. Such an equivalent discrete time channel can be found for example in Gigabit Ethernet 1000 Base-T over copper, where $B=4$, $m=8, m'=2, S=8$, $Z=5$. See K. Azadet, "Gigabit Ethernet Over Unshielded Twisted

Pair Cables," Int'l Symposium on VLSI Technology, Systems, and Applications, Taipei (Jun. 1999), incorporated by reference herein.

As the complexity for the precomputation of branch metrics grows exponentially with the number of information bits m, there might be cases where the precomputation of multi-dimensional branch metrics as shown in FIG. 4 might be too computationally expensive for large signal constellation sizes. However, performing precomputations of the branch metrics only for the one-dimensional components of the code can significantly reduce the complexity.

The 1-dimensional branch metric in the dimension j is precomputed by evaluating the following expressions:

$$\tilde{\lambda}_{n,j}\left(z_{n,j}, a_{n,j}, \tilde{\alpha}_j\right) = \left(z_{n,j} - a_{n,j} + \tilde{u}_j\left(\tilde{\alpha}_j\right)\right)^2, \qquad (16)$$

$$\tilde{u}_j\left(\tilde{\alpha}_j\right) = -\sum_{i=1}^{L} f_{i,j}\,\tilde{a}_{n-i,j}, \qquad (17)$$

where $\tilde{\alpha}_j = \left(\tilde{a}_{n-L,j},..,\tilde{a}_{n-1,j}\right)$ is a particular assignment for the channel state $\alpha_j = \left(a_{n-L,j},..,a_{n-1,j}\right)$ in dimension j.

There are $V = Z^L$ possible 1-dimensional channel assignments $\tilde{\alpha}_j$. For a given channel assignment $\tilde{\alpha}_j$, C inputs $a_{n,j}$ have to be considered to calculate all possible 1-dimensional branch metrics $\tilde{\lambda}_{n,j}\left(z_{n,j}, a_{n,j}, \tilde{\alpha}_j\right)$, where C, $C \le Z$ is the number of 1-dimensional subsets. Each of these C inputs $a_{n,j}$ corresponds to the point in the corresponding subset to which $\left(z_{n,j}\right)$ has been sliced to after the cancellation of the intersymbol interference according to equations (16) and (17). Consequently, considering all B dimensions, a total $N = B \times C \times V$ 1-dimensional branch metrics have to be precomputed. This can be considerably less than the number of precomputations necessary for multidimensional precomputations as discussed above in the section entitled "Precomputation of Branch Metrics." In the case of the Gigabit Ethernet 1000 Base-T, with $C = 2$ $L = 1$, and $Z = 5$ 1- dimensional precomputation yields a total of

$4 \times 2 \times 5 = 40$ 1-dimensional branch metric computations, whereas multi-dimensional precomputation results in $2^3 \times 2^9 = 4096$ 4-dimensional branch metric computations.

The selection of the appropriate 1-dimensional branch metrics for further processing in RSSE is given by:

$$\lambda_{n,j}(z_{n,j}, a_{n,j}, \rho_n) = sel\{\Lambda_{n,j}(z_{n,j}, a_{n,j}), \hat{\alpha}_{n,j}(\rho_n)\} \quad (18)$$

where $\Lambda_{n,j}(z_{n,j}, a_{n,j})$ is the vector containing all V possible 1-dimensional branch metrics $\tilde{\lambda}_{n,j}(z_{n,j}, a_{n,j}, \tilde{\alpha}_j)$ under input $a_{n,j}$ for different one-dimensional channel assignments $\tilde{\alpha}_j$ and $\hat{\alpha}_{n,j}(\rho_n)$ is the survivor sequence in dimension j leading to state ρ_n. This can be implemented using a V to 1 MUX compared to the 2^{mL} to 1 MUX needed for multi-dimensional precomputation (e.g., in the 1000 Base-T example above, 5 to 1 MUXs are required c.f. to 256 to 1 MUXs). After the appropriate 1D branch metrics have been selected, the multidimensional branch metric is given as

$$\lambda_n(z_n, a_n, \rho_n) = \sum_{j=1}^{B} \lambda_{n,j}(z_{n,j}, a_{n,j}, \rho_n) \quad (19).$$

FIG. 6 illustrates the architecture 600 for 1-dimensional precomputation for multi-dimensional RSSE. Each 1D-ISIC 620-n calculates the V ISI cancellation terms $\tilde{u}_j(\tilde{\alpha}_j)$. For each of these $\tilde{u}_j(\tilde{\alpha}_j)$, the corresponding 1D-LABMC 610-n precomputes C one-dimensional branch metrics per channel assignment and dimension in the 1D-LABMU 610. The MUXU 630 selects for each state the appropriate one-dimensional branch metrics dependent on the survivor symbols in the SPC 660-n. Each multi-dimensional BMC (MD-BMC) 640-n calculates the multi-dimensional branch metrics by using the selected 1-dimensional branch metrics. The critical path now comprises one MUX 630, MD-BMC 640, ACSC 650 and SPC 660. The MD-BMC 640 performs $B-1$ additions and consequently has a minor contribution to the overall critical path, as the number of dimensions B is typically low.

PREFILTERING

It has been shown that the complexity for the precomputation of branch metrics increases exponentially with the channel memory L. However, using the prefilter 710, shown in FIG. 7, can shorten the channel memory. As the equivalent discrete time channel after a whitened matched filter is minimum-phase, the channel memory can be truncated with a decision feedback prefilter (DFP) to low values of L without significant performance loss for RSSE, as described in E. F. Haratsch, "High-Speed VLSI Implementation of Reduced Complexity Sequence Estimation Algorithms With Application to Gigabit Ethernet 1000 Base-T," Int'l Symposium on VLSI Technology, Systems, and Applications, Taipei (Jun. 1999) and United States Patent Application Serial Number 09/326,785, filed June 4, 1999 and entitled "Method and Apparatus for Reducing the Computational Complexity and Relaxing the Critical Path of Reduced State Sequence Estimation (RSSE) Techniques," each incorporated by reference herein. Alternatively, the prefilter 710 could be implemented as a linear filter, such as those described in D.D. Falconer and F.R. Magee, "Adaptive Channel Memory Truncation for Maximum-Likelihood Sequence Estimation," The Bell Systems Technical Journal, Vol. 52, No. 9, 1541-62 (Nov. 1973), incorporated by reference herein.

Thus, for channels with large channel memories where the precomputation of branch metrics is too expensive, a prefilter could be used to truncate the channel memory such that precomputation becomes feasible.

1000-BASE T GIGABIT ETHERNET EXAMPLE

The following is an example of a specific implementation for a 1000 Base-T Gigabit Ethernet receiver. For a detailed discussion of the 1000 Base-T Gigabit Ethernet standard and related terminology and computations used herein, see, for example, M. Hatamian et al., "Design considerations for Gigabit Ethernet 1000 Base-T twisted pair transceivers," Proc. CICC, Santa Clara, CA, pp. 335-342, May 1998, incorporated by reference herein.

A decision-feedback prefilter for the 1000 Base-T Gigabit Ethernet implementation is shown in FIG. 8. The look-ahead computation of 1D branch metrics by one of the 1D-LABMU units of FIG. 6 for the 1000 Base-T Gigabit Ethernet implementation is shown in FIG. 9. FIG. 10 illustrates the selection of the 1D branch metrics by the multiplexer of FIG. 6 for the 1000 Base-T Gigabit Ethernet implementation. Finally, FIG. 11 illustrates the register exchange network (SPC n) for state one for the 1000 Base-T Gigabit Ethernet implementation, where an illustrative merge depth of 14 is utilized for the SMU.

Decision-Feedback Prefilter

A decision-feedback prefilter 800 that truncates the postcursor memory length on wire pair j from fourteen to one is shown in FIG. 8. The decision-feedback prefilter 800 resembles the structure of a decision-feedback equalizer (DFE) as it uses tentative decisions obtained by its own slicer to remove the tail of the postcursor channel impulse response.

Precomputation of 1D branch metrics

As the effective postcursor channel memory is one after the decision-feedback prefilter 800, the computational complexity for look-ahead precomputations of 1D branch metrics on each wire pair is modest. The speculative 1D branch metric for wire pair j under the assumption that the channel memory contains $\tilde{a}_{n-1,j}$ is

$$\tilde{\lambda}_{n,j}\left(y_{n,j}, a_{n,j}, \tilde{a}_{n-1,j}\right) = \left(y_{n,j} - a_{n,j} - f_{1,j}\tilde{a}_{n-1,j}\right)^2. \qquad (20)$$

As there are 5 possible values for $\tilde{a}_{n,j}$, and as $y_{n,j}$ after removal of intersymbol interference has to be sliced to the closest representative of both 1D subsets A as well as B, a total of 10 1D branch metrics have to be precomputed per wire pair. This is shown in FIG. 9, where the slicers 910-n calculate the difference to the closest point in 1D subset A or B. There is one clock cycle time for one addition, slicing, and squaring. It should be noted that the computational complexity of precomputing branch metrics increases exponentially with the channel memory. If the channel memory were two, 50

1D branch metrics would have to be precomputed per wire pair, and for a channel memory of three this number would increase to 250.

Selection of 1D Branch Metrics

The MUXU 630 selects for each wire pair j and code state ρ_n the appropriate 1D branch metrics corresponding to subsets A and B based on the past survivor symbol $\hat{a}_{n-i,j}(\rho_n)$. This is done with 5:1 MUXs 1010 as shown in FIG. 10. In total, 64 such multiplexers are needed.

Computation of 4D Branch Metrics

The 4D-BMU 640 adds up the 1D branch metrics to calculate the 4D branch metrics corresponding to state transitions in the trellis. The 4D-BMU 640 is in the critical loop. Bringing the 4D-BMU 640 out of the critical loop by look-ahead precomputations of 4D branch metrics would be impractical in terms of computational complexity, as shown in the example discussed above in the section entitled "Precomputation of Multi-Dimensional Trellis Codes." It can be easily seen that there are too many possibilities, which must be considered.

Add-Compare-Select

For each state, a 4-way ACS has to be performed. To speed up the processing, the architecture proposed in P.J. Black and T.H. Meng, "A 140-Mb/s, 32-state, radix-4 Viterbi decoder," IEEE JSSC, vol. 27, pp. 1877-1885, Dec. 1992, has been chosen, where the minimum path metric among the 4 candidates is selected by 6 comparisons in parallel. State metric normalization is done using modulo arithmetic, See, A.P. Hekstra, "An Alternative To Metric Rescaling In Viterbi Decoders", IEEE Trans. Commun., vol. 37, pp. 1220-1222, Nov. 1989.

Survivor Memory

In Viterbi decoding, usually the trace-back architecture (TBA) is the preferred architecture for the survivor memory as it has considerably less power consumption than

the register exchange architecture (REA). R. Cypher and C.B. Shung, "Generalized Trace-Back Techniques For Survivor Memory Management In The Viterbi Algorithm," Journal of VLSI Signal Processing, vol. 5, pp. 85-94, 1993. However, as the trace-back architecture (TBA) introduces latency it cannot be used to store the survivor symbols, which are required in the DFU or MUXU with zero latency. Thus, a hybrid survivor memory arrangement seems to be favorable for a reduced state sequence estimation (RSSE) implementation for a channel of memory length L. The survivors corresponding to the L past decoding cycles are stored in a register exchange architecture (REA), and survivors corresponding to later decoding cycles in a trace-back architecture (TBA). Before symbols are moved from the register exchange architecture (REA) to the trace-back architecture (TBA), they are mapped to information bits to reduce the word size. However, in 1000 Base-T the register exchange architecture (REA) must be used for the entire survivor memory, as the latency introduced by the trace-back architecture (TBA) would lead to a violation of the tight latency budget specified for the receiver in the 1000 Base-T standard. Likewise, symbols moved from the first register exchange architecture (REA) to the second register exchange architecture (REA) are mapped to information bits to reduce the word size.

The survivor memory architecture is shown in FIG. 11, where only the first row corresponding to state one is shown. $SX_n(\rho_n)$ denotes the decision for 4D subset SX for a transition from state ρ_n (for definition of 4D subsets see, Hatamian et al.), $\hat{b}_{n-i}(\rho_n)$ are the 8 information bits which correspond to the 4D survivor symbol $\hat{a}_{n-i}(\rho_n)$ and $d_n(1)$ is the 2-bit decision of the ACS for state one. As the channel memory seen by the reduced state sequence estimation (RSSE) is one, only the first column stores 4D symbols, which are represented by 12 bits and are fed into the MUXU. After this first column, the survivor symbols are mapped to information bits and then stored as 8 bits. For a merge depth of 14, this architecture needs 928 REGs compared to 1344 REGs in an SMU which does not apply the hybrid memory partition, where all decisions are stored as 12 bit 4D symbols.

It is to be understood that the embodiments and variations shown and described herein are merely illustrative of the principles of this invention and that various modifications may be implemented by those skilled in the art without departing from the scope and spirit of the invention.

Claims The claims defining the invention are as follows:

1. A method for processing a signal received from a dispersive channel using a reduced complexity sequence estimation technique, said channel having a channel memory, said method comprising the steps of:

precomputing branch metrics for each possible value of said channel memory;

selecting one of said precomputed branch metrics based on past decisions from corresponding states; and

selecting a path having a best path metric for a given state.

2. The method of claim 1, wherein said precomputed branch metrics for a transition from channel assignment $\tilde{\alpha}$ under input a_n is given by:

$$\tilde{\lambda}_n(z_n, a_n, \tilde{\alpha}) = (z_n - a_n + \tilde{u}(\tilde{\alpha}))^2 .$$

where an intersymbol interference estimate for a particular channel assignment $\tilde{\alpha} = (\tilde{a}_{n-L}, .., \tilde{a}_{n-1})$ can be obtained by evaluating the following equation:

$$\tilde{u}(\tilde{\alpha}) = -\sum_{i=1}^{L} f_i \tilde{a}_{n-i} .$$

3. The method of claim 1, wherein said path metric is an accumulation of said corresponding branch metrics over time.

4. The method of claim 1, wherein an appropriate branch metrics $\lambda_n(z_n, a_n, \rho_n)$ is selected from said precomputed branch metrics $\tilde{\lambda}_n(z_n, a_n, \tilde{\alpha})$ using the survivor path $\hat{\alpha}_n(\rho_n)$:

$$\lambda_n(z_n, a_n, \rho_n) = sel\{\Lambda_n(z_n, a_n, \rho_n), \hat{\alpha}_n(\rho_n)\}.$$

where $\Lambda_n(z_n, a_n, \rho_n)$ is a vector containing the branch metrics $\tilde{\lambda}_n(z_n, a_n, \tilde{\alpha})$, which can occur for a transition from state ρ_n under input a_n for different channel assignments $\tilde{\alpha}$ and $\hat{\alpha}_n(\rho_n)$ is the survivor sequence leading to state ρ_n.

5. The method of claim 1, wherein said best path metric is a minimum or maximum path metric.

6. The method of claim 1, wherein said reduced complexity sequence estimation technique is a reduced state sequence estimation (RSSE) technique.

7. The method according to claim 6, wherein said reduced state sequence estimation (RSSE) technique is a decision-feedback sequence estimation (DFSE) technique.

8. The method according to claim 6, wherein said reduced state sequence estimation (RSSE) technique is a parallel decision-feedback equalization (PDFE) technique.

9. The method of claim 1, wherein said reduced complexity sequence estimation technique is an implementation of the Viterbi algorithm.

10. The method of claim 1, wherein said reduced complexity sequence estimation technique is an implementation of the M algorithm.

11. The method of claim 1, wherein said past decisions from corresponding states are based on past symbols from a corresponding survivor path cell (SPC).

12. The method of claim 1, wherein said past decisions from corresponding states are based on past decisions from a corresponding add-compare-select cell (ACSC).

13. A method for processing a multi-dimensional trellis code signal received from a dispersive channel using a reduced complexity sequence estimation technique, said channel having a channel memory, said method comprising the steps of:

precomputing a one-dimensional branch metric for each possible value of said channel memory and for each dimension of the multi-dimensional trellis code;

selecting one of said precomputed one-dimensional branch metric based on past decisions from corresponding states; and

combining said selected one-dimensional branch metrics to obtain a multi-dimensional branch metric.

14. The method of claim 13, wherein said one-dimensional branch metric in the dimension j is precomputed by evaluating the following expressions:

$$\tilde{\lambda}_{n,j}(z_{n,j}, a_{n,j}, \tilde{\alpha}_j) = (z_{n,j} - a_{n,j} + \tilde{u}_j(\tilde{\alpha}_j))^2 \text{ and } \tilde{u}_j(\tilde{\alpha}_j) = -\sum_{i=1}^{L} f_{i,j}\tilde{a}_{n-i,j},$$

where $\tilde{\alpha}_j = (\tilde{a}_{n-L,j},..,\tilde{a}_{n-1,j})$ is a particular assignment for the channel state $\alpha_j = (a_{n-L,j},..,a_{n-1,j})$ in dimension j.

15. The method of claim 13, wherein said selection of an appropriate one-dimensional branch metrics for further processing with a reduced complexity sequence estimator is given by:

$$\lambda_{n,j}(z_{n,j}, a_{n,j}, \rho_n) = sel\{\Lambda_{n,j}(z_{n,j}, a_{n,j}), \hat{\alpha}_{n,j}(\rho_n)\}$$

where $\Lambda_{n,j}(z_{n,j}, a_{n,j})$ is the vector containing possible one-dimensional branch metrics $\tilde{\lambda}_{n,j}(z_{n,j}, a_{n,j}, \tilde{\alpha}_j)$ under input $a_{n,j}$ for different one-dimensional channel assignments $\tilde{\alpha}_j$ and $\hat{\alpha}_{n,j}(\rho_n)$ is the survivor sequence in dimension j leading to state ρ_n.

16. The method of claim 13, wherein said past decisions from corresponding states are based on past symbols from a corresponding survivor path cell (SPC).

17. The method of claim 13, wherein said past decisions from corresponding states are based on past decisions from a corresponding add-compare-select cell (ACSC).

18. A method for processing a multi-dimensional trellis code signal received from a dispersive channel using a reduced complexity sequence estimation technique, said channel having a channel memory, said method comprising the steps of:

precomputing a one-dimensional branch metric for each possible value of said channel memory and for each dimension of the multi-dimensional trellis code;

combining said one-dimensional branch metric into at least two-dimensional branch metrics; and

selecting one of said at least two-dimensional branch metrics based on past decisions from corresponding states.

19. The method of claim 18, wherein said one-dimensional branch metric in the dimension j is precomputed by evaluating the following expressions:

$$\tilde{\lambda}_{n,j}\left(z_{n,j},a_{n,j},\tilde{\alpha}_j\right)=\left(z_{n,j}-a_{n,j}+\tilde{u}_j\left(\tilde{\alpha}_j\right)\right)^2 \text{ and } \tilde{u}_j\left(\tilde{\alpha}_j\right)=-\sum_{i=1}^{L} f_{i,j}\tilde{a}_{n-i,j} ,$$

where $\tilde{\alpha}_j=\left(\tilde{a}_{n-L,j},..,\tilde{a}_{n-1,j}\right)$ is a particular assignment for the channel state $\alpha_j=\left(a_{n-L,j},..,a_{n-1,j}\right)$ in dimension j.

20. The method of claim 18, wherein said selection of an appropriate at least two-dimensional branch metrics corresponding to a particular state and input for further processing with a reduced complexity sequence estimator is based on the survivor symbols for said state and said at least two dimensions and said selection is performed among all precomputed at least two-dimensional branch metrics for said state, input and different channel assignments for said dimensions.

21. The method of claim 18, wherein said past decisions from corresponding states are based on past symbols from a corresponding survivor path cell (SPC).

22. The method of claim 18, wherein said past decisions from corresponding states are based on past decisions from a corresponding add-compare-select cell (ACSC).

23. The method of claim 18, further comprising the step of combining said selected at least two-dimensional branch metric to obtain a multi-dimensional branch metric.

24. A method for processing a signal received from a dispersive channel using a reduced complexity sequence estimation technique, said channel having a channel memory, said method comprising the steps of:

prefiltering said signal to shorten said channel memory;

precomputing branch metrics for each possible value of said shortened channel memory;

selecting one of said precomputed branch metrics based on past decisions from corresponding states; and

selecting a path having a best path metric for a given state.

25. The method of claim 24, wherein said prefiltering step further comprises the step of processing less significant taps with a lower complexity cancellation algorithm that cancels the less significant taps using tentative decisions and processing more significant taps with a reduced state sequence estimation (RSSE) technique.

26. The method according to claim 24, wherein said lower complexity cancellation algorithm is a decision feedback prefilter (DFP) technique.

27. The method according to claim 24, wherein said lower complexity cancellation algorithm utilizes a linear equalizer technique.

28. The method according to claim 24, wherein said lower complexity cancellation algorithm is a soft decision feedback prefilter (DFP) technique.

29. The method according to claim 24, wherein said lower complexity cancellation algorithm reduces the intersymbol interference associated with said less significant taps.

30. The method according to claim 24, wherein said more significant taps comprise taps below a tap number, U, where U is a prescribed number less than L.

31. The method according to claim 24, wherein said reduced complexity sequence estimation technique is a decision-feedback sequence estimation (DFSE) technique.

32. The method according to claim 24, wherein said reduced complexity sequence estimation technique is a parallel decision-feedback equalization (PDFE) technique.

33. The method according to claim 24, wherein said reduced complexity sequence estimation technique is a reduced state sequence estimation (RSSE) technique.

34. The method according to claim 24, wherein said reduced complexity sequence estimation technique is an implementation of the Viterbi algorithm.

35. The method according to claim 24, wherein said reduced complexity sequence estimation technique is an implementation of the M algorithm.

36. The method of claim 24, wherein said past decisions from corresponding states are based on past symbols from a corresponding survivor path cell (SPC).

37. The method of claim 24, wherein said past decisions from corresponding states are based on past decisions from a corresponding add-compare-select cell (ACSC).

38. A method for processing a signal received from a dispersive channel using a reduced complexity sequence estimation technique, said channel having a channel memory, said method comprising the steps of:

prefiltering said signal to shorten said channel memory;

precomputing a one-dimensional branch metric for each possible value of said shortened channel memory and for each dimension of the multi-dimensional trellis code;

combining said one-dimensional branch metric into at least two-dimensional branch metrics; and

selecting one of said at least two-dimensional branch metrics based on past decisions from corresponding states.

39. A hybrid survivor memory architecture for a reduced complexity sequence estimator for a channel having a channel memory of length L, comprising:

a register exchange architecture (REA) for storing the survivors corresponding to the L past decoding cycles; and

a trace-back architecture (TBA) for storing survivors corresponding to later decoding cycles, wherein symbols moved from said register exchange architecture (REA) to said trace-back architecture (TBA) are mapped to information bits.

40. The survivor memory architecture of claim 39, wherein said reduced complexity sequence estimation technique is a reduced state sequence estimation (RSSE) technique.

41. The survivor memory architecture of claim 39, wherein said reduced complexity sequence estimation technique is an implementation of the Viterbi algorithm.

42. The survivor memory architecture of claim 39, wherein said reduced complexity sequence estimation technique is an implementation of the M algorithm.

43. A hybrid survivor memory architecture for a reduced complexity sequence estimator for a channel having a channel memory of length L, comprising:

a first register exchange architecture (REA) for storing the survivors corresponding to the L past decoding cycles; and

a second register exchange architecture (REA) for storing survivors corresponding to later decoding cycles, wherein symbols moved from said first register exchange architecture (REA) to said second register exchange architecture (REA) are mapped to information bits.

44. The survivor memory architecture of claim 43, wherein said reduced complexity sequence estimation technique is a reduced state sequence estimation (RSSE) technique.

45. The survivor memory architecture of claim 43, wherein said reduced complexity sequence estimation technique is an implementation of the Viterbi algorithm.

46. The survivor memory architecture of claim 43, wherein said reduced complexity sequence estimation technique is an implementation of the M algorithm.

47. A reduced complexity sequence estimator for processing a signal received from a dispersive channel having a channel memory, comprising:

a look-ahead branch metrics unit for precomputing branch metrics for each possible value of said channel memory;

a multiplexer for selecting one of said precomputed branch metrics based on past decisions from corresponding states; and

an add-compare-select unit for selecting a path having a best path metric for a given state.

48. The reduced complexity sequence estimator of claim 47, wherein said past decisions from corresponding states are based on past symbols from a corresponding survivor path cell (SPC).

49. The reduced complexity sequence estimator of claim 47, wherein said past decisions from corresponding states are based on past decisions from a corresponding add-compare-select cell (ACSC).

50. A reduced complexity sequence estimator for processing a signal received from a dispersive channel having a channel memory, comprising:

a look-ahead branch metrics unit for precomputing a one-dimensional branch metric for each possible value of said channel memory and for each dimension of the multi-dimensional trellis code;

a multiplexer for selecting one of said precomputed one-dimensional branch metric based on past decisions from corresponding states; and

a multi-dimensional branch metric cell for combining said selected one-dimensional branch metrics to obtain a multi-dimensional branch metric.

51. The reduced complexity sequence estimator of claim 50, wherein said past decisions from corresponding states are based on past symbols from a corresponding survivor path cell (SPC).

52. The reduced complexity sequence estimator of claim 50, wherein said past decisions from corresponding states are based on past decisions from a corresponding add-compare-select cell (ACSC).

53. A reduced complexity sequence estimator for processing a signal received from a dispersive channel having a channel memory, comprising:

a look-ahead branch metrics unit for precomputing a one-dimensional branch metric for each possible value of said channel memory and for each dimension of the multi-dimensional trellis code;

means for combining said one-dimensional branch metric into at least two-dimensional branch metrics;

a multiplexer for selecting one of said at least two-dimensional branch metrics based on past decisions from corresponding states; and

a multi-dimensional branch metric cell for combining said selected at least two-dimensional branch metric to obtain a multi-dimensional branch metric.

54. The reduced complexity sequence estimator of claim 53, wherein said past decisions from corresponding states are based on past symbols from a corresponding survivor path cell (SPC).

55. The reduced complexity sequence estimator of claim 53, wherein said past decisions from corresponding states are based on past decisions from a corresponding add-compare-select cell (ACSC).

56. A reduced complexity sequence estimator for processing a signal received from a dispersive channel having a channel memory, comprising:

a prefilter to shorten said channel memory;

a look-ahead branch metrics unit for precomputing branch metrics for each possible value of said channel memory;

a multiplexer for selecting one of said precomputed branch metrics based on past decisions from corresponding states; and

an add-compare-select unit for selecting a path having a best path metric for a given state.

57. The reduced complexity sequence estimator of claim 56, wherein said past decisions from corresponding states are based on past symbols from a corresponding survivor path cell (SPC).

58. The reduced complexity sequence estimator of claim 56, wherein said past decisions from corresponding states are based on past decisions from a corresponding add-compare-select cell (ACSC).

59. A reduced complexity sequence estimator for processing a signal received from a dispersive channel having a channel memory, comprising:

a prefilter to shorten said channel memory;

a multi-dimensional look-ahead branch metrics unit for precomputing a one-dimensional branch metric for each possible value of said shortened channel memory and for each dimension of the multi-dimensional trellis code;

means for combining said one-dimensional branch metric into at least two-dimensional branch metrics; and

a multiplexer for selecting one of said at least two-dimensional branch metrics based on past decisions from corresponding states.

60. A method for processing a signal, said method being substantially as described herein in relation to any one embodiment with reference to Figs. 1, 4 - 11.

61. A hybrid survivor memory architecture substantially as described herein in relation to any one embodiment with reference to Figs. 1, 4 - 11.

62. A reduced complexity sequence estimator substantially as described herein in relation to any one embodiment with reference to Figs. 1, 4 - 11.

DATED this Thirteenth Day of December, 2000

Lucent Technologies Inc.

Patent Attorneys for the Applicant

SPRUSON & FERGUSON

FIG. 1

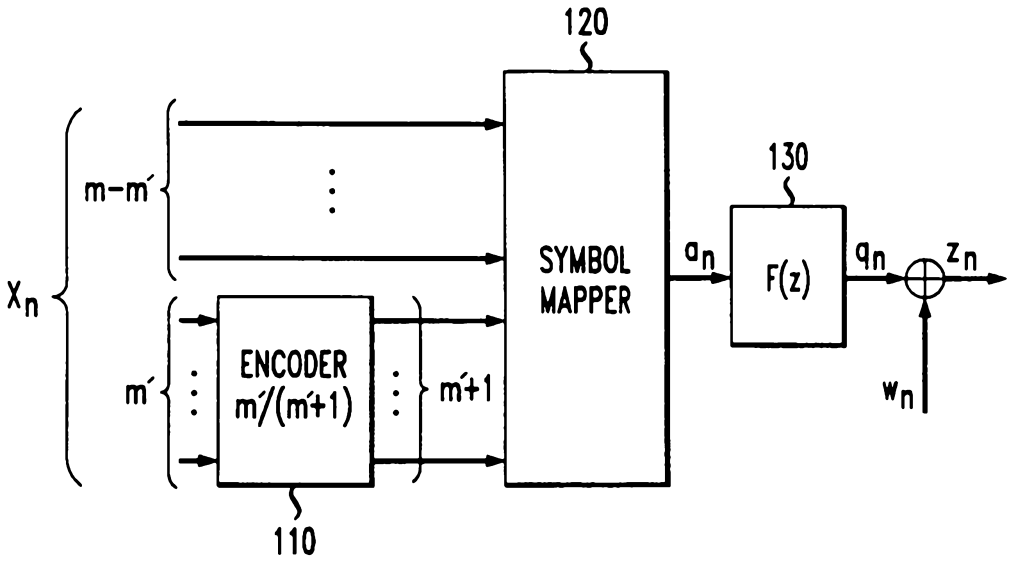


FIG. 2
PRIOR ART

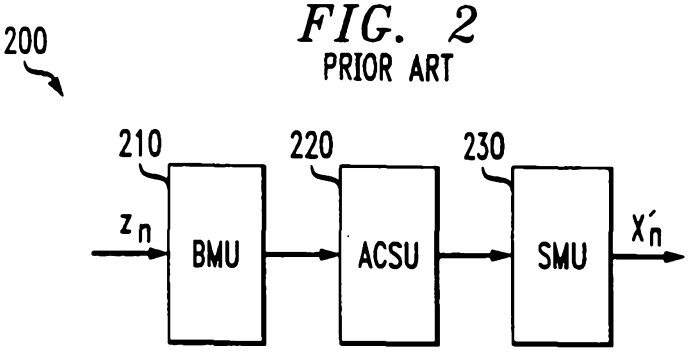


FIG. 3
PRIOR ART

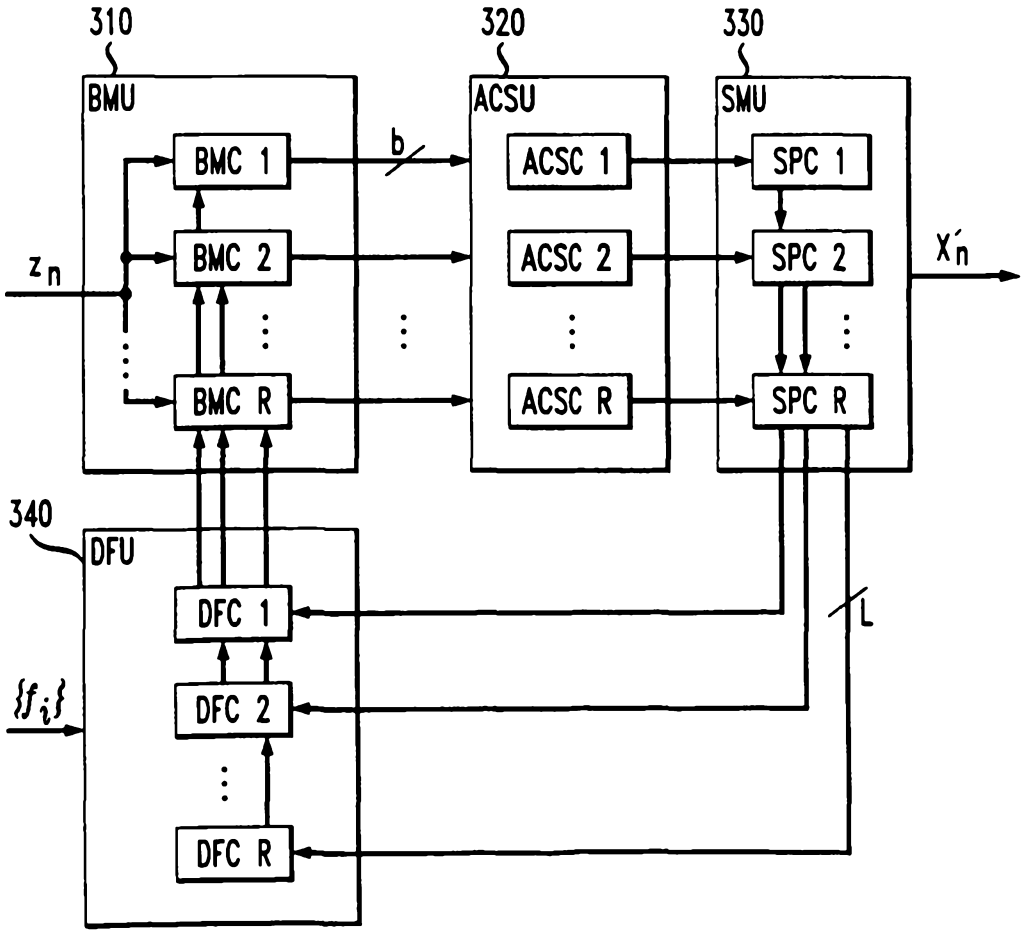


FIG. 4

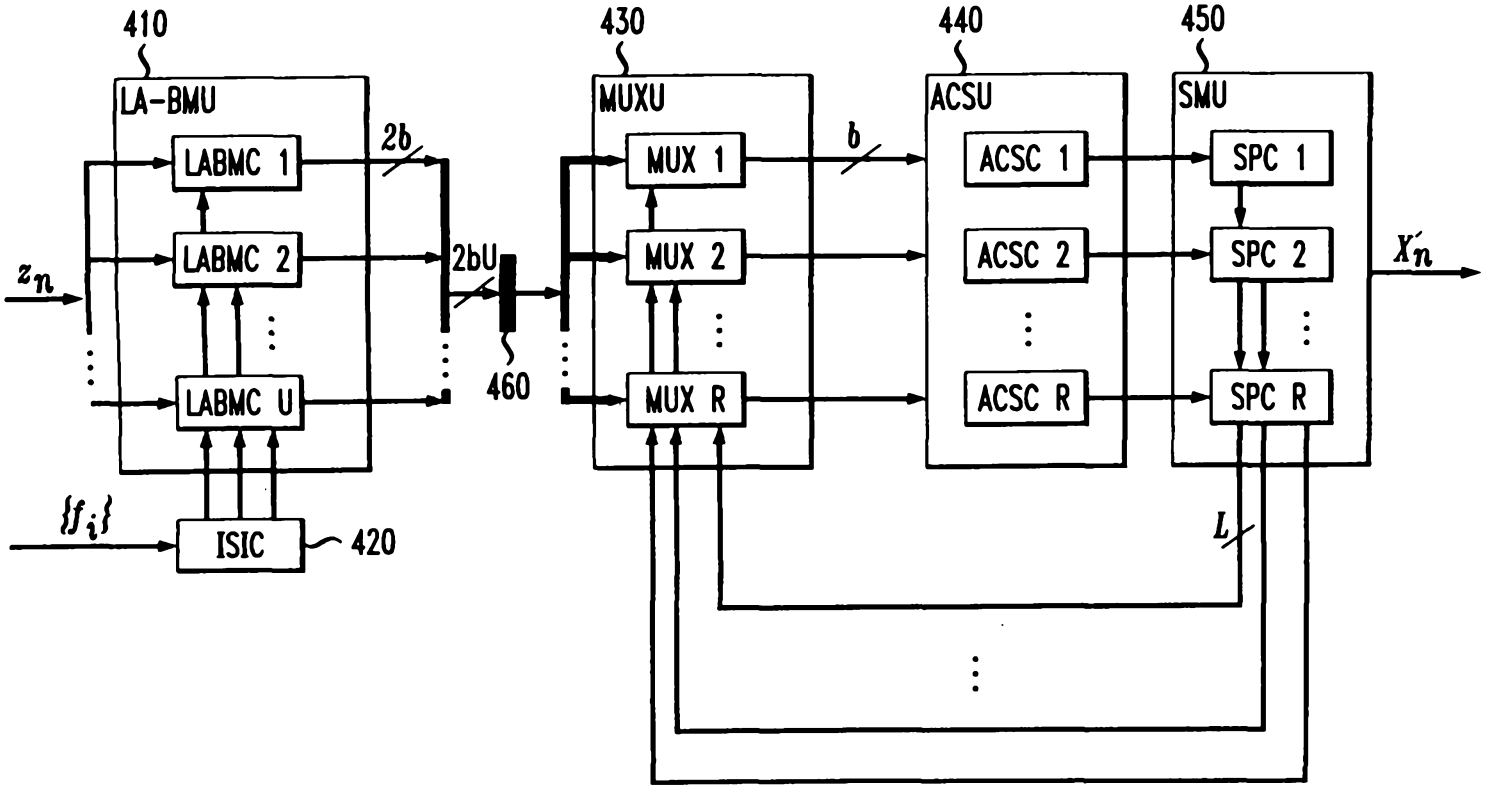
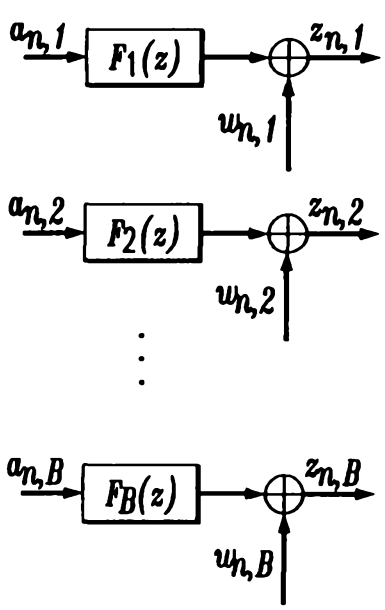

410
LA-BMU
LABMC 1
LABMC 2
LABMC U
2b
2bU
460
430
MUXU
MUX 1
MUX 2
MUX R
b
440
ACSU
ACSC 1
ACSC 2
ACSC R
450
SMU
SPC 1
SPC 2
SPC R
z_n
X'_n
$\{f_i\}$
ISIC
420
L

FIG. 5


$a_{n,1}$
$F_1(z)$
$z_{n,1}$
$u_{n,1}$
$a_{n,2}$
$F_2(z)$
$z_{n,2}$
$u_{n,2}$
$a_{n,B}$
$F_B(z)$
$z_{n,B}$
$u_{n,B}$

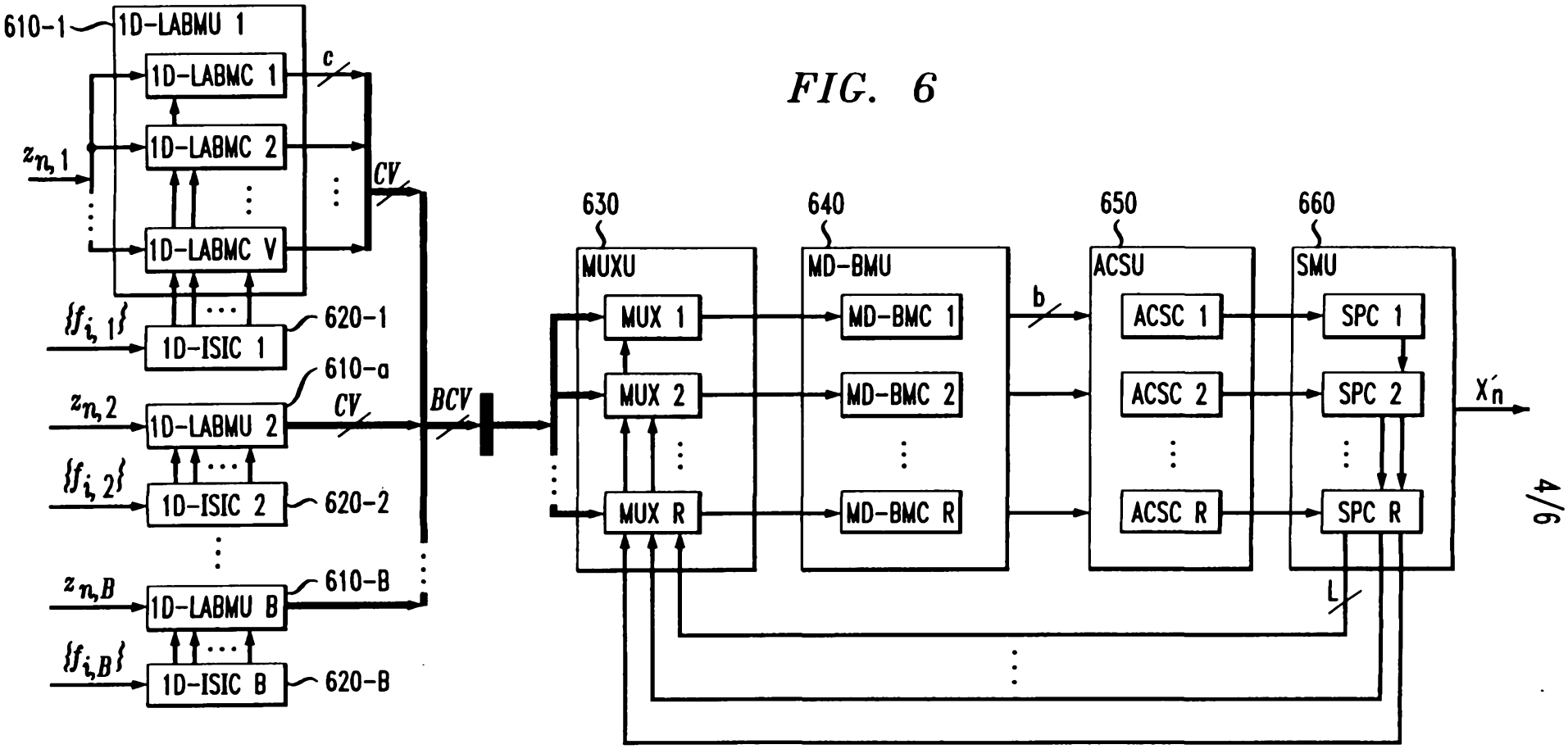

610-1
1D-LABMU 1
1D-LABMC 1
1D-LABMC 2
1D-LABMC V
c
CV
$z_{n,1}$
$\{f_{i,1}\}$
1D-ISIC 1
620-1
610-a
$z_{n,2}$
CV
BCV
1D-LABMU 2
$\{f_{i,2}\}$
1D-ISIC 2
620-2
$z_{n,B}$
610-B
1D-LABMU B
$\{f_{i,B}\}$
1D-ISIC B
620-B
630
MUXU
MUX 1
MUX 2
MUX R
640
MD-BMU
MD-BMC 1
MD-BMC 2
MD-BMC R
b
650
ACSU
ACSC 1
ACSC 2
ACSC R
660
SMU
SPC 1
SPC 2
SPC R
L
X'_n

FIG. 6

FIG. 7

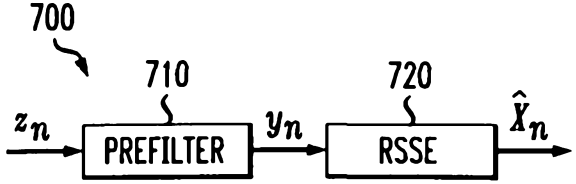

700
710
720
z_n
PREFILTER
y_n
RSSE
$\hat{X}_n$

FIG. 8

800

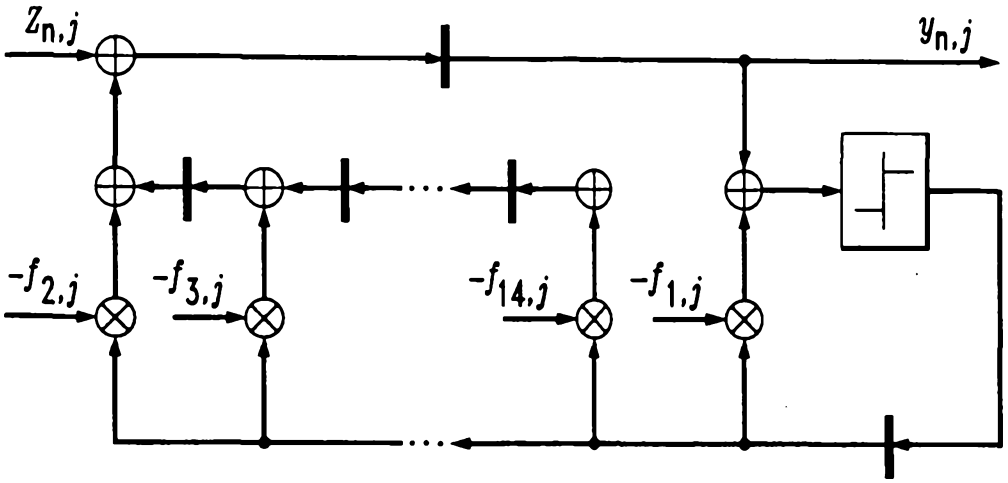


FIG. 9

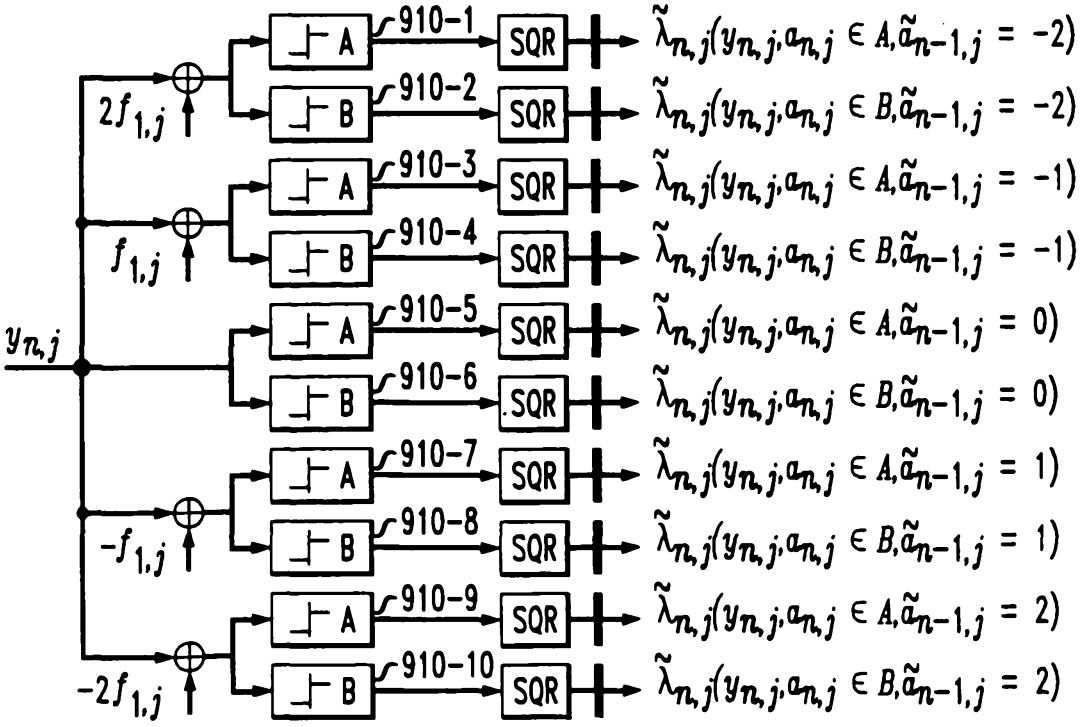


FIG. 10

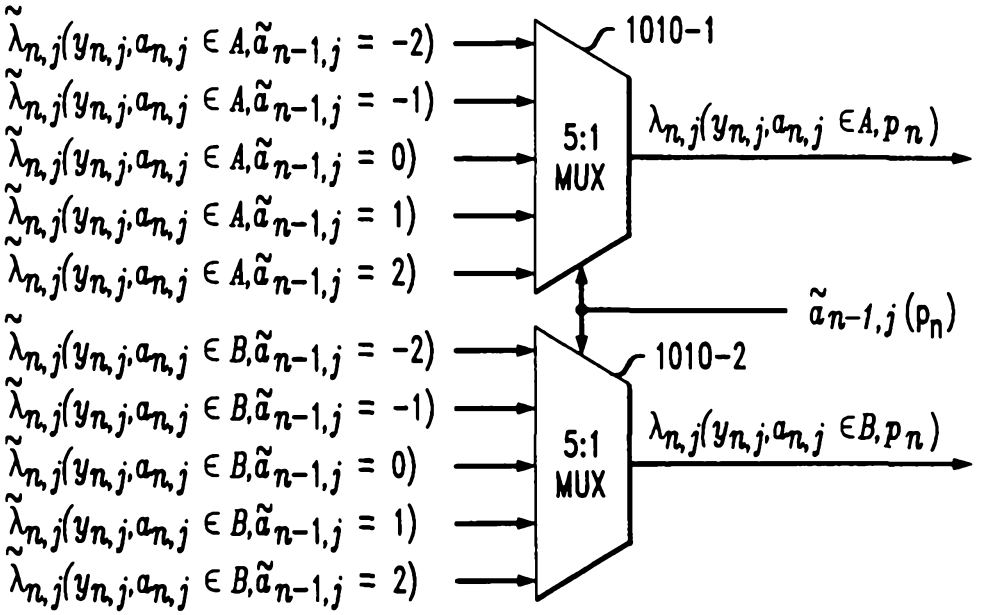
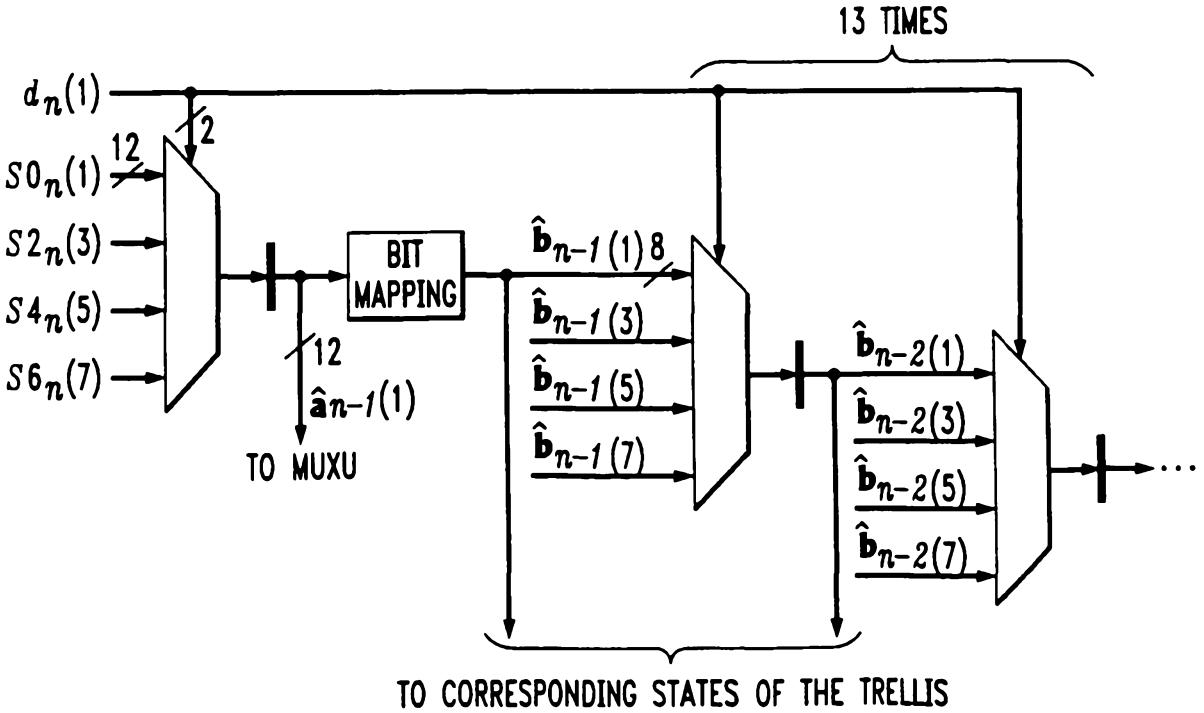

$\tilde{\lambda}_{n,j}(y_{n,j},a_{n,j} \in A,\tilde{a}_{n-1,j} = -2)$
$\tilde{\lambda}_{n,j}(y_{n,j},a_{n,j} \in A,\tilde{a}_{n-1,j} = -1)$
$\tilde{\lambda}_{n,j}(y_{n,j},a_{n,j} \in A,\tilde{a}_{n-1,j} = 0)$
$\tilde{\lambda}_{n,j}(y_{n,j},a_{n,j} \in A,\tilde{a}_{n-1,j} = 1)$
$\tilde{\lambda}_{n,j}(y_{n,j},a_{n,j} \in A,\tilde{a}_{n-1,j} = 2)$
1010-1
5:1 MUX
$\lambda_{n,j}(y_{n,j},a_{n,j} \in A,p_n)$
$\tilde{a}_{n-1,j}(p_n)$
$\tilde{\lambda}_{n,j}(y_{n,j},a_{n,j} \in B,\tilde{a}_{n-1,j} = -2)$
$\tilde{\lambda}_{n,j}(y_{n,j},a_{n,j} \in B,\tilde{a}_{n-1,j} = -1)$
$\tilde{\lambda}_{n,j}(y_{n,j},a_{n,j} \in B,\tilde{a}_{n-1,j} = 0)$
$\tilde{\lambda}_{n,j}(y_{n,j},a_{n,j} \in B,\tilde{a}_{n-1,j} = 1)$
$\tilde{\lambda}_{n,j}(y_{n,j},a_{n,j} \in B,\tilde{a}_{n-1,j} = 2)$
1010-2
5:1 MUX
$\lambda_{n,j}(y_{n,j},a_{n,j} \in B,p_n)$

FIG. 11


13 TIMES
$d_n(1)$
2
12
$S0_n(1)$
$S2_n(3)$
$S4_n(5)$
$S6_n(7)$
BIT MAPPING
12
$\hat{a}_{n-1}(1)$
TO MUXU
$\hat{b}_{n-1}(1)$ 8
$\hat{b}_{n-1}(3)$
$\hat{b}_{n-1}(5)$
$\hat{b}_{n-1}(7)$
$\hat{b}_{n-2}(1)$
$\hat{b}_{n-2}(3)$
$\hat{b}_{n-2}(5)$
$\hat{b}_{n-2}(7)$
TO CORRESPONDING STATES OF THE TRELLIS