PARALLEL STRUCTURES FOR JOINT CHANNEL ESTIMATION AND DATA DETECTION OVER FADING CHANNELS

M. Javad Omidi **P. G. Gulak** **S. Pasupathy**
Department of Electrical and Computer Engineering
University of Toronto, Toronto, Ontario, Canada

Abstract— In this paper new parallel structures are proposed for joint data and channel estimation over frequency selective Rayleigh fading channels. Maximum Likelihood Sequence Estimation (MLSE) is implemented using the per-survivor processing (PSP) method [1]. The Kalman filter and the Recursive Least Squares (RLS) algorithm are considered as estimation methods. A Square-root implementation of the Kalman filter is discussed. The algorithm used for the measurement update in the Kalman filter results in significant simplicity, once it is used for realization of the RLS algorithm. Two parallel and pipelined architectures are introduced for the RLS algorithm, and an overall architecture is proposed to implement the MLSE receiver, combining the Viterbi decoder and the channel estimator.

1. INTRODUCTION

An optimum detection technique for the received data over a frequency-selective multipath fading channel is MLSE. The Viterbi algorithm is usually used to implement the MLSE receiver; however, to achieve ideal performance in the receiver perfect knowledge about the behavior of the channel is required. The fading channel is a time variant system and its impulse response has to be estimated based on the available information at the receiver, (i.e., the received signal and the detected data sequence).

There are different approaches for estimating the channel impulse response as explained in [2], and most of them suffer from the problem of inherent decision delay, that causes poor tracking performance in the channel estimation. In a method based on the application of PSP [1][3], this difficulty is eliminated by estimating the channel states along the surviving paths associated with each state of the trellis diagram.

The performance of the receiver is affected by the accuracy of the channel estimator. Estimation accuracy depends on the estimation algorithm, the method of implementation, and the numerical precision of digital hardware. The Kalman filter and the RLS algorithm both are recursive algorithms and can be employed as the channel estimation method. Numerical errors and round-off noise are causes for concern in the hardware implementation of these algorithms. Square-root filtering is an effective method to combat these problems and various architectures are

0-7803-3134-6/96 $5.00 © 1996 IEEE

proposed for square-root implementation of the Kalman filter and RLS algorithm [4],[5].

In the following sections we will consider the implementation of the estimator for channel estimation in a communication receiver; and different architectures will be proposed for the channel estimator and the combination of the RLS and the Viterbi algorithms in an MLSE receiver structure. Section 2 is an overview of the mobile communication system under consideration, and the proposed receiver algorithm for joint data detection and channel estimation. In section 3, we present the algorithms required for the implementation of the Kalman filter. Different methods for implementation of the RLS algorithm, and new pipelined structures for implementation of joint RLS and the Viterbi algorithm are proposed in section 4. Finally concluding remarks are given in section 6.

2. THE COMMUNICATION SYSTEM

The signal model for a communication system using a DQPSK signaling scheme is shown in Fig. 1. In this context, the fading channel includes the transmitter shaping filter that leads to a corresponding state space model for this channel [6]. The state of such a system is a vector composed of r subsequent channel impulse responses as

$$\boldsymbol{x}_k = (\boldsymbol{h}_k, \boldsymbol{h}_{k-1}, \boldsymbol{h}_{k-2}, \ldots, \boldsymbol{h}_{k-r+1})^T \tag{1}$$

where at sampling time k the channel impulse response, $\boldsymbol{h}_k$, is a $(\beta+1)$ dimensional complex Gaussian random vector

$$\boldsymbol{h}_k = (h_{k,0}, h_{k,1}, \ldots, h_{k,\beta})^T \tag{2}$$

A model for a two-ray Rayliegh fading channel is shown in Fig. 2. The Gaussian complex random signals, $x(k)$ and $y(k)$, are shaped in the fading filter according to the maximum Doppler frequency shift, to produce the multiplicative coefficients. The fading filter can be approximated with the third order transfer function of

$$P(z) = \frac{D}{1 - Az^{-1} - Bz^{-2} - Cz^{-3}} \tag{3}$$

Based on (3) an ARMA representation for the channel impulse response is introduced in [3] as

$$\boldsymbol{h}_k = A\boldsymbol{I}\boldsymbol{h}_{k-1} + B\boldsymbol{I}\boldsymbol{h}_{k-2} + C\boldsymbol{I}\boldsymbol{h}_{k-3} + D\boldsymbol{I}\boldsymbol{w}_k \tag{4}$$

where $\boldsymbol{w}_k$ is a $(m+1) \times 1$ zero mean white Gaussian process with the covariance matrix defined as $E\{\boldsymbol{w}_k \boldsymbol{w}_l^T\} = \boldsymbol{Q}\delta_{kl}$. According to (4), $\boldsymbol{h}_k$ only depends on its three past values and the state vector of (1) only includes three successive impulse

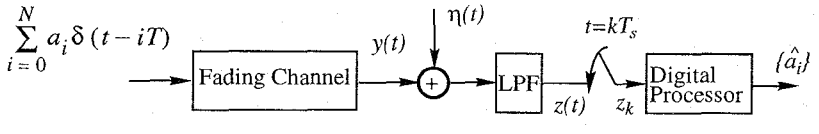

Fig. 1: The Signal Model for the communication system.

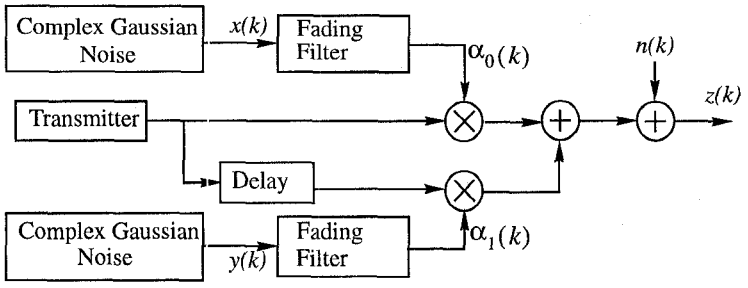

Fig. 2: The fading channel model

responses, i.e. $r=3$. Using (1) and (4) we can write

$$\boldsymbol{x}_{k+1} = \begin{bmatrix} \boldsymbol{AI} & \boldsymbol{BI} & \boldsymbol{CI} \\ \boldsymbol{I} & 0 & 0 \\ 0 & \boldsymbol{I} & 0 \end{bmatrix} \boldsymbol{x}_k + \begin{bmatrix} \boldsymbol{DI} \\ 0 \\ 0 \end{bmatrix} \boldsymbol{w}_k \tag{5}$$

or

$$\boldsymbol{x}_{k+1} = \boldsymbol{F}\boldsymbol{x}_k + \boldsymbol{G}\boldsymbol{w}_k \tag{6}$$

where $\boldsymbol{F}$ and $\boldsymbol{G}$ are $3(\beta+1)\times 3(\beta+1)$ and $3(\beta+1)\times(\beta+1)$ matrices respectively, and $\boldsymbol{I}$ is the identity matrix. $\boldsymbol{F}$ is called the state transition matrix and $\boldsymbol{G}$ is the process noise coupling matrix.

By defining the $3(\beta+1)\times 1$ vector $\boldsymbol{H}_k$ as

$$\boldsymbol{H}_k = (b_k, b_{k-1}, b_{k-2}, \ldots, b_{k-\beta}, 0, \ldots, 0) \tag{7}$$

where b_k is the transmitted information sequence at the sampling intervals [6], we can write the received signal, z_k, as

$$z_k = \boldsymbol{H}_k \boldsymbol{x}_k + n_k \tag{8}$$

This represents the convolution sum where, $\boldsymbol{H}_k$ is the input data sequence to a fading channel with impulse response $\boldsymbol{x}_k$, and n_k is the additive Gaussian noise with the covariance of $E\{n_k n_l^*\} = N_o \delta_{kl}$.

Equations (6) and (8) describe a linear time varying system. The state of this system is $\boldsymbol{x}_k$, or the impulse response of the channel, and an estimation method has to be used for channel estimation. There are different estimation methods; among them the Kalman filter is optimum for minimizing the mean square estimation error [7]. However, the Kalman filter is a computationally intensive algorithm, and in practice suboptimal methods are more advantageous due to the resulting implementation simplicity.

To avoid the decision delay in data detection, the joint data and channel estimation method of [3] can be used. In this method, there is a channel estimate for every possible transmitted data sequence, $\boldsymbol{H}_k$. Each estimator uses its own hypothesized data vector for $\boldsymbol{H}_k$ and based on that and the received signal, it provides an estimate of the channel impulse response. Only the surviving paths keep and update their channel estimate; and hence the data sequence of the shortest path is used for the channel estimation along the same path.

The Simulated System

To illustrate the design methodology, a data communication system based on the IS-54 standard is considered. The modulation is DQPSK with four possible symbols (±1±j), and a symbol rate of 25 ksymbol/s. As in the IS-54 standard, the differentially encoded data sequence is arranged into 162 symbol frames. The first 14 symbols of each frame is a training preamble sequence to help the adaptation of the channel estimator. For the shaping filter at the transmitter, we implement an FIR filter which approximates a raised cosine frequency response with an excess bandwidth of 25% (slightly different from the 35% in the IS-54). The length of the shaping filter discrete impulse response is equal to one symbol interval and hence the ISI at the receiver is due to the multipath nature of the channel.

The fading channel is a symbol-spaced two-path model with time varying complex coefficients. The two fading paths are independent with equal strength. The total length of the channel impulse response is two symbol intervals and by taking one sample per symbol interval at the receiver the length of the impulse response will be two samples, i.e. in (7) $\boldsymbol{H}_k$ is a 1×6 vector with two nonzero components.

For estimating the channel impulse response, both the RLS algorithm and the Kalman filter can be used. In the following section we will consider the square-root filtering method for the implementation of these algorithms.

3. IMPLEMENTING THE ESTIMATOR

Channel estimation can be carried out with the Kalman filter or the RLS algorithm, which are both recursive algorithms. The Kalman filter consists of two parts: *Measurement Update Equations*, and *Time Update Equations*. The RLS algorithm is almost identical to the measurement update equations of the Kalman filter [3]. The Kalman filter can be used when some a priori information about the channel is available at the receiver (i.e. the maximum Doppler frequency shift over the fading channel). When this information is not available the RLS algorithm, which is a suboptimal method, can be used instead.

The main reason for the differences between theory and practice of implementing these algorithms can be found in an error analysis of numerical methods involved. The solution is defined in terms of real number system with infinite precision, but it is implemented on digital computers in finite precision. Computer round off errors can seriously degrade the performance of the estimator. Using more precision in computations can reduce roundoff errors but the procedural details of the implementation algorithm can also influence the accuracy of the result. At the same precision, mathematically equivalent implementations can have different numerical stabilities, and some methods of implementation are more robust to roundoff errors. Factorization methods and "square root" filtering are well known for their numerical stability and are widely employed as implementation techniques [4][5].

Square-root Filtering

The so-called "square root" filter implementations have generally better error propagation bounds than the conventional Kalman filter equations. In the square root forms of the Kalman filter, matrices are factorized, and triangular square roots

are propagated in the recursive algorithm to preserve the symmetry of the covariance (Information) matrices in the presence of roundoff errors.

A square root algorithm is proposed in [4] to compute the measurement update equations of the Kalman filter for real numbers; however, it is not in itself complete for use in implementing the Kalman filter. In the following we will modify the algorithm to include complex numbers and then we will add some procedures for computing the time update equations.

To estimate the states of the system described by (6) and (8), the Kalman filter and the RLS algorithm can be employed as in Fig. 3, where $\hat{\boldsymbol{x}}_k$ is the estimated state of the system, $\boldsymbol{P}_k$ is the error covariance matrix of this estimation, and λ is the RLS algorithm forgetting factor [7]. It is clear that the RLS algorithm is almost identical to the measurement update equations of the Kalman filter.

The measurement update equation for the covariance matrix of the Kalman filter can be written as

$$\boldsymbol{P}_{k|k} = \boldsymbol{P}_k - \boldsymbol{P}_k \boldsymbol{H}_k^T R_k^{-1} \boldsymbol{H}_k \boldsymbol{P}_k \tag{9}$$

The implementation algorithm is based on working with ***LDU*** (unit lower triangular, diagonal, unit upper triangular) factorizations of $\boldsymbol{P}_k$ and $\boldsymbol{P}_{k|k}$, and since $\boldsymbol{P}_k$ and $\boldsymbol{P}_{k|k}$ are symmetric, $\boldsymbol{U}=\boldsymbol{L}^T$.

It can be shown [4] that by choosing a suitable orthogonal transformation matrix Θ we can have

$$\begin{bmatrix} N_o^{1/2} & \boldsymbol{H}_k \boldsymbol{P}_k^{1/2} \\ 0 & \boldsymbol{P}_k^{1/2} \end{bmatrix} \Theta = \begin{bmatrix} R_k^{1/2} & 0 \\ \boldsymbol{P}_k \boldsymbol{H}_k^T R_k^{-T/2} & \boldsymbol{P}_{k|k}^{1/2} \end{bmatrix} \tag{10}$$

and this can be immediately verified by "squaring" both sides of (10). Generally, computing the triangular factor $\boldsymbol{P}_{k|k}^{1/2}$ requires taking arithmetic square roots, which are usually more expensive than multiplication or division. This can be avoided by using ***LDU*** factorizations

$$\boldsymbol{P}_k = \boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^T \quad \text{and} \quad \boldsymbol{P}_{k|k} = \boldsymbol{L}_p \boldsymbol{D}_p \boldsymbol{L}_p^T \tag{11}$$

The Kalman Filter Algorithm	The RLS Algorithm
Measurement Update Equations:	
$\hat{\boldsymbol{x}}_{k\|k} = \hat{\boldsymbol{x}}_k + \boldsymbol{K}_k(z_k - \boldsymbol{H}_k\hat{\boldsymbol{x}}_k)$	$\hat{\boldsymbol{x}}_{k+1} = \hat{\boldsymbol{x}}_k + \boldsymbol{K}_k(z_k - \boldsymbol{H}_k\hat{\boldsymbol{x}}_k)$
$\boldsymbol{K}_k = \boldsymbol{P}_k\boldsymbol{H}_k^T R_k^{-1}$	$\boldsymbol{K}_k = \boldsymbol{P}_k\boldsymbol{H}_k^T R_k^{-1}$
$R_k = \boldsymbol{H}_k\boldsymbol{P}_k\boldsymbol{H}_k^T + N$	$R_k = \boldsymbol{H}_k\boldsymbol{P}_k\boldsymbol{H}_k^T + \lambda$
$\boldsymbol{P}_{k\|k} = \boldsymbol{P}_k - \boldsymbol{K}_k\boldsymbol{H}_k\boldsymbol{P}_k$	$\boldsymbol{P}_{k+1} = \lambda^{-1}(\boldsymbol{P}_k - \boldsymbol{K}_k\boldsymbol{H}_k\boldsymbol{P}_k)$
Time Update Equations:	
$\hat{\boldsymbol{x}}_{k+1} = \boldsymbol{F}\hat{\boldsymbol{x}}_{k\|k}$	
$\boldsymbol{P}_{k+1} = \boldsymbol{F}\boldsymbol{P}_{k\|k}\boldsymbol{F}^T + \boldsymbol{G}\boldsymbol{Q}\boldsymbol{G}^T$	

Fig. 3: The Kalman filter and RLS algorithms

Dropping all time-index subscripts (10) can be rewritten as

$$\begin{bmatrix} 1 & \boldsymbol{HL} \\ 0 & \boldsymbol{L} \end{bmatrix} \begin{bmatrix} N_o & 0 \\ 0 & \boldsymbol{D} \end{bmatrix}^{1/2} \Theta = \begin{bmatrix} 1 & 0 \\ \boldsymbol{K} & \boldsymbol{L}_p \end{bmatrix} \begin{bmatrix} R & 0 \\ 0 & \boldsymbol{D}_p \end{bmatrix}^{1/2} \tag{12}$$

Therefore to compute the measurement update equation for the covariance matrix we start with the left hand side of (12) and by applying an orthogonal transformation, the upper triangular matrix on the left side can be converted to the lower triangular matrix on the right side. This is possible by introducing weighted norms and rotating the vector $[1\ p_2]$ to lie along the vector [1 0], keeping equality of the weighted norms

$$\begin{bmatrix} 1 & p_2 \end{bmatrix} \begin{bmatrix} d_{p1} & 0 \\ 0 & d_{p2} \end{bmatrix} \begin{bmatrix} 1 \\ p_2^* \end{bmatrix} = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} d_{q1} & 0 \\ 0 & d_{q2} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \tag{13}$$

where p^* is the complex conjugate of p.

It can be verified that by knowing all parameters in the right hand side, we must choose

$$d_{q1} = d_{p1} + |p_2|^2 d_{p2} \tag{14}$$

and

$$d_{q2} = \frac{d_{p1} d_{p2}}{d_{q1}} \tag{15}$$

to obtain the proper orthogonal transformation matrix in

$$\begin{bmatrix} 1 & p_2 \end{bmatrix} \begin{bmatrix} d_{p1} & 0 \\ 0 & d_{p2} \end{bmatrix}^{1/2} \Theta = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} d_{q1} & 0 \\ 0 & d_{q2} \end{bmatrix}^{1/2} \tag{16}$$

By comparing (12) and (16) we can find that the values of d_{q1} and d_{q2} are components of R and $\boldsymbol{D}_p$ in the right hand side of (12) and the next step is to find $\boldsymbol{K}$ and $\boldsymbol{L}_p$ in the unit lower triangular matrix. This can be done by applying the transformation in (16) to an arbitrary vector $[p_1', p_2']$, and we obtain

$$\begin{bmatrix} p_1' & p_2' \end{bmatrix} \begin{bmatrix} d_{p1} & 0 \\ 0 & d_{p2} \end{bmatrix}^{1/2} \Theta = \begin{bmatrix} q_1' & q_2' \end{bmatrix} \begin{bmatrix} d_{q1} & 0 \\ 0 & d_{q2} \end{bmatrix}^{1/2} \tag{17}$$

where

$$q_2' = -p_2 p'_1 + p'_2 \tag{18}$$

and

$$q_1' = p'_1 + \left(p_2^* \frac{d_{p2}}{d_{q1}} \right) q_2' \tag{19}$$

A parallel architecture for implementation of this algorithm with real numbers is proposed in [4].

Implementing Time Update Equations

The above algorithm would be sufficient for implementing the RLS estimator, since the RLS algorithm is basically the same as the measurement update equations

of the Kalman filter. However, for implementing the Kalman filter we need to calculate the time update equations. Since in the above algorithm instead of $\boldsymbol{P}_{k|k}$, its factors, $\boldsymbol{L}_p$ and $\boldsymbol{D}_p$ are computed, we need to employ an algorithm to use these factors. The Weighted Gram-Schmidt Orthogonalization is usually employed for this purpose.

In this method the covariance update equation implementation is based on a Block Matrix Factorization. In the time update equations of the Kalman filter the covariance update equation can be rewritten in the following matrix form

$$\boldsymbol{P}_{k+1} = \begin{bmatrix} \boldsymbol{F}\boldsymbol{P}_{k|k}^{1/2} & \boldsymbol{G}\boldsymbol{Q}^{1/2} \end{bmatrix} \begin{bmatrix} \boldsymbol{P}_{k|k}^{T/2}\boldsymbol{F}^T \\ \boldsymbol{Q}^{T/2}\boldsymbol{G}^T \end{bmatrix} \tag{20}$$

Again if we use the $\boldsymbol{LDU}$ factorization for the covariance matrix as in (11) and show the diagonal matrix $\boldsymbol{Q}$ with $\boldsymbol{D}_q$, after dropping all time index subscripts, (20) becomes

$$\boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^T = \begin{bmatrix} \boldsymbol{F}\boldsymbol{L}_p & \boldsymbol{G} \end{bmatrix} \begin{bmatrix} \boldsymbol{D}_p & 0 \\ 0 & \boldsymbol{D}_q \end{bmatrix} \begin{bmatrix} \boldsymbol{L}_p^T\boldsymbol{F}^T \\ \boldsymbol{G}^T \end{bmatrix} \tag{21}$$

$\boldsymbol{L}_p$ and $\boldsymbol{D}_p$ in the right side of the equation are known from the measurement update procedure, and $\boldsymbol{L}$ and $\boldsymbol{D}$ have to be computed.The $\boldsymbol{LDU}$ factorization of (21) can be performed based on the Weighted Gram-Schmidt Orthogonalization [8].

Before applying this algorithm a matrix multiplication is required to compute the $\boldsymbol{FL}$. Also in the time update equations, the $\hat{\boldsymbol{x}}_{k|k}$ vector obtained at the output of measurement update procedure, must be pre-multiplied by $\boldsymbol{F}$. Both of these multiplications can be carried out at one stage by appending the vector $\hat{\boldsymbol{x}}_{k|k}$ to the matrix $\boldsymbol{L}$, and multiplying the combination by $\boldsymbol{F}$. If the multiplication is performed using array processors, in this way, the same structure can perform both of these multiplications, without any change in the hardware.

The above algorithms were employed to simulate the Kalman filter and the RLS algorithm with different number of bits for the mantissa in the floating point operations. Based on the simulation results the Kalman filter requires 22 bits in the mantissa to have a reasonable bit error rate (BER) performance at the receiver. While for the RLS algorithm the number of bits required is approximately 12. Of course the best attainable result with the RLS algorithm is inferior to that of the Kalman filter which requires more intense computations.

In the following section we propose different parallel structures for the implementation of the RLS algorithm for channel estimation in the communication system of section 2. We will also present a pipelined architecture for merging the estimator and the Viterbi algorithm for joint data and channel estimation.

4. PARALLEL STRUCTURES FOR THE RLS ALGORITHM

The square-root filtering method of section 3 can be used for both the measurement update equations of the Kalman filter and the RLS algorithm, however, in the following we will show that the implementation can be simplified drastically for the RLS algorithm leading to an affordable hardware realization.

As described in section 2, the total length of the channel impulse response is assumed to be equal to 2 samples and $\boldsymbol{H}_k$ is a 1×6 vector with two nonzero components. In the RLS algorithm of Fig. 3, $\boldsymbol{x}_k$ is only used in the first equation and is multiplied by $\boldsymbol{H}_k$ in which only the first two elements are nonzero. This means that in the RLS algorithm, where the matrix $\boldsymbol{F}$ is not used, we are not considering the ARMA representation of (4) and at each time instance only the previous impulse response is used to generate a new estimate. This will reduce the size of the matrices by a factor of r=3 compared to the Kalman filter. Another interesting consequence is that the lower triangular Matrix $\boldsymbol{L}$ in the $\boldsymbol{LDU}$ decomposition will reduce to a 2×2 matrix with only one sub-diagonal element to be updated in each iteration. The algorithm of Fig. 4 implements the RLS method based on the above observations. The matrices are reduced in size and L is essentially a scalar. The state vector $\boldsymbol{X}$, the covariance factors L and $\boldsymbol{D}$, and the *Lambda* factor will be set to initial values before calling the function for the first time. Then the output values can be used as the input for the next iteration along with the new received signal Z and the new hypothesis vector $\boldsymbol{H}$.

A parallel structure for this algorithm is shown in Fig. 5, this is a simplified version for the structure proposed in [4]. At the left side, by arranging the input data as shown in the figure, $\boldsymbol{H}_k\boldsymbol{L}$ and $\boldsymbol{H}_k\boldsymbol{x}_k$-$z_k$ are computed and passed to the processor μ. Since *H(1)* and *H(2)* choose one of the four values (±1±j), the multipliers will be trivial. The input data is also buffered in a delay unit until the coefficients for the processors on the right hand side are calculated in the μ processor. Once a_1 and a_2 are computed and loaded into the processing units, and *b1* and *A0* are ready as inputs, the new values for the state estimates and the covariance factor L_p will be computed.

The internal structure of the μ processor is shown in Fig. 6. It computes the required coefficients and the covariance factors of the vector $\boldsymbol{D}_p$. Before any computation, the input values are first arranged in the buffers and then applied to the processing elements. It is helpful to note that different precisions are required at

```
function [Xp,Lp,Dp] = RLS(Z,H,X,L,D,Lambda)
    A0=H*X-Z;
    b(1)=H(1)+H(2)*L;
    b(2)=H(2);
    Delta(3) = Lambda;
    for i= 2:-1:1,
        Delta(i) = Delta(i+1) + (b(i)*b(i)'*D(i));
        M=D(i)/Delta(i);
        a(i)= b(i)'*M;
        Dp(i) = Delta(i+1)*M/sqrt(Lambda);
    end
    Xp(1)= X(1) - A0*a(1);
    Xp(2)= X(2) - A0*a(2);
    Lp= L - b(1)*a(2);
```

Fig. 4: The simplified algorithm for the RLS estimator

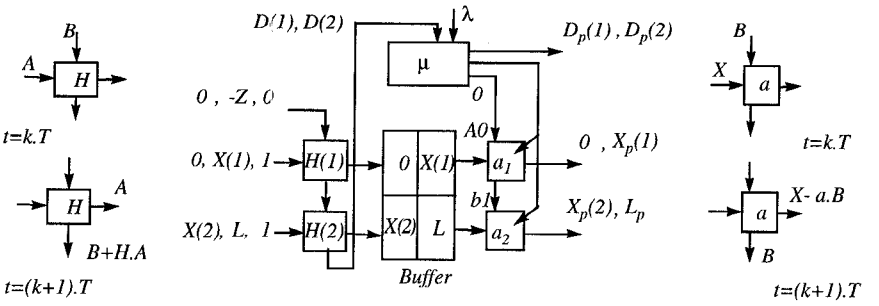

Fig. 5: A parallel structure for implementation of the RLS Algorithm

different processing units, for example for representing λ, 4 bits are sufficient, and this will reduce the size of required hardware.

In the final receiver structure more than one channel estimation is required during each symbol interval. Using one estimator for this purpose slows down the process and building several identical units of the above structure is not cost efficient. A proper solution to this problem is to seek pipeline methods where the same hardware can be utilized to carry out the required channel estimations at a higher speed and with more cost-effectiveness. The above structure is not suitable for pipelining, and the parallelism among the instructions of the algorithm in Fig. 4 has to be exploited to obtain an appropriate structure for pipelining.

It can be verified that the structure of Fig. 7 is implementing the above RLS algorithm. There are eight distinct pipeline stages, and the computational result of each one is passed to the subsequent stage. The new channel estimates and new covariance factors are computed in different stages, and moreover, the proposed architecture computes the required branch metrics (BM) for the Viterbi algorithm. The overall speed-up compared to the non-pipelined structure depends on the latency of the pipeline stage with maximum delay.

In the joint data and channel estimation method the RLS estimator has to be used in combination with the Viterbi decoder, and the above architecture can be employed in the overall structure of the receiver. To obtain a systolic structure for the Viterbi algorithm, it can be formulated as a form of matrix multiplication. In the

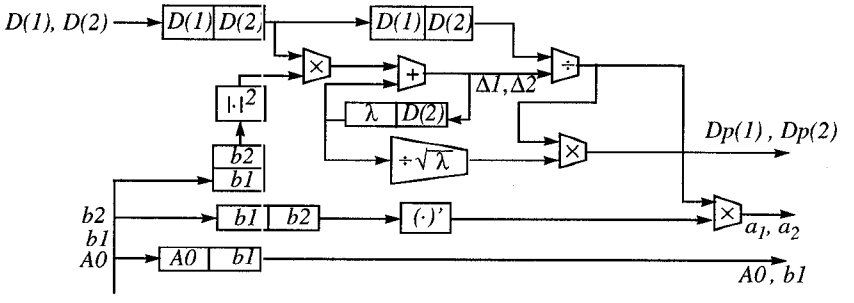

Fig. 6: The structure of the processor μ used in the RLS algorithm implementation found in Fig. 5

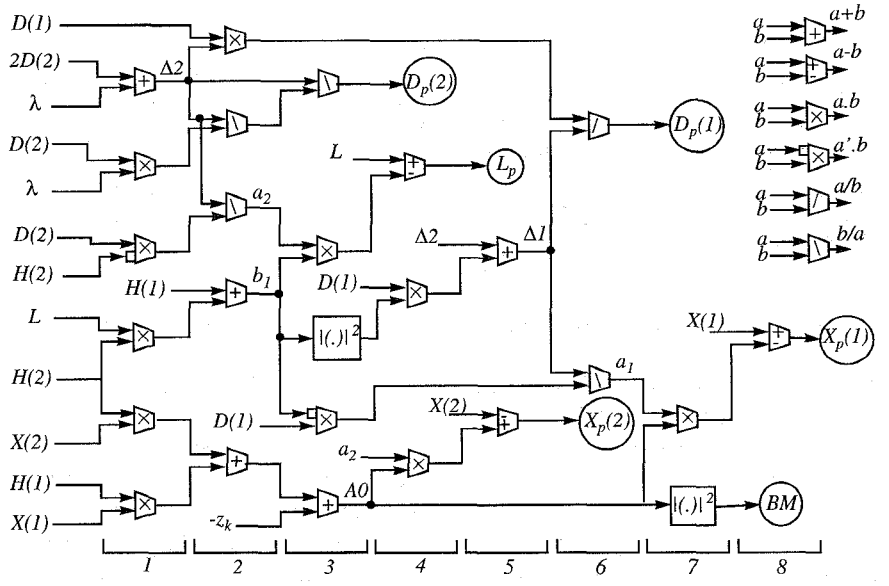

Fig. 7: The pipeline structure for implementing the RLS channel

Viterbi algorithm if the node value at $t=k$ is shown by $\Gamma^{[k]}$ and the branch metric from node i to node j is b_{ij}, then to obtain the node values at $t=k+1$, the Viterbi algorithm computes

$$\begin{bmatrix}\Gamma_1 & \Gamma_2 & \Gamma_3 & \Gamma_4\end{bmatrix}^{[k+1]} = \begin{bmatrix}\Gamma_1 & \Gamma_2 & \Gamma_3 & \Gamma_4\end{bmatrix}^{[k]} * \begin{bmatrix} b_{11} & b_{12} & \cdot & b_{14} \\ \cdot & \cdot & \cdot & \cdot \\ b_{41} & b_{42} & \cdot & b_{44}\end{bmatrix} \tag{22}$$

The operation ' * ' is not an ordinary multiplication and the elements of the left side vector can be written as

$$\Gamma_j^{[k+1]} = (\Gamma_1^{[k]} + b_{1j}) \otimes (\Gamma_2^{[k]} + b_{2j}) \otimes \ldots \otimes (\Gamma_4^{[k]} + b_{4j}) \tag{23}$$

The '+' operator is conventional addition, and the operation '⊗' denotes taking minimum. Hence (23) becomes

$$\Gamma_j^{[k+1]} = Min_{1 \le i \le 4}(\Gamma_i^{[k]} + b_{ij}) \tag{24}$$

Therefore it is possible to consider the systolic architectures for vector-matrix multiplications proposed in the literature [9] to implement the Viterbi algorithm, and the pipelined structure of Fig. 7 can be employed in a design for computation of the branch metrics based on the channel estimates.

The parallel structure in Fig. 8 is proposed for the joint implementation of the RLS estimator and the Viterbi algorithm. The buffer on top contains four sets of data for different trellis nodes. X is the estimated state, L and D are covariance factors, Γ is the node value in the Viterbi algorithm, and P contains the path information obtained in the Viterbi algorithm. All of the data sets are propagated to the four pipelined RLS estimators sequentially. Here we only need four pipeline

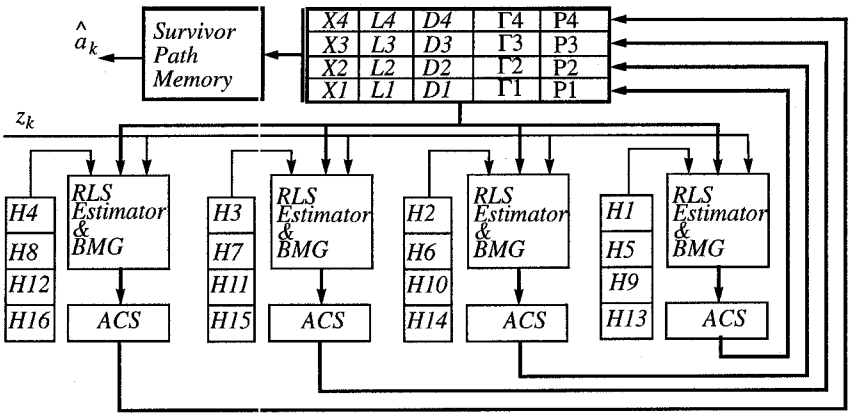

Fig. 8: The parallel architecture for implementation of the generalized Viterbi algorithm.

stages and the computation tasks in eight different stages of Fig. 7 can be merged to obtain four new stages with balanced latencies. Each estimator also receives the hypothesized transmitted data, H, related to the branches that end in the same node, and the received signal z_k. In four subsequent pipeline stages the channel estimates and the branch metrics are computed and the output of the Branch Metric Generator (BMG) is passed to the Add Compare Select (ACS) unit. The ACS unit adds the original node value and the computed branch metric, and selects the survivor path. The information of each surviving path, including the states of the channel estimator are passed to the top buffer to be used at the next symbol interval.

The above Viterbi algorithm with the RLS estimators was simulated as the receiver structure for the communication system of section 2. By employing the RLS estimation algorithm in this receiver the BER performance will be superior to the case of using the Least Mean Squares (LMS) estimator [3]. In fact, at a BER=10^{-3} we observed a 3 dB improvement with the RLS algorithm and by using the Kalman estimation method this improvement will be 7 dB. The mantissa length associated with the floating point operations could be reduced to 12 bits for the RLS algorithm with no change in the BER. This determines the minimum requirements in a digital hardware implementation.

5. CONCLUSION

In this paper we have considered the receiver structure for data communication over Rayleigh fading channels. The MLSE receiver estimates the channel impulse response to obtain branch metrics for the Viterbi algorithm. The estimation accuracy affects the overall BER performance of the receiver.

The Kalman filter and the RLS algorithm were introduced as channel estimators, and it was shown that the RLS algorithm is basically the same as the measurement update equations of the Kalman filter. Square-root filtering was employed to avoid numerical errors, and two algorithms were proposed for the measurement update and the time update equations of the Kalman filter. For the communication system

of section 2, the RLS algorithm yields a very simple implementation. Two parallel structures were proposed for the RLS algorithm, one of which can be used in a pipelined structure. Finally an overall structure was proposed for the receiver, including the Viterbi algorithm and the estimator.

References

[1] R. Raheli, A. Polydoros, and C. Tzou, "Per-survivor Processing: a general approach to MLSE in uncertain environments," *IEEE Transaction on Communications*, vol. 43, pp. 354-364, Feb/ Mar/Apr 1995.

[2] H. Kubo, K. Murakami, and T. Fujino,"An adaptive Maximum-likelihood sequence estimator for fast time-varying intersymbol interference channels," *IEEE Transactions on Communications*, Vol. 42, pp. 1872-1880, Feb/Mar/Apr 1994.

[3] M. Javad Omidi, S. Pasupathy, and P. Glenn Gulak, "Joint Data and Kalman Estimation of Fading Channel Using A Generalized Viterbi Algorithm," *Proceedings of IEEE International Conference on Communications*, 1996.

[4] J. M. Jover, and T. Kailath, "A parallel architecture for Kalman filter measurement update and parameter estimation," *Automatica*, vol. 22, No. 1, pp. 43-57, 1986.

[5] M. Bayoumi, P. Rao, and B. Alhalabi, "VLSI parallel architecture for Kalman filter an algorithm specific approach," *Journal of VLSI Signal Processing*, 4, pp. 147-163, Kluwer Academic Publishers, 1992.

[6] Q. Dai, and E. Shwedyk, "Detection of bandlimited signals over frequency selective Rayleigh Fading channels," *IEEE Trans on Communications*, vol. 42, pp. 941-950, Feb/ Mar/Apr 1994.

[7] B. D. O. Anderson, and J. B. Moore, "Optimal Filtering," *Prentice-Hall*, 1979.

[8] M. S. Grewal, and A. P. Andrews, "Kalman filtering, theory and practice," *Prentice-Hall*, 1993.

[9] S. Y. Kung, "VLSI array processors," *Prentice-Hall*, 1988.