計測自動制御学会東北支部第１６４回研究集会(1996.11.22)
資料番号１６４－４

FADDEEVAアルゴリズムにもとづく滞在時間最小型カルマンフィルタのＶＬＳＩアーキテクチャ

VLSI Architecture of Kalman Filters with A Least Latency based on FADDEEVA Algorithm

○高橋　強*，恒川　佳隆**，三浦　守**
Kyo Takahashi,Yoshitaka Tsunekawa,Mamoru Miura
*岩手県立高度技術専門学院，**岩手大学
* Iwate pref. high-tech college　** Iwate University

キーワード：カルマンフィルター（Kalman Filters），滞在時間（Latency），
Faddeevaアルゴリズム（Faddeeva Algorithm），シストリックアレー（Systolic Array）
連絡先：〒０２３　岩手県水沢市佐倉河字東広町６６－２　岩手県立高度技術専門学院
制御システム科　高橋　強,Tel.0197-22-4422，Fax.0197-23-6189

１．はじめに

カルマンフィルタは宇宙工学の分野で人工衛星の軌道計算に用いられて以来、コンピュータの発達とともに応用分野を拡げ、現在では通信工学、制御工学にとどまらず統計学など幅広い分野で応用されている[1]。しかし計算の複雑さから、実時間処理に対してはその適用範囲が一般に制限されてきた。近年進展のめざましいディジタルシグナルプロセッサ（Digital Signal Processor;DSP)を用いて低次数のカルマンフィルタを実現しても、実時間処理は非常に困難である。最近、処理速度を向上させる目的で、シストリックアレイを用いた研究が多くなされている[2,3]。ところがシストリックアレイでは、高い処理速度が得られる反面、滞在時間（Latency）は大きくなり、フィードバック制御に応用する場合、システムの安定性を損なうという問題がある。

本研究では、カルマンフィルタの滞在時間を最小化するために、Faddeevaアルゴリズム[2,5]を用いて処理を高度に並列化したＶＬＳＩアーキテクチャを提案する。本アーキテクチャでは、まずカルマンフィルタ本来の並列性に着目し、各パラメータ計算を可能な限り並列化している。つぎに、演算を並列化するためにFaddeevaアルゴリズムを適用し、特にカルマンゲインを高速に計算して滞在時間を大幅に減少させている。これは、処理の並列化とともに、本来、次数回分必要な処理時間のかかる除算を最後に１度だけ行えばよいことに着目し、滞在時間の増加を回避している。最後に滞在時間最小型カルマンフィルタの構成を示し、滞在時間を評価している。これにより滞在時間が非常に小さく、本アーキテクチャが有効性であることを示す。

２．カルマンフィルタ

次のような$k \geq 0$に対して定義される線形・有限次元の離散時間システムがある。

$$x_{k+1} = F_k x_k + G_k w_k \qquad (1.1)$$

$$y_k = H_k x_k + v_k \qquad (1.2)$$

ここで、x_kは状態ベクトル、y_k　は観測信号と呼ばれるそれぞれn次元、m次元のベクトルであり、w_kとv_kはそれぞれシステム雑音、観測雑音と呼ばれるl次元及びm次元のベクトルである。また、F_kを状態遷移行列、G_kを駆動行列、H_kを観測行列と呼び、一般に確定した$n\times n$、$n\times l$、$m\times n$行列である。また、w_kとv_kは互いに独立で、それぞれ平均値ベクトル零、既知の共分散行列Σw_kとΣv_kを持つガウス白色雑音であるとするとカルマンフィルタは次のように表される。

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(y_k - H_k \hat{x}_{k|k-1}) \qquad (2.1)$$

$$\hat{x}_{k+1|k} = F_k \hat{x}_{k|k} \tag{2.2}$$

$$K_k = \hat{\Sigma}_{k|k-1} H_k^T [H_k \hat{\Sigma}_{k|k-1} H_k^T + \Sigma v_k]^{-1} \tag{2.3}$$

$$\hat{\Sigma}_{k|k} = \hat{\Sigma}_{k|k-1} - K_k H_k \hat{\Sigma}_{k|k-1} \tag{2.4}$$

$$\hat{\Sigma}_{k+1|k} = F_k \hat{\Sigma}_{k|k} F_k^T + G_k \Sigma w_k G_k^T \tag{2.5}$$

ただし、初期値は、$\hat{x}_{0|-1} = \bar{x}_0, \hat{\Sigma}_{0|-1} = \Sigma x_0$ である[4]。

3．Faddeevaアルゴリズム

Faddeevaアルゴリズムは、行列の乗算や逆行列の計算を行列演算を直接計算せずに求める方法で、ガウスの消去法[5]を用いて行列を三角化及び零化することにより実行される。

3.1 Faddeevaアルゴリズムの概要

未知の行列 X を含む方程式 $AX = B$ があり、逆行列 A^{-1} を求めずに $CX + D$ を求める問題を考える。この過程を以下のように示す。

A	B
$-C$	D

A	B
$-C+WA$	$D+WB$

ここで、$W = CA^{-1}$ とすると、

$$D + WB = D + CA^{-1}B$$
$$= CX + D$$

となり、以下の様になる。

A	B
0	$CX+D$

以上は、左下行列を零化することで、同時に既知の行列 A、B、C、D から $CX + D$ が右下行列に得られることを示している。

このアルゴリズムの特徴は、前向き処理のみで実行可能なことである。このことはＶＬＳＩ実現において、データの流れを単純化でき、規則的にハードウエアを構成することが可能になる。

3.2 上三角化におけるピボット選択の検討

Faddeevaアルゴリズムにおいて左上行列の上三角化は、ガウスの消去法を用いて行うが、この場合、一般に対角要素が零になると計算が続けられない。そこで、ピボット選択を行い対角要素が零にならないように行の入れかえ操作を行う必要がある。3×3行列の例を図１に示す。ここで要素の右肩括弧は、何段目の処理に対する入力信号であるかを表す。

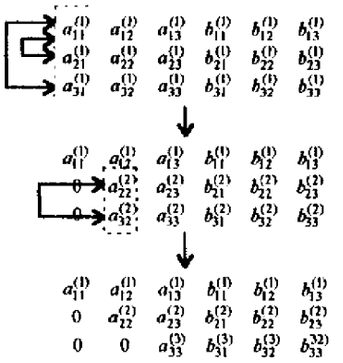

図１　ガウスの消去法におけるピボット選択

これら零判断と行の入れ替え操作は滞在時間を大幅に増加させ、しかもアーキテクチャを複雑にする。しかしカルマンフィルタに適用する場合、行列Ａに相当する式(2.3)の［］内は正則であるため零ピボットが現れない[1,6]。したがって、判断と行の入れ替え操作は不要であり、Faddeevaのアルゴリズムを適用することにより滞在時間の大幅な減少と、ＶＬＳＩアーキテクチャを単純で規則的な構成にすることが可能になる。

4．滞在時間最小型アーキテクチャ

4.1 パラメータ計算の並列化

滞在時間を減少させるために、まず式 (2) に示されるパラメータ計算を並列化する。図２に、カルマンフィルタのブロック図を示す。同図より $\hat{\Sigma}_{k|k}$ と $\hat{x}_{k|k}$ の計算、$\hat{\Sigma}_{k+1|k}$ と $\hat{x}_{k+1|k}$ の計算を並列に実行可能であることがわかる。

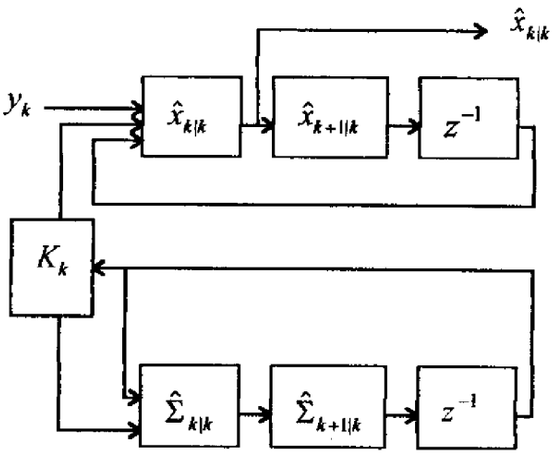

図２　カルマンフィルタのブロック図

さらに各計算にFaddeevaアルゴリズムを適用した場合、計算手順は以下のようになる。

- $K_k = \hat{\Sigma}_{k|k-1} H_k^T \left[H_k \hat{\Sigma}_{k|k-1} H_k^T + \Sigma v_k \right]^{-1}$

1. $\begin{array}{c|c} I & H_k^T \\ \hline -\hat{\Sigma}_{k|k-1} & 0 \end{array} \rightarrow \hat{\Sigma}_{k|k-1} H_k^T$ (4.1)

2. $\begin{array}{c|c} I & \hat{\Sigma}_{k|k-1} H_k^T \\ \hline -H_k & \Sigma v_k \end{array}$

$\rightarrow H_k \hat{\Sigma}_{k|k-1} H_k^T + \Sigma v_k$ (4.2)

3. $\begin{array}{c|c} H_k \hat{\Sigma}_{k|k-1} H_k^T + \Sigma v_k & I \\ \hline -H_k^T & 0 \end{array}$

$\rightarrow H_k^T \left[H_k \hat{\Sigma}_{k|k-1} H_k^T + \Sigma v_k \right]^{-1}$ (4.3)

4. $\begin{array}{c|c} I & H_k^T \left[H_k \hat{\Sigma}_{k|k-1} H_k^T + \Sigma v_k \right]^{-1} \\ \hline -\hat{\Sigma}_{k|k-1} & 0 \end{array}$

$\rightarrow K_k$ (4.4)

- $\hat{\Sigma}_{k|k} = \hat{\Sigma}_{k|k-1} - K_k H_k \hat{\Sigma}_{k|k-1}$

1. $\begin{array}{c|c} I & \hat{\Sigma}_{k|k-1} \\ \hline -H_k & 0 \end{array} \rightarrow H_k \hat{\Sigma}_{k|k-1}$ (4.5)

2. $\begin{array}{c|c} I & H_k \hat{\Sigma}_{k|k-1} \\ \hline K_k & \hat{\Sigma}_{k|k-1} \end{array} \rightarrow \hat{\Sigma}_{k|k}$ (4.6)

- $\hat{\Sigma}_{k+1|k} = F_k \hat{\Sigma}_{k|k} F_k^T + G_k \Sigma w_k G_k^T$

$= F_k \hat{\Sigma}_{k|k} F_k^T + \Sigma' w_k$

1. $\begin{array}{c|c} I & F_k^T \\ \hline -\hat{\Sigma}_{k|k} & 0 \end{array} \rightarrow \hat{\Sigma}_{k|k} F_k^T$ (4.7)

2. $\begin{array}{c|c} I & \hat{\Sigma}_{k|k} F_k^T \\ \hline -F_k & \Sigma' w_k \end{array} \rightarrow \hat{\Sigma}_{k+1|k}$ (4.8)

- $\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k \left(y_k - H_k \hat{x}_{k|k-1} \right)$

1. $\begin{array}{c|c} I & \hat{x}_{k|k-1} \\ \hline H_k & y_k \end{array} \rightarrow y_k - H_k \hat{x}_{k|k-1}$ (4.9)

2. $\begin{array}{c|c} I & y_k - H_k \hat{x}_{k|k-1} \\ \hline -K_k & \hat{x}_{k|k-1} \end{array} \rightarrow \hat{x}_{k|k}$ (4.10)

以上より、カルマンフィルタの処理は図３のようになり、Faddeevaアルゴリズムを用いることでカルマンゲインの一部も並列化できる。

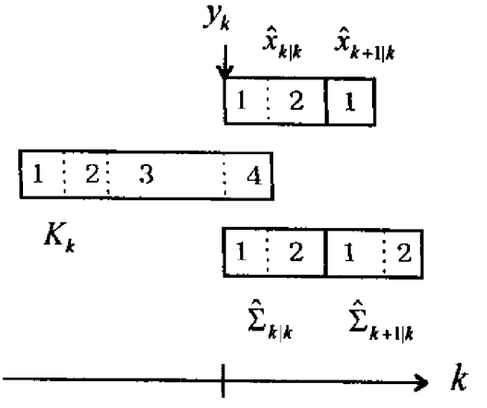

図３　計算の並列化

4.2　上三角化と零化の並列化

ここでは、Faddeevaアルゴリズムにより演算レベルでも更に並列化が可能であることを示す。また、データフローを用いることにより滞在時間を最小化することが可能であるため、ここではデータフローを用いることにする。

Faddeevaアルゴリズムにおいて、上三角化を行ってから零化を行うと滞在時間が増加してしまう。しかし、それらは同じ対角要素を用いて実行されることに着目すると、図４の様に処理を並列化することが可能になる。また本来、図４の各計算段階毎に除算が行われるが、除算は他の演算に比較して演算時間が数十倍かかるため[7]、これも滞在時間を大幅に増加させる原因になる。しかし、各計算段階で除算を行わずに係数とし残しておき、それらの分母をすべて乗算した後、一度だけ除算を実行することで同様の結果を得ることができる。更に、求めたいのは左下行列であるので、除算結果は行列Dに対して作用させるだけでよい。

以上より、各計算段階は図５に示すデータフローにより実行される場合、滞在時間が最小になる。

次に、カルマンゲイン以外の計算は左上行列が単位行列であるため、単なる行列の乗算になる。滞在時間が最小になるのは、図６に示すデータフローで計算される場合である。

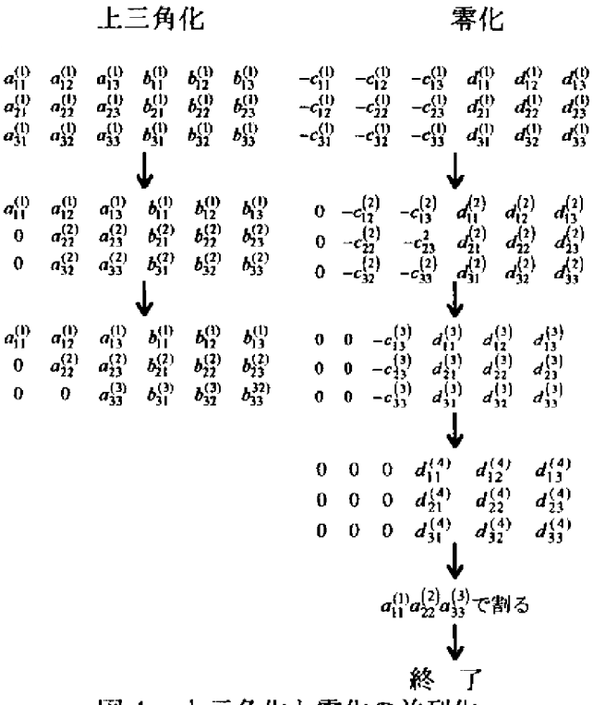

図４　上三角化と零化の並列化

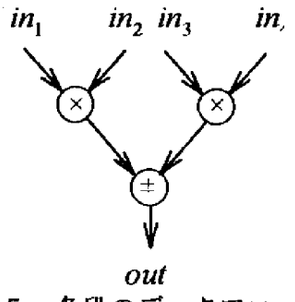

図５　各段のデータフロー

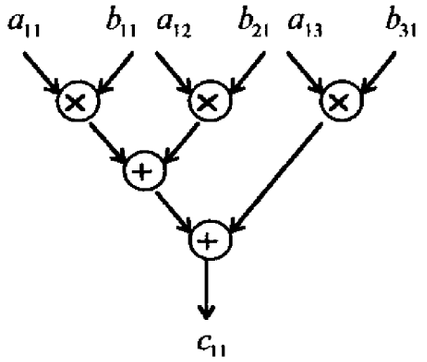

図６　行列乗算のデータフロー

５．実現法

前章で検討した滞在時間最小のデータフロー（図５、図６）を用いて、カルマンフィルタを構成する２種類の基本モジュールを構成する。

5.1　行列乗算の基本モジュール

前章で示したように、式（4.3）以外の計算は単なる行列の乗算となるため、図６のデータフローを適用した基本モジュールＢＭ１を用いる。ＢＭ１を図７に示す。これをカルマンフィルタの計算式に用いる場合、各パラメータを行列Ａ、Ｂ、Ｃに対応させて用いる。

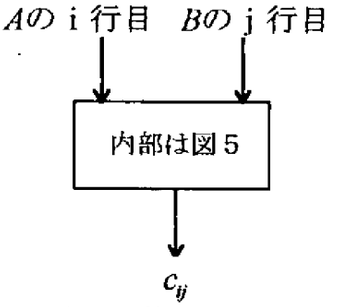

図７　行列乗算の基本モジュール
ＢＭ１
$C = AB$ を実行

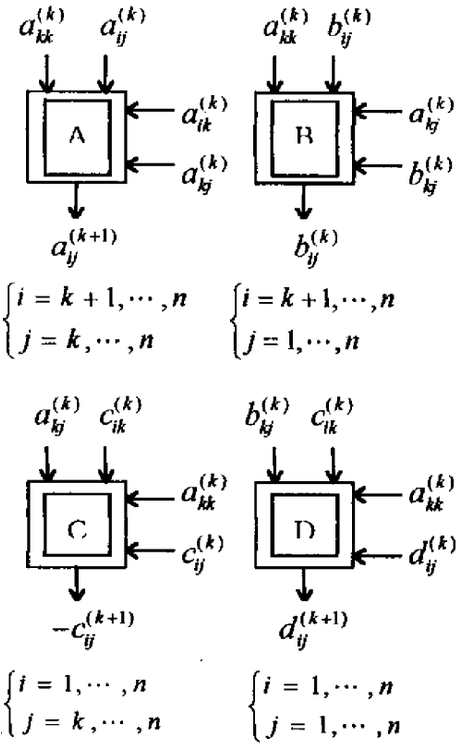

図８　Faddeevaアルゴリズムの
基本モジュールＢＭ２

5.2　上三角化と零化の基本モジュール

式（4.3）に対して用いる基本モジュールＢＭ２は、図５に示されるデータフローを用いて構成するとき滞在時間は最小になる。ＢＭ２を図８に示す。なお、行列Ａ、Ｂ、Ｃ、Ｄはそれぞれ、$H_k\hat{\Sigma}_{k|k-1}H_k^T+\Sigma v_k$、$I$、$H_k^T$、$0$に対応する。

5.3　滞在時間最小型カルマンフィルタの構成

これまで検討した基本モジュールＢＭ１、ＢＭ２を用いて、滞在時間最小型カルマンフィルタを構成する。図９に全体の構成を示す。ここで、ＰＢ１、ＰＢ２はプロセッシングブロックで、それらの構成を図１０、図１１に示す。なお、次数n,l,mを３とした。同図より提案アーキテクチャでは、ほぼ２種類の基本モジュールＢＭ１、ＢＭ２のみで構成されており、高次のカルマンフィルタに対しては、ＢＭ１、ＢＭ２を増やすだけで構成することができる。したがって、提案したアーキテクチャはモジュール性、規則性に優れＶＬＳＩ実現向きである。

６．滞在時間の評価

本アーキテクチャの滞在時間を評価し有効性を示す。まず、ＢＭ１、ＢＭ２の滞在時間を求め、それらをもちいて全体の滞在時間を求める。ここで、乗算の処理時間をt_m、加減算の処理時間をt_{as}、除算の処理時間をt_dとする。

［ＢＭ１の滞在時間］

$n\times n$行列に用いるＢＭ１の処理時間t_{BM1}は、並列化により乗算１回、加減算$log_2 n$回分の処理時間になり、

$$t_{BM1}=t_m+t_{as}log_2 n$$

となる。

［ＢＭ２の滞在時間］

$n\times n$行列に用いるＢＭ２の処理時間t_{BM2}は、並列化により乗算１回、加減算１回分の処理時間になり、

$$t_{BM2}=t_{as}+t_m$$

となる。

［ＰＢ１の滞在時間］

ＰＢ１はＢＭ１を並列に用いるため、滞在時間t_{PB1}はt_{BM1}に等しい。

［ＰＢ２の滞在時間］

ＰＢ２の滞在時間t_{PB2}は、ＰＢ２－＊のなかでは処理時間の最も大きいＰＢ２－Ｄの処理時間、除算１回、乗算１回分の処理時間になる。したがって、

$$t_{PB2}=t_d+n(t_m+t_{as})$$

になる。

以上より、図９に示されるカルマンフィルタ全体の滞在時間t_kは、

$$\begin{aligned}t_k&=t_{PB1}+6t_{PB2}\\&=t_d+(n+6)t_m+(n+6log_2 n)t_{as}\end{aligned}$$

になる。

図１２は、次数に対する滞在時間をＤＳＰ１個で実現した場合と比較した図である。なお、$t_{as}=50ns,t_m=50ns,t_d=2400ns$[7]とした。同図より、滞在時間は非常に小さい値となり、本アーキテクチャが滞在時間の減少に非常に有効であることがわかる。

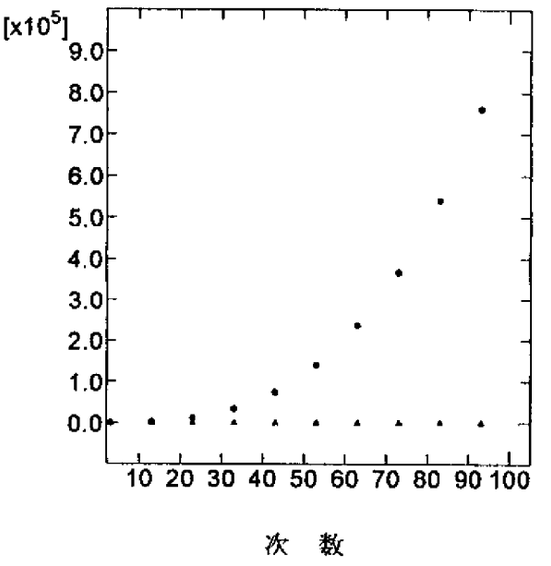

図１２　滞在時間の比較
▲：提案したアーキテクチャ
●：ＤＳＰ

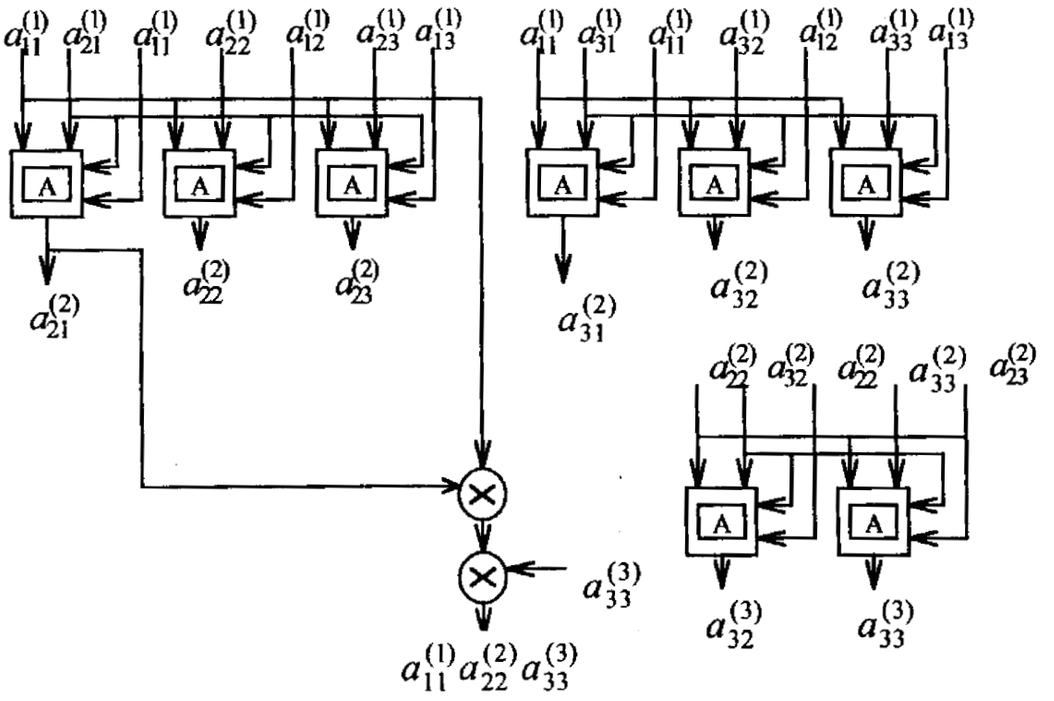

図１１（ｂ）　プロセッシングブロック２－Ａ
（ＰＢ２－Ａ）の構成

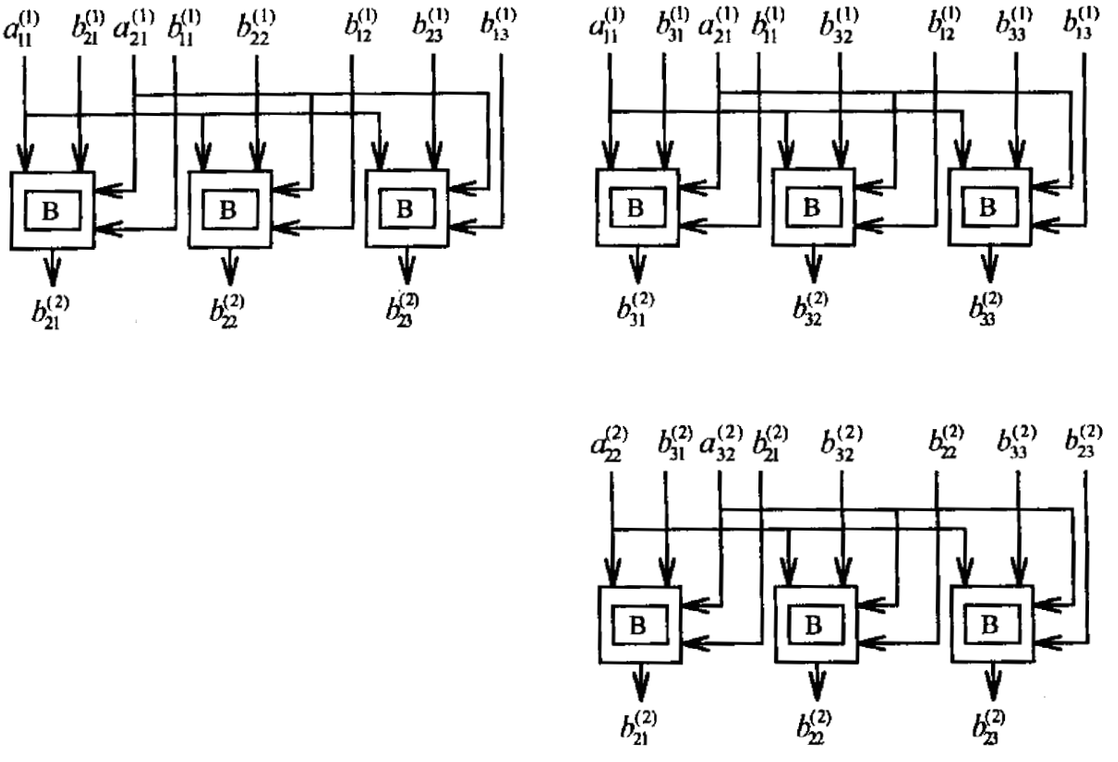

図１１（ｃ）　プロセッシングブロック２－Ｂ
（ＰＢ－２Ｂ）の構成

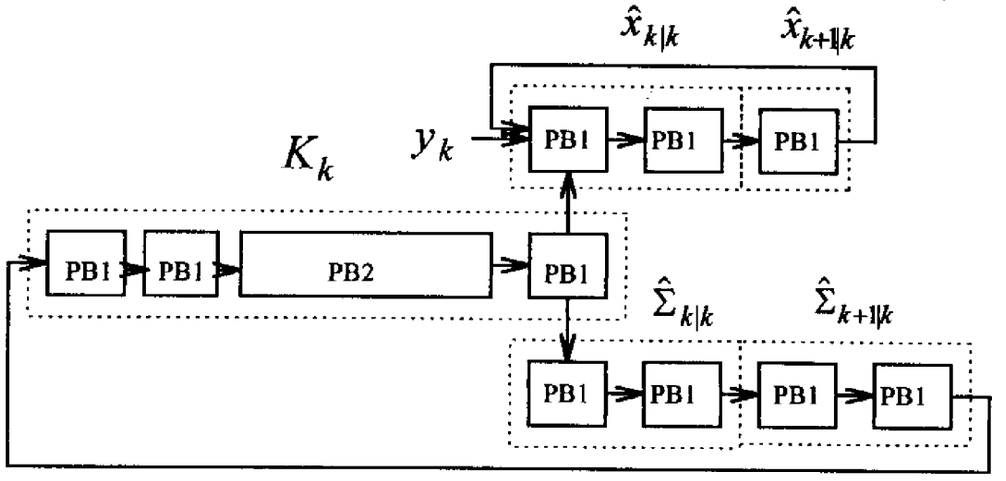

図９　滞在時間最小型カルマンフィルタの全体構成

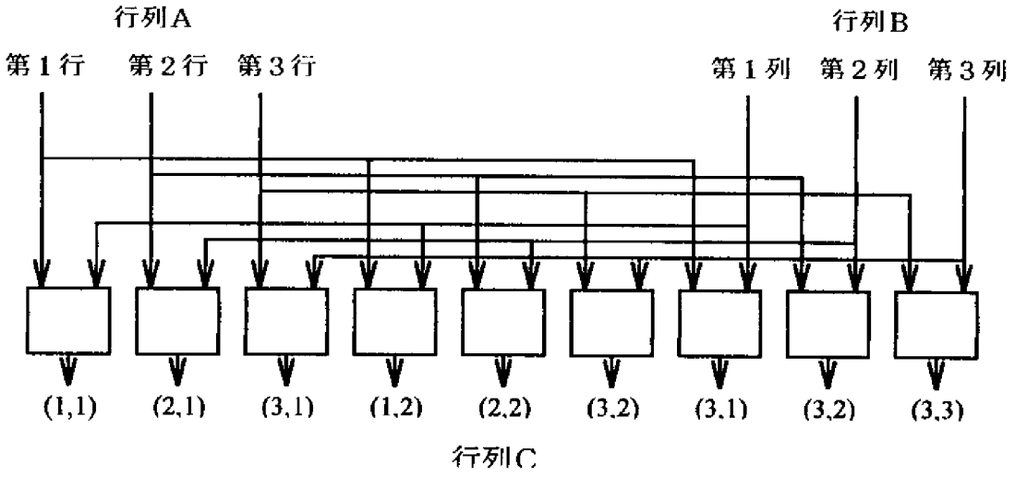

図１０　プロセッシングブロック（ＰＢ１）の構成
内部は基本モジュールＢＭ１（.）は行列の要素を表す

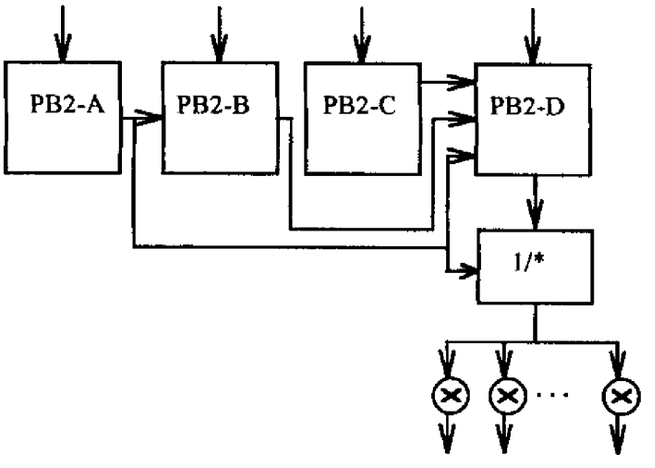

図11(a)　プロセッシングブロック１（ＰＢ１）の構成

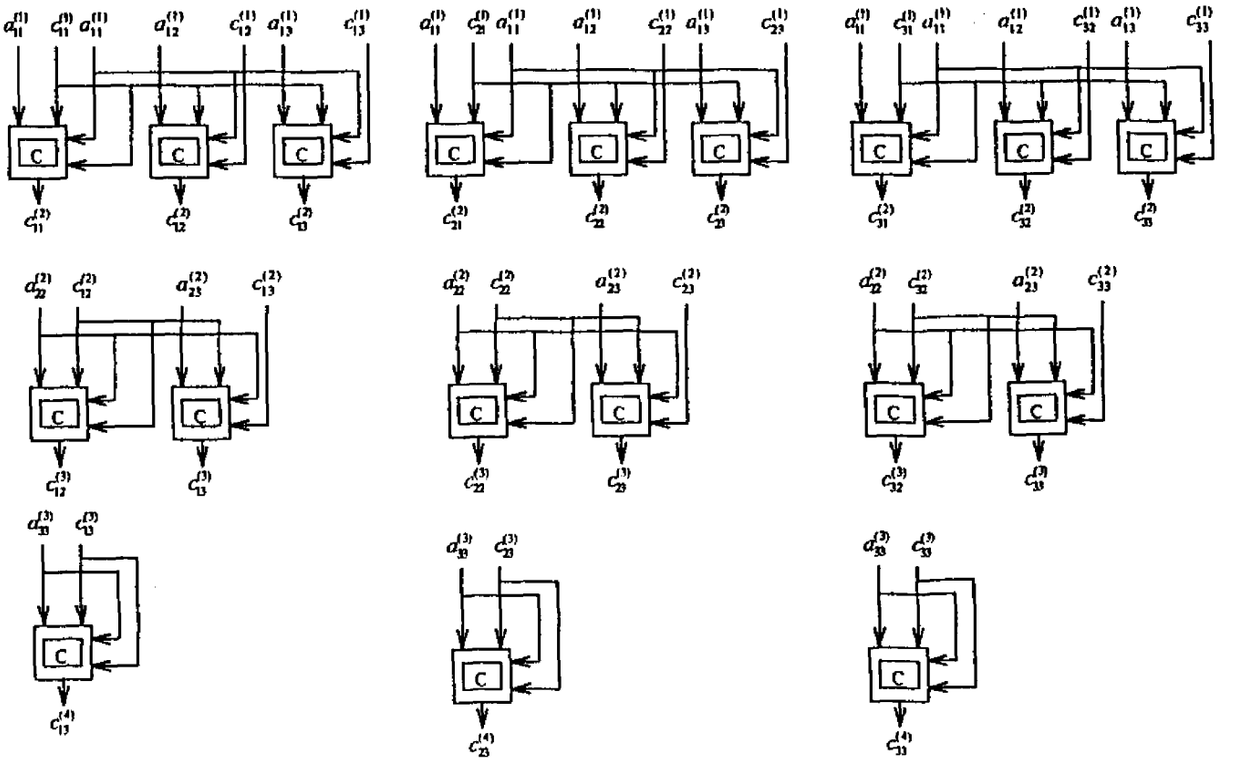

図１１(d) プロセッシングブロック２－Ｃ
（ＰＢ２－Ｃ）の構成

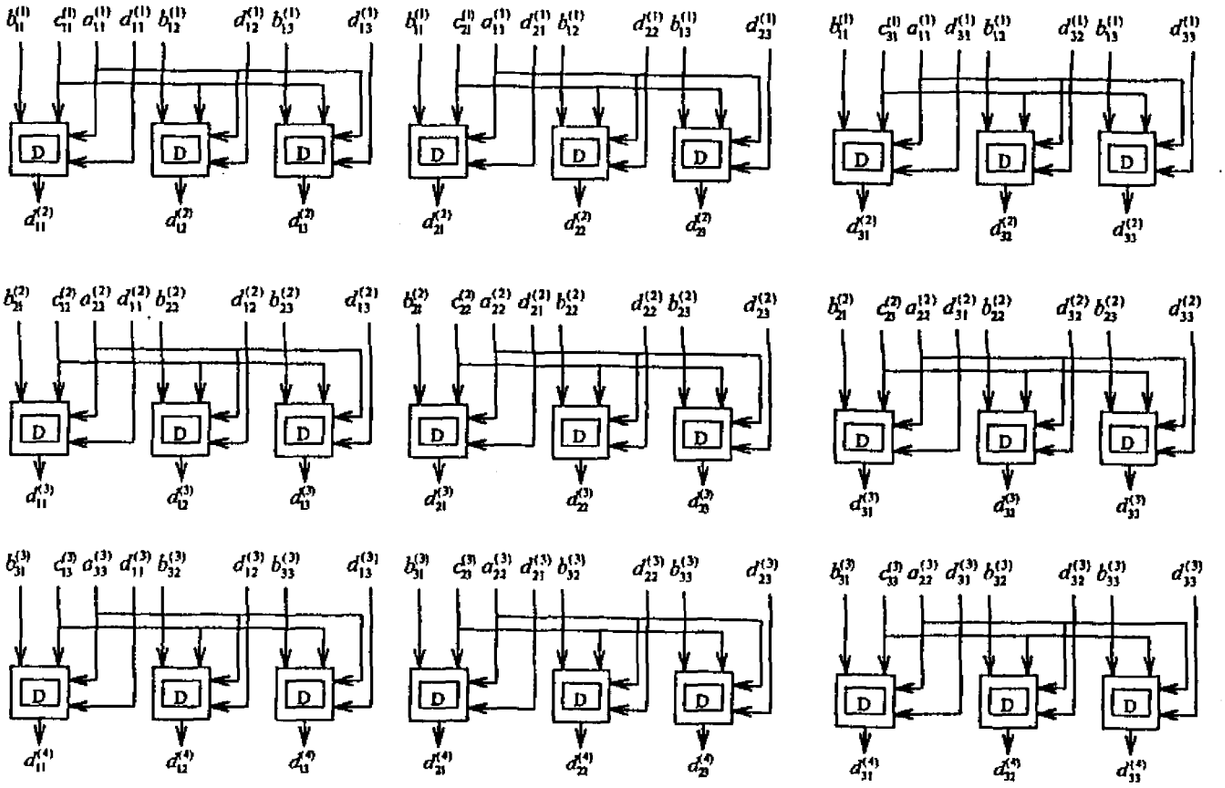

図１１(e) プロセッシングブロック２－Ｄ
（ＰＢ２－Ｄ）の構成

7.あとがき

本研究では、Faddeevaアルゴリズムを用いた滞在時間最小型カルマンフィルタのＶＬＳＩアーキテクチャを提案した。これにより、きわめて小さい滞在時間を達成できることがわかった。今後、ハードウエア設計を行い評価を進める予定である。

参考文献

1)片山　徹，“応用カルマンフィルタ”，朝倉書店

2)H.G.Yeh，“Systolic Implementation on Klman Filters”,IEEE Trans.,Acoust.Speech Signal Processiong vol.36,pp.1514-1517,Sept.1988

3)Kung,Hwang,“Systolic Array Design for Kalman Filtering”,IEEE Trans.,Signal Processing ,vol.39,No.1, Jan.1991

4)中野，西山，“パソコンで解くカルマンフィルタ”，丸善出版

5)G.M.Megson,“A Fast Faddeev Array”,IEEE Trans.Comp.,vol.41,No.12,Dec.1992

6)Gilbert Strang，“線形代数とその応用”，産業図書

7)TMS320c5X ユーザーズマニュアル，TEXAS INSTRUMENTS,1994