(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2010-39516
(P2010-39516A)
(43)公開日　平成22年2月18日(2010.2.18)

(51)Int.Cl.	F I		テーマコード（参考）
G06F 7/523 (2006.01)	G06F 7/523	A	5B016

審査請求 未請求　請求項の数 14　ＯＬ　（全 37 頁）

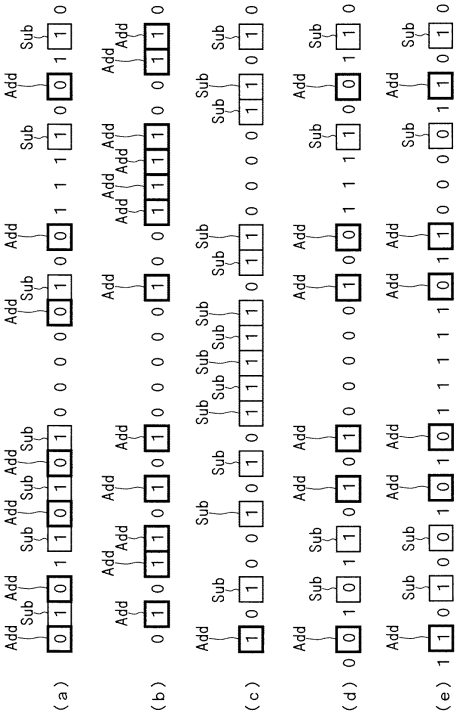
(21)出願番号　特願2008-198153 (P2008-198153)
(22)出願日　平成20年7月31日 (2008.7.31)

(71)出願人　000006633
京セラ株式会社
京都府京都市伏見区竹田鳥羽殿町６番地
(74)代理人　100088672
弁理士　吉竹　英俊
(74)代理人　100088845
弁理士　有田　貴弘
(72)発明者　小林　芳直
神奈川県大和市下鶴間１６２３－１４　株式会社京セラディスプレイ研究所大和事業所内

Ｆターム(参考)　5B016 AA01　BA06　EA04

(54)【発明の名称】演算装置、画像表示装置、および演算方法

(57)【要約】
【課題】乗算における加算および減算の最大回数を低減することで、演算装置の小型化および演算速度の向上を図ることができる技術を提供する。
【解決手段】二進法に係る被乗数と乗数との乗算を行う演算装置において、乗数におけるビットの数値の下桁側からの配列に応じて、加算基調の演算を行った後に、該加算基調の演算および減算基調の演算のうちの少なくとも一方の演算を順次に行うことを決定する。ここで、加算基調の演算では、乗数におけるビットの数値が下桁側から順に１、０となる場合に被乗数に係る加算を行い、下桁側から順に１、１となる場合に減算基調の演算に移行しつつ被乗数に係る減算を行う。また、減算基調の演算では、符号拡張乗数におけるビットの数値が下桁側から順に０、１となる場合に被乗数に係る減算を行い、下桁側から順に０、０となる場合に加算基調の演算に移行しつつ被乗数に係る加算を行う。
【選択図】図１０



【特許請求の範囲】
【請求項１】
二進法でそれぞれ表現される被乗数と乗数との乗算を行う演算装置であって、
前記乗数を記憶する乗数記憶部と、
前記乗数におけるビットの数値の下桁側からの配列に応じて、加算基調の演算を行った後に、該加算基調の演算および減算基調の演算のうちの少なくとも一方の演算を順次に行うことを決定する決定部と、
を備え、
前記加算基調の演算が、
前記配列が下桁側から順に１、０となる場合に前記被乗数に係る加算を行い、前記配列が下桁側から順に１、１となる場合に前記減算基調の演算に移行しつつ前記被乗数に係る減算を行う演算であり、
前記減算基調の演算が、
前記配列が下桁側から順に０、１となる場合に前記被乗数に係る減算を行い、前記配列が下桁側から順に０、０となる場合に前記加算基調の演算に移行しつつ前記被乗数に係る加算を行う演算であることを特徴とする演算装置。
【請求項２】
請求項１に記載の演算装置であって、
前記被乗数を記憶する被乗数記憶部と、
前記決定部による決定結果に応じた演算を行う演算部と、
を更に備えることを特徴とする演算装置。
【請求項３】
二進法でそれぞれ表現される被乗数と乗数との乗算を行う演算装置であって、
前記被乗数を記憶する被乗数記憶部と、
前記乗数におけるビットの数値の下桁側からの配列に応じて、加算基調の演算を行った後に、該加算基調の演算および減算基調の演算のうちの少なくとも一方の演算を順次に行う演算部と、
を備え、
前記加算基調の演算が、
前記配列が下桁側から順に１、０となる場合に前記被乗数に係る加算を行い、前記配列が下桁側から順に１、１となる場合に前記減算基調の演算に移行しつつ前記被乗数に係る減算を行う演算であり、
前記減算基調の演算が、
前記配列が下桁側から順に０、１となる場合に前記被乗数に係る減算を行い、前記配列が下桁側から順に０、０となる場合に前記加算基調の演算に移行しつつ前記被乗数に係る加算を行う演算であることを特徴とする演算装置。
【請求項４】
請求項２または請求項３に記載の演算装置であって、
前記演算部が、
前記被乗数の符号ビットを負論理で扱い且つその他のビットを正論理で扱うことで前記被乗数に係る加算を行い、前記被乗数の符号ビットを正論理で扱い且つその他のビットを負論理で扱うことで前記被乗数に係る減算を行うことで得られた演算結果に、所定の定数を加算することによって、前記被乗数と前記乗数との乗算結果を導出し、
前記所定の定数が、
前記被乗数に０を代入して、前記配列に応じて、前記加算基調の演算を行った後に、該加算基調の演算および前記減算基調の演算のうちの少なくとも一方の演算を行うことで得られる演算結果の２の補数であることを特徴とする演算装置。
【請求項５】
請求項２または請求項３に記載の演算装置であって、
前記演算部が、

前記被乗数の符号ビットを負論理で扱い且つその他のビットを正論理で扱うことで前記被乗数に係る加算を行い、前記被乗数の符号ビットを正論理で扱い且つその他のビットを負論理で扱うことで前記被乗数に係る減算を行うことで得られた演算結果に、所定の定数を加算することによって、前記被乗数と前記乗数との乗算結果を導出し、
前記所定の定数が、
前記演算部によって前記加算および減算を行う際に負論理で扱われたビットに係る項に１を代入し、正論理で扱われたビットに係る項に０を代入した上で加算した結果の２の補数であることを特徴とする演算装置。
【請求項６】
二進法でそれぞれ表現される被乗数と乗数との乗算を行う演算装置であって、
前記乗数を記憶する乗数記憶部と、
前記乗数におけるビットの数値の下桁側からの配列を認識して、前記配列が下桁側から順に０が連続した後に１、０となる場合に前記被乗数に係る加算を行い、前記配列が下桁側から順に０が連続した後に１、１となる場合に前記被乗数に係る減算を行い、前記配列が下桁側から順に１が連続した後に０、１となる場合に前記被乗数に係る減算を行い、前記配列が下桁側から順に１が連続した後に０、０となる場合に前記被乗数に係る加算を行うことを決定する決定部と、
を備えることを特徴とする演算装置。
【請求項７】
請求項６に記載の演算装置であって、
前記被乗数を記憶する被乗数記憶部と、
前記決定部による決定結果に応じた演算を行う演算部と、
を更に備えることを特徴とする演算装置。
【請求項８】
二進法でそれぞれ表現される被乗数と乗数との乗算を行う演算装置であって、
前記被乗数を記憶する被乗数記憶部と、
前記乗数におけるビットの数値の配列が下桁側から順に０が連続した後に１、０となる場合に前記被乗数に係る加算を行い、前記配列が下桁側から順に０が連続した後に１、１となる場合に前記被乗数に係る減算を行い、前記配列が下桁側から順に１が連続した後に０、１となる場合に前記被乗数に係る減算を行い、前記配列が下桁側から順に１が連続した後に０、０となる場合に前記被乗数に係る加算を行う演算部と、
を備えることを特徴とする演算装置。
【請求項９】
請求項７または請求項８に記載の演算装置であって、
前記演算部が、
前記被乗数の符号ビットを負論理で扱い且つその他のビットを正論理で扱うことで前記被乗数に係る加算を行い、前記被乗数の符号ビットを正論理で扱い且つその他のビットを負論理で扱うことで前記被乗数に係る減算を行うことで得られた演算結果に、所定の定数を加算することによって、前記被乗数と前記乗数との乗算結果を導出し、
前記所定の定数が、
前記演算部によって前記加算および減算を行う際に負論理で扱われたビットに係る項に１を代入し、正論理で扱われたビットに係る項に０を代入した上で加算した結果の２の補数であることを特徴とする演算装置。
【請求項１０】
請求項２、請求項３、請求項７、及び請求項８の何れかに記載の演算装置であって、
前記演算部が、
前記被乗数の符号拡張を行いつつ前記被乗数に係る加算および減算を行うことで、前記被乗数と前記乗数との乗算結果を導出することを特徴とする演算装置。
【請求項１１】
請求項１から請求項１０の何れかに記載の演算装置を用いて入力画像信号を表示用画像

信号に変換する変換部と、
前記表示用画像信号に基づいて画像を表示する表示部と、
を備えることを特徴とする画像表示装置。
【請求項１２】
請求項１１に記載の画像表示装置であって、
前記変換部が、
明度および色差を示す信号を所定のルールに従って各色の階調を示す信号に変換するための乗算に、請求項１から請求項１０の何れかに記載の演算装置を用いることを特徴とする画像表示装置。
【請求項１３】
演算装置において二進法でそれぞれ表現される被乗数と乗数との乗算を行う演算方法であって、
(ａ)加算基調の演算を行うステップと、
(ｂ)前記(ａ)ステップの後に、前記乗数におけるビットの数値の下桁側からの配列に応じて、前記加算基調の演算および減算基調の演算のうちの少なくとも一方の演算を順次に行うステップと、
を備え、
前記加算基調の演算が、
前記配列が下桁側から順に１、０となる場合に前記被乗数に係る加算を行い、前記配列が下桁側から順に１、１となる場合に前記減算基調の演算に移行しつつ前記被乗数に係る減算を行う演算であり、
前記減算基調の演算が、
前記配列が下桁側から順に０、１となる場合に前記被乗数に係る減算を行い、前記配列が下桁側から順に０、０となる場合に前記加算基調の演算に移行しつつ前記被乗数に係る加算を行う演算であることを特徴とする演算方法。
【請求項１４】
演算装置において二進法でそれぞれ表現される被乗数と乗数との乗算を行う演算方法であって、
(ａ)前記乗数におけるビットの数値の配列が下桁側から順に０が連続した後に１、０となる場合に前記被乗数に係る加算を行うステップと、
(ｂ)前記配列が下桁側から順に０が連続した後に１、１となる場合に前記被乗数に係る減算を行うステップと、
(ｃ)前記配列が下桁側から順に１が連続した後に０、１となる場合に前記被乗数に係る減算を行うステップと、
(ｄ)前記配列が下桁側から順に１が連続した後に０、０となる場合に前記被乗数に係る加算を行うステップと、
を備えることを特徴とする演算方法。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、被乗数と乗数との乗算を行う演算装置、および演算方法、ならびにその演算装置を利用した画像表示装置に関する。
【背景技術】
【０００２】
マトリックス状にドットを表示する画像表示装置としては、例えば、有機ＥＬディスプレイや液晶ディスプレイなどがある。このようなディスプレイにおいて、ＮＴＳＣ（National Television Standards Committee）の規格に基づく画像信号を可視的に出力する場合には、まず、ＩＴＵ－Ｒ（国際電気通信連合無線通信部門）の勧告ＢＴ６０１で推奨されている符号化パラメータなどを用いた上で、ＹＣ分離（輝度信号と色差信号とを高精度に分離する技術）が行われてＹ，Ｃｒ，Ｃｂの値が抽出される。そして、所定のアルゴリ

ズムに従って、Ｙ，Ｃｒ，Ｃｂの値が適宜被乗数とされて所定の定数（すなわち乗数）が乗じられることで加算値および減算値が算出され、Ｒ，Ｇ，Ｂの値が導出される（例えば、特許文献１など）。

【０００３】

例えば、Ｙ，Ｃｒ，Ｃｂの値から下式(１)～(３)で示す所定の変換式によって、Ｒ，Ｇ，Ｂの値が算出される。

【０００４】

Ｒ＝Ｙ×(定数１)＋Ｃｒ×(定数２)＋(定数３)　・・・(１)

Ｇ＝Ｙ×(定数１)＋Ｃｂ×(定数４)＋Ｃｒ×(定数５)＋(定数６)　・・・(２)

Ｂ＝Ｙ×(定数１)＋Ｃｂ×(定数７)＋(定数８)　・・・(３)。

【０００５】

そして、一般的な演算装置では二進法が用いられ、更に、定数には負の値が存在するため、Ｃｒ，Ｃｂの値は負の数となることもある。このような二進法で表現される負の値を含む数値であるＹ，Ｃｒ，Ｃｂの値と乗数との積を求めるアルゴリズムとしては、ブースの乗算アルゴリズムや、定数加算法による乗算アルゴリズムなどがある（例えば、特許文献２など）。

【０００６】

例えば、ブースの乗算アルゴリズムは、乗数の下の桁から順に１の次に０となる場合は被乗数の加算を行い、乗数の下の桁から順に０の次に１となる場合は被乗数の減算を行うことで、乗算を実現するものである。ここで、ブースの乗算アルゴリズムについて簡単に説明する。

【０００７】

図２６は、ブースの乗算アルゴリズムの演算処理フローを示すフローチャートである。まず、乗数の最下位に０が付加され（ステップＳＳ１）、カウント値ｎが初期値である１に設定され（ステップＳＳ２）、乗数の下位からｎ＋１番目とｎ番目のビットのパターンが"０１"であるか否か判定される（ステップＳＳ３）。ここで、ビットのパターンが"０１"であれば、被乗数が符号拡張された上で加算されて（ステップＳＳ４）、ステップＳＳ７に進み、ビットのパターンが"０１"でなければ、乗数の下位からｎ＋１番目とｎ番目のビットのパターンが"１０"であるか否か判定される（ステップＳＳ５）。ここで、ビットのパターンが"１０"であれば、被乗数が符号拡張された上で減算されて（ステップＳＳ６）、ステップＳＳ７に進み、ビットのパターンが"０１"でなければ、ステップＳＳ７に進む。そして、ステップＳＳ７では、乗数の最上位までビットのパターンの判定が終了したか否か判定される。ここで、ビットのパターンの判定が終了していなければ、被乗数が左（すなわち上位側）に１ビット分シフトされつつ、カウント値ｎが１加算されて、ステップＳＳ３に進む。一方、ビットのパターンの判定が終了していれば、本動作フローが終了される。

【０００８】

図２７は、十進法で表された数値に対応する二進法の４ビットの整数表記を示す図である。具体的には、図２７では、十進法で表された数値（－８，・・・，＋７）に対して、二進法の４ビット整数表記（１０００，・・・，０１１１）がそれぞれ対応付けられている。

【０００９】

図２８は、十進法で５×(－６)と表される計算をブースの乗算アルゴリズムによって演算する例を説明するための図である。十進法の５は二進法では"０１０１"と示され、十進法の(－６)は二進法では"１０１０"と示される。ここでは、まず、乗数"１０１０"の最下位に０が付加されて"１０１００"とされ、下から２桁ずつビットのパターンが判定されると、まず、ビットのパターンが"００"であるために加減算が行われず、次に２の位に移って、ビットのパターンが"１０"であるために被乗数が符号拡張されて減算される。つまり、被乗数を２倍した"０１０１０"が符号拡張されて"０００１０１０"となり、その２の補数"１１１０１１０"が加算される。次に４の位に移って、ビットのパターンが"０１"

であるために被乗数が符号拡張されて加算される。つまり、被乗数を４倍した"０１０１００"が符号拡張されて"０００１０１００"とされて加算される。更に８の位に移って、ビットのパターンが"１０"であるために被乗数が符号拡張されて減算される。つまり、被乗数を８倍した"０１０１０００"が符号拡張されて"００１０１０００"となり、その２の補数"１１０１１０００"が加算される。そして、演算結果は"１１１０００１０"となる。換言すれば十進法の(－３０)＝（－2^7＋2^6＋2^5＋2^1）が求まる。

【００１０】

また、定数加算法による乗数アルゴリズムは、乗数に含まれる１の値の数だけ被乗数の加算を行うものである。ここで、定数加算法による乗数アルゴリズムについて簡単に説明する。

【００１１】

図２９および図３０は、定数加算法の乗算アルゴリズムの演算処理フローを示すフローチャートである。まず、カウント値ｎが初期値である１に設定され（ステップＳＰ１）、乗数の下位からｎ番目のビットが符号ビット、すなわち最上位ビット（ＭＳＢ：Most Significant Bit）であるか否か判定される（ステップＳＰ２）。ここで、乗数の下位からｎ番目のビットが符号ビットであれば、ステップＳＰ６に進み、符号ビットでなければステップＳＰ３に進む。ステップＳＰ３では、乗数の下位からｎ番目のビットが"１"であるか否か判定される。ここでは、乗数の下位からｎ番目のビットが"１"であれば、被乗数の加算（ここでは、符号ビットは負論理、それ以外のビットは正論理の加算）が行われて（ステップＳＰ４）、ステップＳＰ５に進み、乗数の下位からｎ番目のビットが"１"でなければステップＳＰ５に進む。また、ステップＳＰ５では、被乗数が左（すなわち上位側）に１ビット分シフトされつつ、カウント値ｎが１加算されて、ステップＳＰ２に進む。

【００１２】

また、ステップＳＰ６では、乗数の下位からｎ番目のビットが"１"であるか否か判定される。ここでは、ビットが"１"であれば、被乗数の減算（但し、符号ビットは正論理、それ以外のビットは負論理の加算）が行われて（ステップＳＰ７）、ステップＳＰ８に進み、乗数の下位からｎ番目のビットが"１"でなければステップＳＰ８に進む。ステップＳＰ８では、図３０で示す処理が行われることで、定数が算出されて、ステップＳＰ１～ＳＰ７の処理によって導出された値にステップＳＰ８で算出された定数が加算されて（ステップＳＰ９）、本動作フローが終了される。なお、図３０では、ステップＳＰ１～ＳＰ７における加減算時に負論理であった項に１を代入するとともに、正論理であった項に０を代入して、加算を行い（ステップＳＰ８１）、その加算結果の２の補数を算出することで（ステップＳＰ８２）、定数が算出される。

【００１３】

図３１は、十進法で５×(－６)と表される計算を定数加算法の乗算アルゴリズムによって演算する例を説明するための図である。まず、上記ステップＳＰ１～ＳＰ７において、図３１(ａ)で示すような加算が行われて、加算結果"０１０１０１０"が得られる。また、上記ステップＳＰ８１において、図３１(ｂ)で示すように、加減算時に負論理であった項に１が代入されるとともに、正論理であった項に０が代入されて、加算が行われ、加算結果"１００１０００"が得られる。そして、上記ステップＳＰ８２において、上記ステップＳＰ８１で得られた加算結果"１００１０００"の２の補数が導出されて、図３１(ｃ)の２行目に記述される定数"１０１１１０００"が得られる。そして、図３１(ｃ)で示すように、上記ステップＳＰ９において、上記ステップＳＰ１～ＳＰ７で得られた加算結果０１０１０１０に、ステップＳＰ８で得られた定数"１０１１１０００"が加算される。そして、この演算結果は"１１１０００１０"となる。つまり、十進法の(－３０)＝（－2^7＋2^6＋2^5＋2^1）が求まる。

【００１４】

なお、定数加算法による乗数アルゴリズムには、バリエーションとして、２の補数の定数加算法による乗数アルゴリズムがある。この乗数アルゴリズムについても簡単に説明する。

【００１５】
図３２および図３３は、２の補数の定数加算法の乗算アルゴリズムの演算処理フローを示すフローチャートである。まず、乗数の２の補数を算出して仮の乗数（仮乗数）とする（ステップＳＴ１）。そして、カウント値ｎが初期値である１に設定され（ステップＳＴ２）、仮乗数の下位からｎ番目のビットが符号ビットであるか否か判定される（ステップＳＴ３）。ここで、仮乗数の下位からｎ番目のビットが符号ビットであれば、ステップＳＴ７に進み、符号ビットでなければステップＳＴ４に進む。ステップＳＴ４では、仮乗数の下位からｎ番目のビットが"１"であるか否か判定される。ここでは、ビットが"１"であれば、被乗数の減算（ここでは、符号ビットは正論理、それ以外のビットは負論理の加算）が行われて（ステップＳＴ５）、ステップＳＴ６に進み、仮乗数の下位からｎ番目のビットが"１"でなければステップＳＴ６に進む。また、ステップＳＴ６では、被乗数が左（すなわち上位側）に１ビット分シフトされつつ、カウント値ｎが１加算されて、ステップＳＴ３に進む。
【００１６】
また、ステップＳＴ７では、仮乗数の下位からｎ番目のビットが"１"であるか否か判定される。ここでは、仮乗数の下位からｎ番目のビットが"１"であれば、被乗数の加算（ここでは、符号ビットは負論理、それ以外のビットは正論理の加算）が行われて（ステップＳＴ８）、ステップＳＴ９に進み、仮乗数の下位からｎ番目のビットが"１"でなければステップＳＴ９に進む。ステップＳＴ９では、図３３で示す処理が行われることで、定数が算出されて、ステップＳＴ１～ＳＴ８の処理によって導出された値にステップＳＴ９で算出される定数が加算されて（ステップＳＴ１０）、本動作フローが終了される。なお、図３３では、ステップＳＴ１～ＳＴ８における加減算時に負論理であった項に１を代入するとともに、正論理であった項に０を代入して、加算を行い（ステップＳＴ９１）、その加算結果の２の補数を算出することで（ステップＳＴ９２）、定数が算出される。
【００１７】
図３４は、十進法で５×(－６)と表される計算を２の補数の定数加算法の乗算アルゴリズムによって演算する例を説明するための図である。まず、上記ステップＳＴ１において、図３４(ａ)で２行目に示す乗数"１０１０"の２の補数"０１１０"が仮乗数として求められる。次に、上記ステップＳＴ２～ＳＴ８において、図３４(ａ)で示すような加算が行われて、加算結果"００１１００"が得られる。また、上記ステップＳＴ９１において、図３４(ｂ)で示すように、加減算時に負論理であった項に１を代入するとともに、正論理であった項に０を代入して、加算が行われ、加算結果"１０１０１０"が得られる。そして、上記ステップＳＴ９２において、上記ステップＳＴ９１で得られた加算結果"１０１０１０"の２の補数が導出されて、図３４(ｃ)の２行目に記述される定数"１１０１０１１０"が得られる。そして、図３４(ｃ)で示すように、上記ステップＳＴ１０において、上記ステップＳＴ２～ＳＴ８で得られた加算結果"００１１００"に、ステップＳＴ９で得られた定数"１１０１０１１０"が加算される。そして、この演算結果は"１１１０００１０"となる。つまり、十進法の(－３０)＝（$-2^7+2^6+2^5+2^1$）が求まる。
【００１８】
【特許文献１】特開２００７－６０４３７号公報
【特許文献２】特開２００２－２２９７７３号公報
【発明の開示】
【発明が解決しようとする課題】
【００１９】
しかしながら、ブースの乗算アルゴリズムでは、乗数に"０１０１"が繰り返して出現する場合には、加減算の演算回数が多くなり、定数加算法による乗算アルゴリズムでは、乗数に"１"が数多く出現する場合には、加算の演算回数が多くなる。また、２の補数の定数加算法による乗算アルゴリズムでも、乗数の２の補数に"１"が数多く出現する場合には、加減算の演算回数が多くなる。そして、演算装置においては、加算および減算の最大回数に合わせて性能およびサイズが決定される。このため、より演算回数が少ない乗算アルゴ

リズムが求められる。

【００２０】

本発明は、上記課題に鑑みてなされたものであり、乗算における加算および減算の最大回数を低減することで、演算装置の小型化および演算速度の向上を図ることができる技術を提供することを目的とする。

【課題を解決するための手段】

【００２１】

上記の課題を解決するために、請求項１の発明は、二進法でそれぞれ表現される被乗数と乗数との乗算を行う演算装置であって、前記乗数を記憶する乗数記憶部と、前記乗数におけるビットの数値の下桁側からの配列に応じて、加算基調の演算を行った後に、該加算基調の演算および減算基調の演算のうちの少なくとも一方の演算を順次に行うことを決定する決定部と、備え、前記加算基調の演算が、前記配列が下桁側から順に１、０となる場合に前記被乗数に係る加算を行い、前記配列が下桁側から順に１、１となる場合に前記減算基調の演算に移行しつつ前記被乗数に係る減算を行う演算であり、前記減算基調の演算が、前記配列が下桁側から順に０、１となる場合に前記被乗数に係る減算を行い、前記配列が下桁側から順に０、０となる場合に前記加算基調の演算に移行しつつ前記被乗数に係る加算を行う演算であることを特徴とする。

【００２２】

また、請求項２の発明は、請求項１に記載の演算装置であって、前記被乗数を記憶する被乗数記憶部と、前記決定部による決定結果に応じた演算を行う演算部と、を更に備えることを特徴とする。

【００２３】

また、請求項３の発明は、二進法でそれぞれ表現される被乗数と乗数との乗算を行う演算装置であって、前記被乗数を記憶する被乗数記憶部と、前記乗数におけるビットの数値の下桁側からの配列に応じて、加算基調の演算を行った後に、該加算基調の演算および減算基調の演算のうちの少なくとも一方の演算を順次に行う演算部と、を備え、前記加算基調の演算が、前記配列が下桁側から順に１、０となる場合に前記被乗数に係る加算を行い、前記配列が下桁側から順に１、１となる場合に前記減算基調の演算に移行しつつ前記被乗数に係る減算を行う演算であり、前記減算基調の演算が、前記配列が下桁側から順に０、１となる場合に前記被乗数に係る減算を行い、前記配列が下桁側から順に０、０となる場合に前記加算基調の演算に移行しつつ前記被乗数に係る加算を行う演算であることを特徴とする。

【００２４】

また、請求項４の発明は、請求項２または請求項３に記載の演算装置であって、前記演算部が、前記被乗数の符号ビットを負論理で扱い且つその他のビットを正論理で扱うことで前記被乗数に係る加算を行い、前記被乗数の符号ビットを正論理で扱い且つその他のビットを負論理で扱うことで前記被乗数に係る減算を行うことで得られた演算結果に、所定の定数を加算することによって、前記被乗数と前記乗数との乗算結果を導出し、前記所定の定数が、前記被乗数に０を代入して、前記配列に応じて、前記加算基調の演算を行った後に、該加算基調の演算および前記減算基調の演算のうちの少なくとも一方の演算を行うことで得られる演算結果の２の補数であることを特徴とする。

【００２５】

また、請求項５の発明は、請求項２または請求項３に記載の演算装置であって、前記演算部が、前記被乗数の符号ビットを負論理で扱い且つその他のビットを正論理で扱うことで前記被乗数に係る加算を行い、前記被乗数の符号ビットを正論理で扱い且つその他のビットを負論理で扱うことで前記被乗数に係る減算を行うことで得られた演算結果に、所定の定数を加算することによって、前記被乗数と前記乗数との乗算結果を導出し、前記所定の定数が、前記演算部によって前記加算および減算を行う際に負論理で扱われたビットに係る項に１を代入し、正論理で扱われたビットに係る項に０を代入した上で加算した結果の２の補数であることを特徴とする。

【００２６】
また、請求項６の発明は、二進法でそれぞれ表現される被乗数と乗数との乗算を行う演算装置であって、前記乗数を記憶する乗数記憶部と、前記乗数におけるビットの数値の下桁側からの配列を認識して、前記配列が下桁側から順に０が連続した後に１、０となる場合に前記被乗数に係る加算を行い、前記配列が下桁側から順に０が連続した後に１、１となる場合に前記被乗数に係る減算を行い、前記配列が下桁側から順に１が連続した後に０、１となる場合に前記被乗数に係る減算を行い、前記配列が下桁側から順に１が連続した後に０、０となる場合に前記被乗数に係る加算を行うことを決定する決定部と、を備えることを特徴とする。
【００２７】
また、請求項７の発明は、請求項６に記載の演算装置であって、前記被乗数を記憶する被乗数記憶部と、前記決定部による決定結果に応じた演算を行う演算部と、を更に備えることを特徴とする。
【００２８】
また、請求項８の発明は、二進法でそれぞれ表現される被乗数と乗数との乗算を行う演算装置であって、前記被乗数を記憶する被乗数記憶部と、前記乗数におけるビットの数値の配列が下桁側から順に０が連続した後に１、０となる場合に前記被乗数に係る加算を行い、前記配列が下桁側から順に０が連続した後に１、１となる場合に前記被乗数に係る減算を行い、前記配列が下桁側から順に１が連続した後に０、１となる場合に前記被乗数に係る減算を行い、前記配列が下桁側から順に１が連続した後に０、０となる場合に前記被乗数に係る加算を行う演算部と、を備えることを特徴とする。
【００２９】
また、請求項９の発明は、請求項７または請求項８に記載の演算装置であって、前記演算部が、前記被乗数の符号ビットを負論理で扱い且つその他のビットを正論理で扱うことで前記被乗数に係る加算を行い、前記被乗数の符号ビットを正論理で扱い且つその他のビットを負論理で扱うことで前記被乗数に係る減算を行うことで得られた演算結果に、所定の定数を加算することによって、前記被乗数と前記乗数との乗算結果を導出し、前記所定の定数が、前記演算部によって前記加算および減算を行う際に負論理で扱われたビットに係る項に１を代入し、正論理で扱われたビットに係る項に０を代入した上で加算した結果の２の補数であることを特徴とする。
【００３０】
また、請求項１０の発明は、請求項２、請求項３、請求項７、及び請求項８の何れかに記載の演算装置であって、前記演算部が、前記被乗数の符号拡張を行いつつ前記被乗数に係る加算および減算を行うことで、前記被乗数と前記乗数との乗算結果を導出することを特徴とする。
【００３１】
また、請求項１１の発明は、請求項１から請求項１０の何れかに記載の演算装置を用いて入力画像信号を表示用画像信号に変換する変換部と、前記表示用画像信号に基づいて画像を表示する表示部と、を備えることを特徴とする。
【００３２】
また、請求項１２の発明は、請求項１１に記載の画像表示装置であって、前記変換部が、明度および色差を示す信号を所定のルールに従って各色の階調を示す信号に変換するための乗算に、請求項１から請求項１０の何れかに記載の演算装置を用いることを特徴とする。
【００３３】
また、請求項１３の発明は、演算装置において二進法でそれぞれ表現される被乗数と乗数との乗算を行う演算方法であって、(ａ)加算基調の演算を行うステップと、(ｂ)前記(ａ)ステップの後に、前記乗数におけるビットの数値の下桁側からの配列に応じて、前記加算基調の演算および減算基調の演算のうちの少なくとも一方の演算を順次に行うステップと、を備え、前記加算基調の演算が、前記配列が下桁側から順に１、０となる場合に前

記被乗数に係る加算を行い、前記配列が下桁側から順に１、１となる場合に前記減算基調の演算に移行しつつ前記被乗数に係る減算を行う演算であり、前記減算基調の演算が、前記配列が下桁側から順に０、１となる場合に前記被乗数に係る減算を行い、前記配列が下桁側から順に０、０となる場合に前記加算基調の演算に移行しつつ前記被乗数に係る加算を行う演算であることを特徴とする。

【００３４】

また、請求項１４の発明は、演算装置において二進法でそれぞれ表現される被乗数と乗数との乗算を行う演算方法であって、(ａ)前記乗数におけるビットの数値の配列が下桁側から順に０が連続した後に１、０となる場合に前記被乗数に係る加算を行うステップと、(ｂ)前記配列が下桁側から順に０が連続した後に１、１となる場合に前記被乗数に係る減算を行うステップと、(ｃ)前記配列が下桁側から順に１が連続した後に０、１となる場合に前記被乗数に係る減算を行うステップと、(ｄ)前記配列が下桁側から順に１が連続した後に０、０となる場合に前記被乗数に係る加算を行うステップと、を備えることを特徴とする。

【発明の効果】

【００３５】

請求項１から請求項１４の何れに記載の発明によっても、乗算における加算および減算の最大回数が低減されるため、演算装置の小型化および演算速度の向上を図ることができる。

【００３６】

請求項４、請求項５、および請求項９の何れに記載の発明によっても、被乗数の符号拡張を行うことなく、乗算を行うことができるため、演算部の構成の簡略化を図ることができる。

【００３７】

請求項１０に記載の発明によれば、演算の簡略化を図ることができる。

【００３８】

請求項１１および請求項１２の何れに記載の発明によっても、入力画像信号から表示用信号に変換するための構成の簡略化と演算速度の高速化とが容易に図られるため、画像表示装置の小型化および製造コストの低減を図ることができる。

【発明を実施するための最良の形態】

【００３９】

以下、本発明の実施形態を図面に基づいて説明する。

【００４０】

＜演算装置の構成＞

図１は、本発明の実施形態に係る演算装置１の機能的な構成を示す機能ブロック図である。演算装置１は、二進法でそれぞれ表現される変数（具体的には、被乗数と乗数）の乗算を行うものであり、シフトレジスタ２、符号拡張・シフトレジスタ３、制御回路部４、および演算回路部５を備える。

【００４１】

シフトレジスタ２は、本発明の被乗数記憶部に相当するものであり、二進法で表現された被乗数のデータの入力に応じて、該被乗数のデータ、すなわちビット列を記憶する。このシフトレジスタ２は、制御回路部４からのシフト制御信号に応答して、記憶している被乗数のデータの桁をシフトする。

【００４２】

符号拡張・シフトレジスタ３は、本発明の乗数記憶部に相当するものであり、二進法で表現された乗数のデータの入力に応じて、該乗数を符号拡張した乗数（以下「符号拡張乗数」とも称する）を生成し、該符号拡張乗数を記憶する。なお、ここで言う「符号拡張」とは、符号付きのデータをビット長の大きいデータに変換する際に、値を変えないようにビットを補ってデータを拡張することである。また、この符号拡張・シフトレジスタ３は、制御回路部４からのシフト制御信号に応答して、記憶している被乗数のデータの桁をシ

フトする。

【００４３】

なお、当該実施形態においては、乗数を符号拡張した「符号拡張乗数」を用いているが、符合拡張を行っていない乗数を用い、当該乗数を乗数記憶部に記憶させる構造に応用することもできる。

【００４４】

制御回路部４は、例えば、論理回路などによって構成され、符号拡張乗数のビットのパターンの認識に応じて、シフトレジスタ２および符号拡張・シフトレジスタ３へのシフト制御信号の出力を制御するとともに、演算回路部５における演算を制御する。具体的には、この制御回路部４は、符号拡張・シフトレジスタ３に記憶される符号拡張乗数におけるビットのパターンを認識し、その認識結果に応じて、被乗数に係る加算を指示する信号（以下「加算信号」とも称する）または被乗数に係る減算を指示する信号（以下「減算信号」とも称する）を演算回路部５に対して出力する。より詳細には、制御回路部４は、符号拡張乗数におけるビットの数値の下桁側からの配列を認識し、その配列の認識結果に応じた演算を順次に行うことを決定して、その決定に応じた加算信号および減算信号のうちの少なくとも一方の信号を順次に出力する。なお、ここで決定される演算には、加算基調の演算と減算基調の演算とがあり、まず、加算基調の演算が行われた後に、加算基調の演算および減算基調の演算のうちの少なくとも一方の演算が順次に行われることが決定される。「加算基調の演算」および「減算基調の演算」の演算内容については後述する。

【００４５】

演算回路部５は、例えば、論理回路などによって構成され、制御回路部４による決定結果に応じて、加算基調の演算および減算基調の演算を行う。具体的には、制御回路部４から演算回路部５に入力される加算信号に応答して、シフトレジスタ２から被乗数のデータを読み出して該被乗数のデータに係る加算を行い、制御回路部４から演算回路部５に入力される減算信号に応答して、シフトレジスタ２から被乗数のデータを読み出して該被乗数のデータに係る減算を行う。そして、演算回路部５において、被乗数の桁を順次に上位側にシフトさせつつ、順次に被乗数の加減算を行うことにより、被乗数と乗数との乗算の演算結果を示すデータ（以下「演算結果データ」とも称する）が導出され、該演算結果データが演算回路部５から出力される。

【００４６】

なお、当該実施形態においては、被乗数をシフトレジスタ２で左にシフトすることによって倍数を作り、演算回路部５で加減算する構造となっているが、当該構造に限定されない。例えば、被乗数のデータを記憶するレジスタを、被乗数のビット列をシフトしないレジスタとして、演算回路部に設けられたシフトレジスタにおいて加減算の結果のビット列を右にシフトさせることで、被乗数のデータを相対的に左にシフトしても良い。この変形例に係る演算装置１ａの機能的な構成を示す機能ブロック図を図２に例示する。図２では、図１で示した演算装置１と同様な構成については同じ符号が付されている。演算装置１ａでは、演算装置１と比較して、シフトレジスタ２がビット列のシフトを行わないレジスタ２ａに変更され、制御回路部４がシフト制御信号の出力先が演算回路部５ａとされた制御回路部４ａに変更され、演算回路部５がシフトレジスタを含む演算回路部５ａに変更されている。そして、演算回路部５ａが、制御回路部４ａからのシフト制御信号に応じて加減算の結果を示すビット列をシフトしつつ、レジスタ２ａから被乗数のデータを読み込んで、加算信号および減算信号に応じて被乗数の加減算を行う。当該構成においては、演算に必要なビットの記憶場所が固定されるので、回路サイズの縮小および動作速度の向上を図ることができる。

【００４７】

＜演算装置における演算処理フロー＞

図３から図５は、演算装置１における演算処理フローを示すフローチャートである。本演算処理フローは、演算装置１への被乗数および乗数のデータの入力に応答して開始される。

【００４８】
ステップＳ１では、符号拡張・シフトレジスタ３により、乗数について１ビット分の符号拡張が行われ、乗数のデータが１ビット増加される。このとき、符号拡張された乗数（符号拡張乗数）のデータが符号拡張・シフトレジスタ３に記憶される。
【００４９】
ステップＳ２では、制御回路部４により、符号拡張乗数の数値の認識対象となるビットの桁を特定するためのカウント値ｎが１に設定される。
【００５０】
ステップＳ３では、制御回路部４により、符号拡張乗数の全ビットに係る数値が認識されたか否か判定される。ここでは、全ビットに係る数値が認識されていなければ、ステップＳ４に進み、全ビットに係る数値が認識されていれば、図５のステップＳ３１に進む。
【００５１】
ステップＳ４では、制御回路部４により、符号拡張乗数の最も下位の桁（最下桁）からｎ番目およびｎ＋１番目のビットの数値が認識される。
【００５２】
ステップＳ５では、制御回路部４により、符号拡張乗数の最下桁からｎ番目のビットの数値が１であるか否か判定される。ここでは、符号拡張乗数の最下桁からｎ番目のビットの数値が１であれば、ステップＳ６に進み、符号拡張乗数の最下桁からｎ番目のビットの数値が１でなければ、ステップＳ７に進む。
【００５３】
ステップＳ６では、制御回路部４により、符号拡張乗数の最下桁からｎ＋１番目のビットの数値が１であるか否か判定される。ここでは、ｎ＋１番目のビットの数値が１であれば、ステップＳ８に進み、ｎ＋１番目のビットの数値が１でなければ、ステップＳ１０に進む。
【００５４】
ステップＳ７では、制御回路部４により、カウント値ｎが１増加されるとともに、制御回路部４からシフト制御信号が出力されて、シフトレジスタ２に記憶される被乗数のデータが１ビット分左にシフトされる。つまり、被乗数のデータが１桁上位側にシフトされる。また、このとき、符号拡張・シフトレジスタ３に記憶される符号拡張乗数のデータが１ビット分右にシフトされる。つまり、符号拡張乗数のデータが１桁下位側にシフトされる。そして、ステップＳ７の処理が終了すると、ステップＳ３に進む。
【００５５】
ステップＳ８では、制御回路部４から減算信号が演算回路部５に出力され、演算回路部５により、シフトレジスタ２に記憶される被乗数のデータに係る減算が行われる。具体的には、被乗数の符号ビットの数値は正論理、その他のビットの数値は負論理で加算される。つまり、被乗数の符号ビットの数値はそのまま加算され、被乗数のその他のビットの数値は反転された上で加算される。
【００５６】
ステップＳ９では、制御回路部４により、カウント値ｎが２つ増加されるとともに、制御回路部４からシフト制御信号が出力されて、シフトレジスタ２に記憶される被乗数のデータが２ビット分左にシフトされる。つまり、被乗数のデータが２桁上位側にシフトされる。また、このとき、符号拡張・シフトレジスタ３に記憶される符号拡張乗数のデータが２ビット分右にシフトされる。つまり、符号拡張乗数のデータが２桁下位側にシフトされる。そして、ステップＳ９の処理が終了すると、図４のステップＳ２１に進む。
【００５７】
ステップＳ１０では、制御回路部４から加算信号が演算回路部５に出力され、演算回路部５により、シフトレジスタ２に記憶される被乗数のデータに係る加算が行われる。具体的には、被乗数の符号ビットの数値は負論理、その他のビットの数値は正論理で加算される。つまり、被乗数の符号ビットの数値は反転されて加算され、被乗数のその他のビットの数値はそのまま加算される。

【００５８】
ステップＳ１１では、制御回路部４により、カウント値ｎが２つ増加されるとともに、制御回路部４からシフト制御信号が出力されて、シフトレジスタ２に記憶される被乗数のデータが２ビット分左にシフトされる。つまり、被乗数のデータが２桁上位側にシフトされる。また、このとき、符号拡張・シフトレジスタ３に記憶される符号拡張乗数のデータが２ビット分右にシフトされる。つまり、符号拡張乗数のデータが２桁下位側にシフトされる。そして、ステップＳ１１の処理が終了すると、ステップＳ３に進む。
【００５９】
図４のステップＳ２１では、図３のステップＳ３と同様に、制御回路部４により、符号拡張乗数の全ビットに係る数値が認識されたか否か判定される。ここでは、全ビットに係る数値が認識されていなければ、ステップＳ２２に進み、全ビットに係る数値が認識されていれば、図５のステップＳ３１に進む。
【００６０】
ステップＳ２２では、制御回路部４により、符号拡張乗数の最下桁からｎ番目およびｎ＋１番目のビットの数値が認識される。
【００６１】
ステップＳ２３では、制御回路部４により、符号拡張乗数の最下桁からｎ番目のビットの数値が０であるか否か判定される。ここでは、符号拡張乗数の最下桁からｎ番目のビットの数値が０であれば、ステップＳ２４に進み、符号拡張乗数の最下桁からｎ番目のビットの数値が０でなければ、ステップＳ２５に進む。
【００６２】
ステップＳ２４では、制御回路部４により、符号拡張乗数の最下桁からｎ＋１番目のビットの数値が０であるか否か判定される。ここでは、符号拡張乗数の最下桁からｎ＋１番目のビットの数値が０であれば、ステップＳ２６に進み、符号拡張乗数の最下桁からｎ＋１番目のビットの数値が０でなければ、ステップＳ２８に進む。
【００６３】
ステップＳ２５では、図３のステップＳ７と同様に、制御回路部４により、カウント値ｎが１増加されるとともに、制御回路部４からシフト制御信号が出力されて、シフトレジスタ２に記憶される被乗数のデータが１ビット分左にシフトされる。つまり、被乗数のデータが１桁上位側にシフトされる。また、このとき、符号拡張・シフトレジスタ３に記憶される符号拡張乗数のデータが１ビット分右にシフトされる。つまり、符号拡張乗数のデータが１桁下位側にシフトされる。そして、ステップＳ２５の処理が終了すると、ステップＳ２１に進む。
【００６４】
ステップＳ２６では、図３のステップＳ１０と同様に、制御回路部４から加算信号が演算回路部５に出力され、演算回路部５により、シフトレジスタ２に記憶される被乗数のデータに係る加算が行われる。具体的には、被乗数の符号ビットの数値は負論理、その他のビットの数値は正論理で加算される。つまり、被乗数の符号ビットの数値は反転されて加算され、被乗数のその他のビットの数値はそのまま加算される。
【００６５】
ステップＳ２７では、図３のステップＳ１１と同様に、制御回路部４により、カウント値ｎが２つ増加されるとともに、制御回路部４からシフト制御信号が出力されて、シフトレジスタ２に記憶される被乗数のデータが２ビット分左にシフトされる。つまり、被乗数のデータが２桁上位側にシフトされる。また、このとき、符号拡張・シフトレジスタ３に記憶される符号拡張乗数のデータが２ビット分右にシフトされる。つまり、符号拡張乗数のデータが２桁下位側にシフトされる。そして、ステップＳ２７の処理が終了すると、図３のステップＳ３に進む。
【００６６】
ステップＳ２８では、図３のステップＳ８と同様に、制御回路部４から減算信号が演算回路部５に出力され、演算回路部５により、シフトレジスタ２に記憶される被乗数のデー

タに係る減算が行われる。具体的には、被乗数の符号ビットの数値は正論理、その他のビットの数値は負論理で加算される。つまり、被乗数の符号ビットの数値はそのまま加算され、被乗数のその他のビットの数値は反転された上で加算される。

【００６７】

ステップＳ２９では、図３のステップＳ９と同様に、制御回路部４により、カウント値ｎが２つ増加されるとともに、制御回路部４からシフト制御信号が出力されて、シフトレジスタ２に記憶される被乗数のデータが２ビット分左にシフトされる。つまり、被乗数のデータが２桁上位側にシフトされる。また、このとき、符号拡張・シフトレジスタ３に記憶される符号拡張乗数のデータが２ビット分右にシフトされる。つまり、符号拡張乗数のデータが２桁下位側にシフトされる。そして、ステップＳ２９の処理が終了すると、ステップＳ２１に進む。

【００６８】

図５のステップＳ３１では、制御回路部４および演算回路部５により、図３のステップＳ８，Ｓ１０、および図４のステップＳ２６，Ｓ２８で加減算された際に負論理で扱われたビットの項に１が代入され、正論理で扱われたビットの項に０が代入された上で加算されることで、オフセット値が算出される。なお、被乗数に０を代入して、ステップＳ１～Ｓ２９の処理を行うことでオフセット値を算出しても良い。このオフセット値は、被乗数が０になった場合には、被乗数と乗数との乗算結果が０となるべきところ、演算の誤差によって生じる０からの値のズレ（ズレ値）を示す。

【００６９】

ステップＳ３２では、制御回路部４および演算回路部５により、ステップＳ３１で算出されたオフセット値の２の補数が所定の定数として算出される。

【００７０】

ステップＳ３３では、図３および図４のステップＳ１～Ｓ２９における順次の加減算によって得られた値に、ステップＳ３２で算出された所定の定数が加算されることで、被乗数と乗数との乗算結果が求められて、本演算処理フローが終了される。ステップＳ３３では、被乗数に０を代入して被乗数と乗数との乗算結果が０からずれる値（ズレ値）を相殺することを目的として、所定の定数が加算される。

【００７１】

図３から図５で示す演算処理フローでは、主に、加算基調の演算、減算基調の演算、および所定の定数の加算が行われる。図３のステップＳ３～Ｓ１１の演算処理が加算基調の演算に相当し、図４のステップＳ２１～Ｓ２９で示される演算処理が減算基調の演算に相当し、図５のステップＳ３１～Ｓ３３の演算処理が所定の定数の加算に相当する。そして、図３で示すように、演算装置１における被乗数と乗数との乗算では、まず、加算基調の演算から開始される。

【００７２】

具体的には、加算基調の演算は、符号拡張乗数におけるビットの数値の下桁側からの配列において「０」が連続して出現する間に「１」が単独で出現すると、被乗数の加算が行われる加算を基調とする演算である。より詳細には、加算基調の演算では、符号拡張乗数におけるビットの数値の下桁側からの配列において、「０」が連続している場合には被乗数の加算も減算も行われず、下桁側から順に見て、ビットの数値に「１」が出現し且つ下桁側からのビットの数値の配列が「１、０」である場合には被乗数の加算が行われ、ビットの数値に「１」が出現し且つ下桁側からのビットの数値の配列が「１、１」である場合には減算基調の演算に移行されつつ被乗数の減算が行われる。つまり、「１」が連続するパターンが出現するまで、「１」が単独して出現すれば、被乗数の加算が行われ、「１」が連続するパターンが出現すれば、減算基調の演算への移行が行われる。

【００７３】

また、減算基調の演算は、符号拡張乗数におけるビットの数値の下桁側からの配列において、「１」が連続して出現する間に「０」が単独で出現すると、被乗数の減算が行われる減算を基調とする演算である。より詳細には、減算基調の演算では、符号拡張乗数にお

けるビットの数値の下桁側からの配列において、「１」が連続している場合には被乗数の加算も減算も行われず、下桁側から順に見て、ビットの数値に「０」が出現し且つ下桁側からのビットの数値の配列が「０、１」である場合には被乗数の減算が行われ、ビットの数値に「０」が出現し且つ下桁側からのビットの数値の配列が「０、０」である場合には加算基調の演算に移行されつつ被乗数の加算が行われる。つまり、「０」が連続するパターンが出現するまで、「０」が単独して出現すれば、被乗数の減算が行われ、「０」が連続するパターンが出現すれば、加算基調の演算への移行が行われる。

【００７４】

換言すれば、制御回路部４は、符号拡張乗数におけるビットの数値の下桁側からの配列を認識して、該配列が下桁側から順に０が連続した後に１、０となる場合に被乗数に係る加算を行い、該配列が下桁側から順に０が連続した後に１、１となる場合に被乗数に係る減算を行うことを決定する。また、制御回路部４は、該配列が下桁側から順に１が連続した後に０、１となる場合に被乗数に係る減算を行い、該配列が下桁側から順に１が連続した後に０、０となる場合に被乗数に係る加算を行うことを決定する。

【００７５】

＜演算装置における演算例＞

図６は、演算装置１における演算例を示す図である。図６では、十進法で表現された「５×(－６)」の乗算を行う例が示されている。図６(ａ)で示すように、被乗数に相当する十進法の「５」は、二進法の４ビット整数表記では「０１０１」であり、乗数に相当する十進法の「－６」は、二進法の４ビット整数表記では「１０１０」である。そして、図６(ａ)で示すように、図３および図４のステップＳ１～Ｓ２９における加減算によって、二進法の７ビット整数表記の「０１０１０１０」が得られる。また、図６(ｂ)で示すように、図５のステップＳ３１における演算により、二進法の７ビット整数表記のオフセット値「１００１０００」が算出される。そして、図６(ｃ)で示すように、図３および図４のステップＳ１～Ｓ２９における加減算によって得られた二進法の７ビット整数表記の「０１０１０１０」に、オフセット値の２の補数「１０１１１０００」が加算されることで、被乗数と乗数との乗算結果「１１１０００１０」が得られる。この二進法の８ビット整数表記の「１１１０００１０」は、十進法では「－３０（＝－2^7＋2^6＋2^5＋2^1）」であり、十進法で表現された「５×(－６)」の乗算が正しく行われていることが分かる。

【００７６】

＜加減算の決定方法のバリエーション＞

上記実施形態に係る演算装置１では、乗数をそのまま符号拡張して被乗数に係る加減算の決定に利用したが、これに限られない。例えば、乗数の２の補数を算出し、この値を仮の乗数として被乗数に係る加減算の決定に利用するようにしても良い。以下では、上記実施形態のようにそのまま符号拡張した乗数を用いて被乗数に係る加減算を決定する乗算アルゴリズムを「ＷＳ法の乗算アルゴリズム」と称し、乗数の２の補数を符号拡張した値を用いて被乗数に係る加減算を決定する乗算アルゴリズムを「２の補数のＷＳ法の乗算アルゴリズム」と称する。なお、ＷＳ法の乗算アルゴリズムと、２の補数のＷＳ法の乗算アルゴリズムとは、加減算の決定方法が異なるが、結果的に被乗数に係る加減算の内容は同一となる。ここで、２の補数のＷＳ法の乗算アルゴリズムについて説明する。

【００７７】

図７から図９は、演算装置１において、２の補数のＷＳ法の乗算アルゴリズムを採用した場合の演算処理フローを示すフローチャートである。本演算処理フローは、演算装置１への被乗数および乗数のデータの入力に応答して開始される。

【００７８】

ステップＳ１０１では、図示を省略する演算回路により、乗数の２の補数が算出され、この算出された値を仮の乗数（仮乗数）として、該仮乗数が符号拡張・シフトレジスタ３に入力される。

【００７９】

ステップＳ１０２では、符号拡張・シフトレジスタ３により、仮乗数について１ビット

分の符号拡張が行われ、仮乗数のデータが１ビット増加される。このとき、符号拡張された仮乗数（符号拡張仮乗数）のデータが符号拡張・シフトレジスタ３に記憶される。

【００８０】

ステップＳ１０３では、制御回路部４により、符号拡張乗数の数値の認識対象となるビットの桁を特定するためのカウント値ｎが１に設定される。

【００８１】

ステップＳ１０４では、制御回路部４により、符号拡張仮乗数の全ビットに係る数値が認識されたか否か判定される。ここでは、全ビットに係る数値が認識されていなければ、ステップＳ１０５に進み、全ビットに係る数値が認識されていれば、図９のステップＳ１３１に進む。

【００８２】

ステップＳ１０５では、制御回路部４により、符号拡張仮乗数の最下桁からｎ番目およびｎ＋１番目のビットの数値が認識される。

【００８３】

ステップＳ１０６では、制御回路部４により、符号拡張仮乗数の最下桁からｎ番目のビットの数値が１であるか否か判定される。ここでは、符号拡張仮乗数の最下桁からｎ番目のビットの数値が１であれば、ステップＳ１０７に進み、符号拡張仮乗数の最下桁からｎ番目のビットの数値が１でなければ、ステップＳ１０８に進む。

【００８４】

ステップＳ１０７では、制御回路部４により、符号拡張仮乗数の最下桁からｎ＋１番目のビットの数値が１であるか否か判定される。ここでは、符号拡張仮乗数の最下桁からｎ＋１番目のビットの数値が１であれば、ステップＳ１０９に進み、符号拡張仮乗数の最下桁からｎ＋１番目のビットの数値が１でなければ、ステップＳ１１１に進む。

【００８５】

ステップＳ１０８では、制御回路部４により、カウント値ｎが１つ増加されるとともに、制御回路部４からシフト制御信号が出力されて、シフトレジスタ２に記憶される被乗数のデータが１ビット分左にシフトされる。つまり、被乗数のデータが１桁上位側にシフトされる。また、このとき、符号拡張・シフトレジスタ３に記憶される符号拡張仮乗数のデータが１ビット分右にシフトされる。つまり、符号拡張仮乗数のデータが１桁下位側にシフトされる。そして、ステップＳ１０８の処理が終了されると、ステップＳ１０４に進む。

【００８６】

ステップＳ１０９では、制御回路部４から加算信号が演算回路部５に出力され、演算回路部５により、シフトレジスタ２に記憶される被乗数のデータに係る加算が行われる。具体的には、被乗数の符号ビットの数値は負論理、その他のビットの数値は正論理で加算される。つまり、被乗数の符号ビットの数値は反転されて加算され、被乗数のその他のビットの数値はそのまま加算される。

【００８７】

ステップＳ１１０では、制御回路部４により、カウント値ｎが２つ増加されるとともに、制御回路部４からシフト制御信号が出力されて、シフトレジスタ２に記憶される被乗数のデータが２ビット分左にシフトされる。つまり、被乗数のデータが２桁上位側にシフトされる。また、このとき、符号拡張・シフトレジスタ３に記憶される符号拡張仮乗数のデータが２ビット分右にシフトされる。つまり、符号拡張仮乗数のデータが２桁下位側にシフトされる。そして、ステップＳ１１０の処理が終了すると、図８のステップＳ１２１に進む。

【００８８】

ステップＳ１１１では、制御回路部４から減算信号が演算回路部５に出力され、演算回路部５により、シフトレジスタ２に記憶される被乗数のデータに係る減算が行われる。具体的には、被乗数の符号ビットの数値は正論理、その他のビットの数値は負論理で加算される。つまり、被乗数の符号ビットの数値はそのまま加算され、被乗数のその他のビット

の数値は反転された上で加算される。

【００８９】

ステップＳ１１２では、制御回路部４により、カウント値ｎが２つ増加されるとともに、制御回路部４からシフト制御信号が出力されて、シフトレジスタ２に記憶される被乗数のデータが２ビット分左にシフトされる。つまり、被乗数のデータが２桁上位側にシフトされる。また、このとき、符号拡張・シフトレジスタ３に記憶される符号拡張仮乗数のデータが２ビット分右にシフトされる。つまり、符号拡張仮乗数のデータが２桁下位側にシフトされる。そして、ステップＳ１１２の処理が終了すると、ステップＳ１０４に進む。

【００９０】

図８のステップＳ１２１では、図７のステップＳ１０４と同様に、制御回路部４により、符号拡張仮乗数の全ビットに係る数値が認識されたか否か判定される。ここでは、全ビットに係る数値が認識されていなければ、ステップＳ１２２に進み、全ビットに係る数値が認識されていれば、図９のステップＳ１３１に進む。

【００９１】

ステップＳ１２２では、制御回路部４により、符号拡張仮乗数の最下桁からｎ番目およびｎ＋１番目のビットの数値が認識される。

【００９２】

ステップＳ１２３では、制御回路部４により、符号拡張仮乗数の最下桁からｎ番目のビットの数値が０であるか否か判定される。ここでは、符号拡張仮乗数の最下桁からｎ番目のビットの数値が０であれば、ステップＳ１２４に進み、符号拡張仮乗数の最下桁からｎ番目のビットの数値が０でなければ、ステップＳ１２５に進む。

【００９３】

ステップＳ１２４では、制御回路部４により、符号拡張仮乗数の最下桁からｎ＋１番目のビットの数値が０であるか否か判定される。ここでは、符号拡張仮乗数の最下桁からｎ＋１番目のビットの数値が０であれば、ステップＳ１２６に進み、符号拡張仮乗数の最下桁からｎ＋１番目のビットの数値が０でなければ、ステップＳ１２８に進む。

【００９４】

ステップＳ１２５では、図７のステップＳ１０８と同様に、制御回路部４により、カウント値ｎが１つ増加されるとともに、制御回路部４からシフト制御信号が出力されて、シフトレジスタ２に記憶される被乗数のデータが１ビット分左にシフトされる。つまり、被乗数のデータが１桁上位側にシフトされる。また、このとき、符号拡張・シフトレジスタ３に記憶される符号拡張仮乗数のデータが１ビット分右にシフトされる。つまり、符号拡張仮乗数のデータが１桁下位側にシフトされる。

【００９５】

ステップＳ１２６では、図７のステップＳ１１１と同様に、制御回路部４から減算信号が演算回路部５に出力され、演算回路部５により、シフトレジスタ２に記憶される被乗数のデータに係る減算が行われる。具体的には、被乗数の符号ビットの数値は正論理、その他のビットの数値は負論理で加算される。つまり、被乗数の符号ビットの数値はそのまま加算され、被乗数のその他のビットの数値は反転された上で加算される。

【００９６】

ステップＳ１２７では、図７のステップＳ１１２と同様に、制御回路部４により、カウント値ｎが２つ増加されるとともに、制御回路部４からシフト制御信号が出力されて、シフトレジスタ２に記憶される被乗数のデータが２ビット分左にシフトされる。つまり、被乗数のデータが２桁上位側にシフトされる。また、このとき、符号拡張・シフトレジスタ３に記憶される符号拡張仮乗数のデータが２ビット分右にシフトされる。つまり、符号拡張仮乗数のデータが２桁下位側にシフトされる。そして、ステップＳ１２７の処理が終了すると、図７のステップＳ１０４に進む。

【００９７】

ステップＳ１２８では、図７のステップＳ１０９と同様に、制御回路部４から加算信号が演算回路部５に出力され、演算回路部５により、シフトレジスタ２に記憶される被乗数

のデータに係る加算が行われる。具体的には、被乗数の符号ビットの数値は負論理、その他のビットの数値は正論理で加算される。つまり、被乗数の符号ビットの数値は反転されて加算され、被乗数のその他のビットの数値はそのまま加算される。

【００９８】

ステップＳ１２９では、図７のステップＳ１１０と同様に、制御回路部４により、カウント値ｎが２つ増加されるとともに、制御回路部４からシフト制御信号が出力されて、シフトレジスタ２に記憶される被乗数のデータが２ビット分左にシフトされる。つまり、被乗数のデータが２桁上位側にシフトされる。また、このとき、符号拡張・シフトレジスタ３に記憶される符号拡張仮乗数のデータが２ビット分右にシフトされる。つまり、符号拡張仮乗数のデータが２桁下位側にシフトされる。そして、ステップＳ１２９の処理が終了すると、ステップＳ１２１に進む。

【００９９】

図９のステップＳ１３１では、図５のステップＳ３１と同様に、制御回路部４および演算回路部５により、図７のステップＳ１０９，Ｓ１１１、および図８のステップＳ１２６，Ｓ１２８で加減算された際に負論理で扱われたビットの項に１が代入され、正論理で扱われたビットの項に０が代入された上で加算されることで、オフセット値が算出される。

【０１００】

ステップＳ１３２では、制御回路部４および演算回路部５により、ステップＳ１３１で算出されたオフセット値の２の補数が所定の定数として算出される。

【０１０１】

ステップＳ１３３では、図７および図８のステップＳ１０１～Ｓ１２９における順次の加減算によって得られた値に、ステップＳ１３２で算出された所定の定数が加算されることで、被乗数と乗数との乗算結果が求められて、本演算処理フローが終了される。

【０１０２】

＜従来技術との比較＞

図１０は、演算装置１における被乗数に係る加減算の回数と、従来技術における被乗数に係る加減算の回数とを比較するための図である。図１０では、二進法で表現された乗数が２６ビットの「０１０１１０１０１０００００１００１１１１００１１０」である場合に、被乗数と乗数との乗算を行う際に、被乗数の加減算が行われる桁、つまり被乗数の加減算の回数が示されている。具体的には、図１０(ａ)がブースの乗算アルゴリズムを用いた場合に被乗数の加減算が行われる桁を示し、図１０(ｂ)が定数加算法の乗算アルゴリズムを用いた場合に被乗数の加減算が行われる桁を示し、図１０(ｃ)が２の補数の定数加算法の乗算アルゴリズムを用いた場合に被乗数の加減算が行われる桁を示し、図１０(ｄ)がＷＳ法の乗算アルゴリズムを用いた場合に被乗数の加減算が行われる桁を示し、図１０(ｅ)が２の補数のＷＳ法の乗算アルゴリズムを用いた場合に被乗数の加減算が行われる桁を示している。なお、図１０では、被乗数に係る加算が行われる桁の数値が太線Ａｄｄで囲まれ、被乗数に係る減算が行われる桁の数値が細線Ｓｕｂで囲まれている。

【０１０３】

図１０(ａ)で示すように、ブースの乗算アルゴリズムを用いた場合には、合計１４回の被乗数に係る加減算（７回の被乗数に係る加算、および７回の被乗数に係る減算）が行われる。また、図１０(ｂ)で示すように、定数加算法の乗算アルゴリズムを用いた場合には、１２回の被乗数に係る加算が行われる。また、図１０(ｃ)で示すように、２の補数の定数加算法の乗算アルゴリズムを用いた場合には、合計１４回の被乗数に係る加減算（１回の被乗数に係る加算、および１３回の被乗数に係る減算）が行われる。これに対して、図１０(ｄ)，(ｅ)で示すように、ＷＳ法の乗算アルゴリズムおよび２の補数のＷＳ法の乗算アルゴリズムを用いた場合には、合計１０回の被乗数に係る加減算（６回の被乗数に係る加算、および４回の被乗数に係る減算）が行われる。したがって、従来技術と比較して、ＷＳ法の乗算アルゴリズムおよび２の補数のＷＳ法の乗算アルゴリズムを用いた場合には、被乗数に係る加減算の回数が低減される。

【０１０４】

ここで、二進法で表現されるＮビット（Ｎは２以上の自然数）の乗数と被乗数とを乗算する場合について、被乗数に係る加減算の最大回数を考える。ブースの乗算アルゴリズムを用いる場合、仮に乗数を構成するビットの数値の配列が１と０とが全長に渡って交互に並ぶものとなると、被乗数に係る加減算の回数が最大回数である（Ｎ－１）回となる。また、定数加算法の乗算アルゴリズムを用いる場合、仮に乗数を構成するビットの数値の配列が１が全長に渡って並ぶものとなると、被乗数に係る加減算の回数が最大回数であるＮ回となる。更に、２の補数の定数加算法の乗算アルゴリズムを用いる場合、仮に乗数の２の補数を構成するビットの数値の配列が１が全長に渡って並ぶものとなると、被乗数に係る加減算の回数が最大回数であるＮ回となる。

【０１０５】

これに対して、ＷＳ法の乗算アルゴリズムを用いる場合には、仮に乗数を構成するビットの数値の配列が１と０とが全長に渡って交互に並ぶものとなると、Ｎが偶数の場合には、被乗数に係る加減算の回数が最大回数であるＮ／２回となり、Ｎが奇数の場合には、被乗数に係る加減算の回数が最大回数である（Ｎ＋１）／２回となる。また、２の補数のＷＳ法の乗算アルゴリズムを用いる場合にも、仮に乗数の２の補数を構成するビットの数値の配列が１と０とが全長に渡って交互に並ぶものとなると、Ｎが偶数の場合には、被乗数に係る加減算の回数が最大回数であるＮ／２回となり、Ｎが奇数の場合には、被乗数に係る加減算の回数が最大回数である（Ｎ＋１）／２回となる。

【０１０６】

したがって、乗数が４ビット以上のものであれば、ブースの乗算アルゴリズム、定数加算法の乗算アルゴリズム、および２の補数の定数加算法の乗算アルゴリズムを用いる従来技術と比べて、ＷＳ法の乗算アルゴリズムおよび２の補数のＷＳ法の乗算アルゴリズムを用いる場合には、乗数と被乗数とを乗算する際における被乗数に係る加減算の最大回数が相対的に少なくなる。そして、演算回路部５の構成や性能が、被乗数に係る加減算の最大回数に合わせたものとなることを考慮すると、ＷＳ法の乗算アルゴリズムおよび２の補数のＷＳ法の乗算アルゴリズムを採用することで、従来技術と比較して、乗算における加算および減算の最大回数が低減されるため、演算装置１の小型化および演算速度の向上を図ることができる。このような効果は、乗数のビット数が増える程、顕著となる。

【０１０７】

図１１は、二進法で表現された乗数が図１０で示すような２６ビットの「０１０１１０１０１００００００１００１１１１００１１０」である場合に、被乗数に係る加減算および所定の定数の加算を行う演算態様を示すイメージ図である。図１１では、被乗数の加算に係るビット列Ａｄｄと、被乗数の減算に係るビット列Ｓｕｂと、所定の定数の加算に係るビット列Ｃｏｎｓｔとが模式的に示されており、ビット列Ａｄｄ，Ｓｕｂ，Ｃｏｎｓｔが相互に加算されることで、被乗数と乗数との乗算結果が得られる。

【０１０８】

なお、以上では、被乗数および乗数の双方が変数である場合にも被乗数と乗数との乗算が可能な構成を示して説明したが、これに限られない。例えば、乗数が既知の定数である場合には、予め乗数のビットの数値の配列を解析して、被乗数の加減算を行う桁、ならびに加算するための所定の定数を求めておくことができる。そして、乗算を実際に行う際には、乗数のビットを解析することなく、シフトレジスタにおいて被乗数のデータを決められた桁までシフトしつつ該被乗数に係る加算または減算を順次に行うとともに、所定の定数を加算するようにすれば良い。このような構成を採用することで、例えば、演算装置１から符号拡張・シフトレジスタ３を取り除き、制御回路部４および演算回路部５の構成の簡略化を図った演算回路を実現することができる。

【０１０９】

＜ＷＳ法の乗算アルゴリズムの画像表示装置への応用＞

図１２は、ビデオカメラ１０、テレビ２０、および画像表示装置３０を備えて構成される映像システムを例示する図である。ビデオカメラ１０は、ＮＴＳＣの規格に基づく画像信号（以下「ＮＴＳＣ信号」と称する）を取得して、該ＮＴＳＣ信号をテレビ２０および

画像表示装置３０に提供する。テレビ２０は、アナログテレビとして構成され、ＮＴＳＣ信号をそのまま利用して、該ＮＴＳＣ信号に基づく動画を表示する。一方、画像表示装置３０は、ＮＴＳＣ信号から、赤(Ｒ)色、緑(Ｇ)色、青(Ｂ)色の３原色の画像信号（以下「ＲＧＢ画像信号」と称する）に変換した上で、該ＲＧＢ画像信号に基づく動画を表示する。この画像表示装置３０では、ＮＴＳＣ信号からＲＧＢ画像信号を生成する際に、上述したＷＳ法の乗算アルゴリズムを利用する。以下、画像表示装置３０の構成について説明する。

【０１１０】

＜画像表示装置の構成＞

図１３は、画像表示装置３０の機能構成を示すブロック図である。画像表示装置３０は、信号変換部３１、ＲＧＢ変換部３２、タイミング制御部３３、表示パネルＤＰ、ＸドライバＸｄ、およびＹドライバＹｄを備える。

【０１１１】

信号変換部３１は、ＮＴＳＣ信号を受け付け、アナログ動画信号であるＮＴＳＣ信号を所定の規格（例えば、ＩＴＵ－Ｒ ＢＴ．６０１）で規定されたフォーマットに沿ってデジタル信号に変換し、該デジタル信号の各フレームを構成する各画素の信号（画素信号）について、いわゆるＹＣ分離（輝度信号と色差信号とを高精度に分離する技術）を行う。そして、信号変換部３１は、ＹＣ分離によって得られる輝度信号であるＹのデータと、色差信号であるＣｒ，Ｃｂのデータとを出力する。

【０１１２】

ＲＧＢ変換部３２は、上式(１)～(３)で示される所定の変換式に沿って、各フレームの画素信号について、入力画像信号であるＹ，Ｃｒ，Ｃｂのデータを表示用画像信号であるＲ，Ｇ，Ｂのデータ（ＲＧＢ画像信号）に変換する。図１４は、ＲＧＢ変換部３２の構成を例示する図である。正数を示すＹのデータが、上式(１)～(３)の演算を行う回路に供され、整数を示すＣｒのデータが、上式(１)，(２)の演算を行う回路に供され、整数を示すＣｂのデータが、上式(２)，(３)の演算を行う回路に供される。このＲＧＢ変換部３２では、上式(１)～(３)に沿った演算に上述したＷＳ法の乗算アルゴリズムを利用する。この演算の具体例については後述する。

【０１１３】

タイミング制御部３３は、ＮＴＳＣ信号の水平同期信号Ｈｓｙｎｃおよび垂直同期信号Ｖｓｙｎｃに基づき、ＸドライバＸｄに制御信号（例えば、スタートパルスＳＴＨ）を出力するとともに、ＹドライバＹｄに制御信号（例えば、スタートパルスＳＴＶ）を出力することで、表示パネルＤＰにおける動画像の表示タイミングを制御する。

【０１１４】

表示パネルＤＰは、例えば、有機ＥＬ素子などの発光素子をそれぞれ含む多数の発光部がマトリックス状に配列されたディスプレイを備え、例えば、横３２０×縦２４０のドットからなるＱＶＧＡの解像度の画像が再生可能に構成されている。ここでは、多数の発光部は、それぞれＲ色の光を発する発光素子を有する発光部と、Ｇ色の光を発する発光素子を有する発光部と、Ｂ色の光を発する発光素子を有する発光部とによって構成されている。そして、表示パネルＤＰは、ＲＧＢ画像信号に基づいて画像を表示する表示部として機能する。

【０１１５】

ＸドライバＸｄは、タイミング制御部３３からのスタートパルスＳＴＨ、およびＲＧＢ変換部３２からのＲＧＢ画像信号に基づき、該ＲＧＢ画像信号に含まれる各画素のデータ信号（画素信号）に応じた電位を表示パネルＤＰに配設された画像信号線に付与する。

【０１１６】

ＹドライバＹｄは、タイミング制御部３３からのスタートパルスＳＴＶに応答して、表示パネルＤＰに配設された走査信号線に所定の電位を付与する。

【０１１７】

＜ＷＳ法の乗算アルゴリズムを用いたＲＧＢ変換＞

Ｙ，Ｃｒ，ＣｂのデータをＲ，Ｇ，Ｂのデータに変換する変換式は、具体的には、下式(４)～(６)で示される。

【０１１８】

Ｒ＝１．１６４×(Ｙ－１６)＋１．５９６×(Ｃｒ－１２８)　・・・(４)

Ｇ＝１．１６４×(Ｙ－１６)－０．３９１×(Ｃｂ－１２８)－０．８１３×(Ｃｒ－１２８)　・・・(５)

Ｂ＝１．１６４×(Ｙ－１６)＋２．０１８×(Ｃｂ－１２８)　・・・(６)。

【０１１９】

ここでは、Ｙが０～２５５の範囲で変化する場合が示されている。そして、上式(４)～(６)において、(Ｙ－１６)となるのは、Ｙの０～１５は同期信号として使用され、１６～２５５が実際の輝度信号になるためである。但し、Ｙが０～１５のときには、輝度信号が負の値を示すことになるが、この場合には、ＲＧＢ色の発光が不要となるため、ＲＧＢの信号を０とする。また、Ｃｒ，Ｃｂについては、正負を含む整数を示す８ビットの数値であり、(Ｃｒ－１２８)および(Ｃｂ－１２８)は、それぞれ８ビットの整数（－１２８～１２７）の範囲で変化する。

【０１２０】

ここで、下式(７)，(８)の関係が成立するものとして、上式(４)～(６)に下式(７)，(８)を代入して整理すると、下式(９)～(１１)が得られる。

【０１２１】

(Ｃｒ－１２８)＝(Ｃｒ*)　・・・(７)

(Ｃｂ－１２８)＝(Ｃｂ*)　・・・(８)。

【０１２２】

Ｒ＝１．１６４×(Ｙ)＋１．５９６×(Ｃｒ*)－１８．６２４　・・・(９)

Ｇ＝１．１６４×(Ｙ)－０．３９１×(Ｃｂ*)－０．８１３×(Ｃｒ*)－１８．６２４　・・・(１０)

Ｂ＝１．１６４×(Ｙ)＋２．０１８×(Ｃｂ*)－１８．６２４　・・・(１１)。

【０１２３】

更に、上式(９)～(１１)の各定数の表現を十進法から二進法に書き換えると、下式(１２)～(１４)が得られる。

【０１２４】

Ｒ＝１．０００１０１０１０×(Ｙ)＋１．１００１１００１×(Ｃｒ*)－１００１０．１０１０００００　・・・(１２)

Ｇ＝１．０００１０１０１０×(Ｙ)－０．０１１００１００×(Ｃｂ*)－０．１１０１０００００×(Ｃｒ*)－１００１０．１０１０００００　・・・(１３)

Ｂ＝１．０００１０１０１０×(Ｙ)＋１０．０００００１０１×(Ｃｂ*)－１００１０．１０１０００００　・・・(１４)。

【０１２５】

図１５～図１７は、ＷＳ法の乗算アルゴリズムを使用して上式(１２)～(１４)にそれぞれ従った演算を行う過程を示す図である。詳細には、図１５では、上式(１２)に従ってＹ，Ｃｒ*のデータからＲのデータを導出する演算を行う過程が示され、図１６では、上式(１３)に従ってＹ，Ｃｂ*，Ｃｒ*のデータからＧのデータを導出する演算を行う過程が示され、図１７では、上式(１４)に従ってＹ，Ｃｂ*のデータからＢのデータを導出する演算を行う過程が示されている。また、ここでは、(Ｙ)を、二進法の表現で最上位が符号ビットではない８ビットの「(Ｙ７)(Ｙ６)(Ｙ５)(Ｙ４)(Ｙ３)(Ｙ２)(Ｙ１)(Ｙ０)」と表し、Ｙ７～Ｙ０には、０または１の数値が入る。また、(Ｃｒ)を、二進法の表現で最上位が符号ビットである８ビットの「(Ｃｒ７)(Ｃｒ６)(Ｃｒ５)(Ｃｒ４)(Ｃｒ３)(Ｃｒ２)(Ｃｒ１)(Ｃｒ０)」と表し、(Ｃｂ)を、二進法の表現で最上位ビットが符号ビットである８ビットの「(Ｃｂ７)(Ｃｂ６)(Ｃｂ５)(Ｃｂ４)(Ｃｂ３)(Ｃｂ２)(Ｃｂ１)(Ｃｂ０)」と表す。なお、Ｃｒ０～Ｃｒ７およびＣｂ０～Ｃｂ７には、０または１の数値が入る。更に、図１５～図１７では、小数点が入る桁を明確化するために、小数点が入る位置に太破線が

付されている。

【０１２６】

ここで、(Ｃｒ*)および(Ｃｂ*)すなわち(Ｃｒ－１２８)および(Ｃｂ－１２８)については、(Ｃｒ)および(Ｃｂ)と比較して最上位ビットが反転する。例えば、(Ｃｒ)の最上位ビットの数値が"１"である場合には、(Ｃｒ－１２８)は正の値となり、その値を表す整数の符号ビットの値は"０"となる。また、(Ｃｒ)の最上位ビットの数値が"０"である場合には、(Ｃｒ－１２８)は負の値となり、その値を表す整数の符号ビットの値は"１"となる。具体的には、(Ｃｒ*)すなわち(Ｃｒ－１２８)は、二進法の表現では「(ｎｏｔＣｒ７)(Ｃｒ６)(Ｃｒ５)(Ｃｒ４)(Ｃｒ３)(Ｃｒ２)(Ｃｒ１)(Ｃｒ０)」となり、(Ｃｂ*)すなわち(Ｃｂ－１２８)は、二進法の表現では「(ｎｏｔＣｂ７)(Ｃｂ６)(Ｃｂ５)(Ｃｂ４)(Ｃｂ３)(Ｃｂ２)(Ｃｂ１)(Ｃｂ０)」となる。なお、(ｎｏｔＣｒ７)は(Ｃｒ７)のビットを反転したものを示し、(ｎｏｔＣｂ７)は(Ｃｂ７)のビットを反転したものを示す。したがって、ＷＳ法の乗算アルゴリズムの演算では、最上位ビットを負論理として加算を行う場合には、最上位ビットの数値は(Ｃｂ７)および(Ｃｒ７)の正論理で加算が行われることになるが、所定の定数を求めるためのオフセット値の算出については、最上位ビットは負論理の項として扱われる。

【０１２７】

以下、図１５～図１７を参照しつつ、被乗数が正負の双方を取り得るＣｂ，Ｃｒの項についてＷＳ法の乗算アルゴリズムを使用して上式(１２)～(１４)にそれぞれ従った演算を行う過程について説明する。なお、図１５～図１７では、ハッチングが付されず且つ単に太枠で囲まれたビットは、正論理で加算されるが、オフセット値の算出については負論理のビットの項として扱われる。また、ハッチングが付され且つ太線で囲まれたビットは、負論理で加算され、且つオフセット値の算出についても負論理のビットの項として扱われる。更に、ハッチングが付され且つ細線で囲まれたビットは、負論理で加算されるが、オフセット値の算出については正論理のビットの項として扱われる。

【０１２８】

図１５(ａ)では、１～４行目が「＋１．００１０１０１０×(Ｙ)」の項の演算を示し、５～９行目が「１．１００１１００１×(Ｃｒ*)」の項の演算を示し、１０行目がオフセット値を相殺するための所定の定数を示し、１１行目が「１００１０．１０１０００００」の負数を示し、１２行目が小数点以下を四捨五入するために加算する数を示す。そして、図１５(ｂ)では、図１５(ａ)で示すビット列に係る加算のうち、１０～１２行目の定数部分をまとめたものが示されている。

【０１２９】

図１６(ａ)では、１～４行目が「＋１．００１０１０１０×(Ｙ)」の項の演算を示し、５～７行目が「－０．０１１００１００×(Ｃｂ*)」の項の演算を示し、８～１０行目が「－０．１１０１００００×(Ｃｒ*)」の項の演算を示し、１１行目がオフセット値を相殺するための所定の定数を示し、１２行目が「１００１０．１０１０００００」の負数を示し、１３行目が小数点以下を四捨五入するために加算する数を示す。そして、図１６(ｂ)では、図１６(ａ)で示すビット列に係る加算のうち、１１～１３行目の定数部分をまとめたものが示されている。

【０１３０】

図１７(ａ)では、１～４行目が「＋１．００１０１０１０×(Ｙ)」の項の演算を示し、５～７行目が「＋１０．０００００１０１×(Ｃｂ*)」の項の演算を示し、８行目がオフセット値を相殺するための所定の定数を示し、９行目が「１００１０．１０１０００００」の負数を示し、１０行目が小数点以下を四捨五入するために加算する数を示す。そして、図１７(ｂ)では、図１７(ａ)で示すビット列に係る加算のうち、８～１０行目の定数部分をまとめたものが示されている。

【０１３１】

このように、Ｙ，Ｃｒ，ＣｂのデータをＲ，Ｇ，Ｂのデータに変換する際に、ＷＳ法の乗算アルゴリズムを採用することで、入力画像信号に相当するＹ，Ｃｒ，Ｃｂのデータか

ら、表示用信号に相当するＲＧＢのデータに変換するための構成の簡略化と演算速度の高速化とが容易に図られる。このため、画像表示装置３０の小型化および製造コストの低減が図られる。

【０１３２】

＜変形例＞

本発明は上述の実施の形態に限定されるものではなく、本発明の要旨を逸脱しない範囲において種々の変更、改良等が可能である。

【０１３３】

◎例えば、上記実施形態に係る演算装置１では、例えば、論理回路を用いて演算を行うことを基本としていたが、これに限られない。例えば、演算装置１における演算を行う構成の一部または全部の構成がＣＰＵにおいてソフトウエアが実行されることで機能的に実現されても良い。以下、被乗数と乗数との乗算アルゴリズムをソフトウエアを用いて実現する演算装置１Ａを具体例として説明する。

【０１３４】

図１８は、本発明の変形例に係る演算装置１Ａの概略構成を示す図である。図１８で示すように、演算装置１Ａは、ＣＰＵ１００およびメモリ２００をバスラインＢＬで接続した一般的なコンピュータの構成を有する。なお、メモリ２００は、適宜ＲＡＭやＲＯＭやハードディスクなどの記憶部を含む。また、バスラインＢＬは、適宜インターフェース（図示を省略）などを介して表示部３００および操作部４００に接続される。そして、ＣＰＵ１００がメモリ２００に格納されるプログラムに従って動作することにより、演算装置１Ａの各種機能および動作が実現される。

【０１３５】

図１９は、演算装置１Ａにおいて実現される機能的な構成を示すブロック図である。図１９で示すように、演算装置１Ａは、ＣＰＵ１００とメモリ２００との協働により機能的な構成として、被乗数保持部１０１、乗数保持部１０２、シフト反転部１０３、デコーダ部１０４、モードマシン部１０５、タイミング制御部１０６、および加算部１０７を備える。例えば、被乗数保持部１０１、および乗数保持部１０２は、主にメモリ２００などによって実現され、シフト反転部１０３、デコーダ部１０４、モードマシン部１０５、タイミング制御部１０６、および加算部１０７は、主に、ＣＰＵ１００などによって実現される。

【０１３６】

被乗数保持部１０１は、外部からバスラインＢＬを介して入力されてくる被乗数のデータを一時的に保持し、該被乗数のデータが適宜シフト反転部１０３によって読み出される。

【０１３７】

乗数保持部１０２は、外部からバスラインＢＬを介して入力されてくる乗数のデータを一時的に保持する。また、乗数保持部１０２では、乗数のデータのビット数に応じて、符号拡張が行われる。そして、該乗数のビット列が２ビットずつ右（すなわち下位側）に順次にシフトされる毎に、該乗数のビット列から連続する３ビットの数値がデコーダ部１０４に読み出される。

【０１３８】

シフト反転部１０３は、被乗数を＋２倍した数値（以下「＋２倍被乗数」と称する）のデータ、被乗数を－１倍した数値（以下「－１倍被乗数」と称する）のデータ、および被乗数を－２倍した数値（以下「－２倍被乗数」と称する）のデータを生成する。つまり、シフト反転部１０３は、＋２倍被乗数、被乗数の＋１倍の数値（以下「＋１倍被乗数」と称する）、－１倍被乗数、および－２倍被乗数の各データを準備する。また、シフト反転部１０３は、デコーダ部１０４からの指令に従って、＋２倍被乗数、＋１倍被乗数、－１倍被乗数、および－２倍被乗数のうちの何れかの数値のデータを加算部１０７に対して出力する。

【０１３９】

デコーダ部１０４は、乗数保持部１０２から読み出した連続する３ビットの数値の配列パターン（ビットパターン）に応じて、モードマシン部１０５に対して、加算基調の演算を行うモード（以下「モード０」と称する）と減算基調の演算を行うモード（以下「モード１」と称する）との間でモードを切り替えるための指令を送出する。また、デコーダ部１０４は、モードマシン部１０５によって設定されているモードを参照しつつ、乗数保持部１０２から読み出した連続する３ビットの数値の配列パターン（ビットパターン）に応じて、＋２倍被乗数、＋１倍被乗数、－１倍被乗数、および－２倍被乗数のうちの何れの数値の加算を行うのかを決定し、決定した結果に応じた指令をシフト反転部１０３に出力する。

【０１４０】

モードマシン部１０５は、デコーダ部１０４からの指令に従って、モード０とモード１との間でモードを切り替えて、該モードの設定状態を示す情報を保持する。

【０１４１】

タイミング制御部１０６は、スタートパルスに応じて、乗数および被乗数のデータの入力のタイミングと、各構成の動作とが適宜同期するように、演算装置１Ａの動作を制御する。

【０１４２】

加算部１０７は、シフト反転部１０３から出力される＋２倍被乗数、＋１倍被乗数、－１倍被乗数、および－２倍被乗数のうちの何れかの数値を順次に加算する。但し、加算部１０７は、デコーダ部１０４において連続する３ビットの数値の配列パターン（ビットパターン）が認識されるごとに、それまでの加算結果を示すビット列が左（すなわち上位側）に２ビットずつシフトされる。なお、加算部１０７における加算では、最上位ビットの桁が揃うように、＋２倍被乗数、＋１倍被乗数、－１倍被乗数、および－２倍被乗数の符号拡張が行われる。

【０１４３】

図２０から図２５は、演算装置１Ａにおける演算処理フローを示すフローチャートである。本演算処理フローは、演算装置１Ａへの被乗数および乗数のデータの入力に応答して開始される。

【０１４４】

ステップＳ２０１では、乗数保持部１０２により、乗数のデータのビット数が２ｎ＋１（ｎは自然数）であるか否か判定される。ここでは、ビット数が２ｎ＋１であれば、ステップＳ２０２に進み、ビット数が２ｎ＋１でなければ、ステップＳ２０３に進む。

【０１４５】

ステップＳ２０２では、乗数保持部１０２により、乗数について２ビット分符号拡張が行われ、ステップＳ２０４に進む。ここで符号拡張された乗数を「符号拡張乗数」と称する。

【０１４６】

ステップＳ２０３では、乗数保持部１０２により、乗数について１ビット分符号拡張が行われ、ステップＳ２０４に進む。ここで符号拡張された乗数を「符号拡張乗数」と称する。

【０１４７】

ステップＳ２０４では、シフト反転部１０３により、被乗数保持部１０１から被乗数のデータが読み出されて、＋２倍被乗数、＋１倍被乗数、－１倍被乗数、および－２倍被乗数がそれぞれ算出される。

【０１４８】

ステップＳ２０５では、モードマシン部１０５により、デフォルトのモードとして、加算基調の演算を行うモード０が設定される。

【０１４９】

ステップＳ２０６では、デコーダ部１０４により、符号拡張乗数の最下位から３ビットが判定対象として設定される。

【０１５０】
ステップＳ２０７では、デコーダ部１０４により、判定対象である３ビットの数値の配列パターン（ビットパターン）が"０００"であるか否か判定される。ここで、ビットパターンが"０００"であれば、加減算を行うことなく、図２２のステップＳ２３１に進み、ビットパターンが"０００"でなければ、ステップＳ２０８に進む。
【０１５１】
ステップＳ２０８では、デコーダ部１０４により、判定対象である３ビットの数値の配列パターン（ビットパターン）が"００１"であるか否か判定される。ここで、ビットパターンが"００１"であれば、ステップＳ２０９に進み、ビットパターンが"００１"でなければ、図２１のステップＳ２１１に進む。
【０１５２】
ステップＳ２０９では、シフト反転部１０３から加算部１０７に対して＋１倍被乗数のデータが出力され、加算部１０７により、＋１倍被乗数が符号拡張されつつ加算される。
【０１５３】
図２１のステップＳ２１１では、デコーダ部１０４により、判定対象である３ビットの数値の配列パターン（ビットパターン）が"０１０"であるか否か判定される。ここで、ビットパターンが"０１０"であれば、ステップＳ２１２に進み、ビットパターンが"０１０"でなければ、ステップＳ２１３に進む。
【０１５４】
ステップＳ２１２では、シフト反転部１０３から加算部１０７に対して＋２倍被乗数のデータが出力され、加算部１０７により、＋２倍被乗数が符号拡張されつつ加算されて、図２２のステップＳ２３１に進む。
【０１５５】
ステップＳ２１３では、デコーダ部１０４により、判定対象である３ビットの数値の配列パターン（ビットパターン）が"０１１"であるか否か判定される。ここで、ビットパターンが"０１１"であれば、ステップＳ２１４に進み、ビットパターンが"０１１"でなければ、ステップＳ２１６に進む。
【０１５６】
ステップＳ２１４では、モードマシン部１０５により、加算基調の演算を行うモード０から減算基調の演算を行うモード１にモード設定が変更される。
【０１５７】
ステップＳ２１５では、シフト反転部１０３から加算部１０７に対して－１倍被乗数のデータが出力され、加算部１０７により、－１倍被乗数が符号拡張されつつ加算され、図２３のステップＳ２３４に進む。なお、ここでは、－１倍被乗数の加算は、＋１倍被乗数の減算と同一の演算となる。
【０１５８】
ステップＳ２１６では、デコーダ部１０４により、判定対象である３ビットの数値の配列パターン（ビットパターン）が"１００"であるか否か判定される。ここで、ビットパターンが"１００"であれば、加減算を行うことなく、図２２のステップＳ２３１に進み、ビットパターンが"１００"でなければ、ステップＳ２１７に進む。
【０１５９】
ステップＳ２１７では、デコーダ部１０４により、判定対象である３ビットの数値の配列パターン（ビットパターン）が"１０１"であるか否か判定される。ここで、ビットパターンが"１０１"であれば、ステップＳ２１８に進み、ビットパターンが"１０１"でなければ、ステップＳ２１９に進む。
【０１６０】
ステップＳ２１８では、シフト反転部１０３から加算部１０７に対して＋１倍被乗数のデータが出力され、加算部１０７により、＋１倍被乗数が符号拡張されつつ加算され、図２２のステップＳ２３１に進む。
【０１６１】

ステップＳ２１９では、デコーダ部１０４により、判定対象である３ビットの数値の配列パターン（ビットパターン）が"１１０"であるか否か判定される。ここで、ビットパターンが"１１０"であれば、ステップＳ２２０に進み、ビットパターンが"１１０"でなければ、ステップＳ２２２に進む。なお、ステップＳ２２２に進む場合には、判定対象の３ビットのビットパターンが"１１１"であることを意味する。

【０１６２】

ステップＳ２２０では、ステップＳ２１４と同様に、モードマシン部１０５により、加算基調の演算を行うモード０から減算基調の演算を行うモード１にモード設定が変更される。

【０１６３】

ステップＳ２２１では、シフト反転部１０３から加算部１０７に対して－２倍被乗数のデータが出力され、加算部１０７により、－２倍被乗数が符号拡張されつつ加算され、図２３のステップＳ２３４に進む。なお、ここでは、－２倍被乗数の加算は、＋２倍被乗数の減算と同一の演算となる。

【０１６４】

ステップＳ２２２では、ステップＳ２１４と同様に、モードマシン部１０５により、加算基調の演算を行うモード０から減算基調の演算を行うモード１にモード設定が変更される。

【０１６５】

ステップＳ２２３では、ステップＳ２１５と同様に、シフト反転部１０３から加算部１０７に対して－１倍被乗数のデータが出力され、加算部１０７により、－１倍被乗数が符号拡張されつつ加算され、図２３のステップＳ２３４に進む。なお、ここでは、－１倍被乗数の加算は、＋１倍被乗数の減算と同一の演算となる。

【０１６６】

図２２のステップＳ２３１では、ステップＳ２０２，Ｓ２０３で符号拡張されたビットも含めた符号拡張乗数の全ビットについてのビットパターンの判定が終了しているか否か判定される。ここでは、符号拡張乗数の全ビットについてのビットパターンの判定が終了していれば、本動作フローが終了され、符号拡張乗数の全ビットについてのビットパターンの判定が終了していなければ、ステップＳ２３２に進む。

【０１６７】

ステップＳ２３２では、乗数保持部１０２により、符号拡張乗数のビット列が２ビット右（すなわち下位側）にシフトされて、デコーダ部１０４による判定対象の３ビットが２ビット左（すなわち上位側）に設定される。つまり、判定対象の３ビットが、符号拡張乗数の２ビット分上位側に変更される。

【０１６８】

ステップＳ２３３では、加算部１０７により、これまでの加算結果を示すビット列が相対的に右（すなわち下位側）に２ビット分シフトされて、図２０のステップＳ２０７に進む。

【０１６９】

図２３のステップＳ２３４では、ステップＳ２０２，Ｓ２０３で符号拡張されたビットも含めた符号拡張乗数の全ビットについてのビットパターンの判定が終了しているか否か判定される。ここでは、符号拡張乗数の全ビットについてのビットパターンの判定が終了していれば、本動作フローが終了され、符号拡張乗数の全ビットについてのビットパターンの判定が終了していなければ、ステップＳ２３５に進む。

【０１７０】

ステップＳ２３５では、乗数保持部１０２により、符号拡張乗数のビット列が２ビット右（すなわち下位側）にシフトされて、デコーダ部１０４による判定対象の３ビットが２ビット左（すなわち上位側）に設定される。つまり、判定対象の３ビットが、符号拡張乗数の２ビット分上位側に変更される。

【０１７１】

ステップＳ２３６では、加算部１０７により、これまでの加算結果を示すビット列が相対的に左（すなわち下位側）に２ビット分シフトされて、図２４のステップＳ２４１に進む。

【０１７２】

なお、ステップＳ２３１，Ｓ２３４で本演算処理フローが終了される場合には、加算部１０７において算出されている加算結果が、被乗数と乗数との乗算結果として、加算部１０７から出力される。

【０１７３】

図２４のステップＳ２４１では、デコーダ部１０４により、判定対象である３ビットの数値の配列パターン（ビットパターン）が"０００"であるか否か判定される。ここで、ビットパターンが"０００"であれば、ステップＳ２４２に進み、ビットパターンが"０００"でなければ、ステップＳ２４４に進む。

【０１７４】

ステップＳ２４２では、モードマシン部１０５により、減算基調の演算を行うモード１から加算基調の演算を行うモード０にモード設定が変更される。

【０１７５】

ステップＳ２４３では、ステップＳ２１８と同様に、シフト反転部１０３から加算部１０７に対して＋１倍被乗数のデータが出力され、加算部１０７により、＋１倍被乗数が符号拡張されつつ加算され、図２２のステップＳ２３１に進む。

【０１７６】

ステップＳ２４４では、デコーダ部１０４により、判定対象である３ビットの数値の配列パターン（ビットパターン）が"００１"であるか否か判定される。ここで、ビットパターンが"００１"であれば、ステップＳ２４５に進み、ビットパターンが"００１"でなければ、ステップＳ２４７に進む。

【０１７７】

ステップＳ２４５では、ステップＳ２４２と同様に、モードマシン部１０５により、減算基調の演算を行うモード１から加算基調の演算を行うモード０にモード設定が変更される。

【０１７８】

ステップＳ２４６では、ステップＳ２１２と同様に、シフト反転部１０３から加算部１０７に対して＋２倍被乗数のデータが出力され、加算部１０７により、＋２倍被乗数が符号拡張されつつ加算され、図２２のステップＳ２３１に進む。

【０１７９】

ステップＳ２４７では、デコーダ部１０４により、判定対象である３ビットの数値の配列パターン（ビットパターン）が"０１０"であるか否か判定される。ここで、ビットパターンが"０１０"であれば、ステップＳ２４８に進み、ビットパターンが"０１０"でなければ、図２５のステップＳ２５１に進む。

【０１８０】

ステップＳ２４８では、シフト反転部１０３から加算部１０７に対して－１倍被乗数のデータが出力され、加算部１０７により、－１倍被乗数が符号拡張されつつ加算され、図２３のステップＳ２３４に進む。なお、ここでは、－１倍被乗数の加算は、＋１倍被乗数の減算と同一の演算となる。

【０１８１】

図２５のステップＳ２５１では、デコーダ部１０４により、判定対象である３ビットの数値の配列パターン（ビットパターン）が"０１１"であるか否か判定される。ここで、ビットパターンが"０１１"であれば、加減算を行うことなく、図２３のステップＳ２３４に進み、ビットパターンが"０１１"でなければ、ステップＳ２５２に進む。

【０１８２】

ステップＳ２５２では、デコーダ部１０４により、判定対象である３ビットの数値の配列パターン（ビットパターン）が"１００"であるか否か判定される。ここで、ビットパタ

ーンが"１００"であれば、ステップＳ２５３に進み、ビットパターンが"１００"でなければ、ステップＳ２５５に進む。

【０１８３】

ステップＳ２５３では、ステップＳ２４２と同様に、モードマシン部１０５により、減算基調の演算を行うモード１から加算基調の演算を行うモード０にモード設定が変更される。

【０１８４】

ステップＳ２５４では、ステップＳ２４３と同様に、シフト反転部１０３から加算部１０７に対して＋１倍被乗数のデータが出力され、加算部１０７により、＋１倍被乗数が符号拡張されつつ加算され、図２２のステップＳ２３１に進む。

【０１８５】

ステップＳ２５５では、デコーダ部１０４により、判定対象である３ビットの数値の配列パターン（ビットパターン）が"１０１"であるか否か判定される。ここで、ビットパターンが"１０１"であれば、ステップＳ２５６に進み、ビットパターンが"１０１"でなければ、ステップＳ２５７に進む。

【０１８６】

ステップＳ２５６では、シフト反転部１０３から加算部１０７に対して－２倍被乗数のデータが出力され、加算部１０７により、－２倍被乗数が符号拡張されつつ加算され、図２３のステップＳ２３４に進む。なお、ここでは、－２倍被乗数の加算は、＋２倍被乗数の減算と同一の演算となる。

【０１８７】

ステップＳ２５７では、デコーダ部１０４により、判定対象である３ビットの数値の配列パターン（ビットパターン）が"１１０"であるか否か判定される。ここで、ビットパターンが"１１０"であれば、ステップＳ２５８に進み、ビットパターンが"１１０"でなければ、加減算を行うことなく、図２３のステップＳ２３４に進む。

【０１８８】

ステップＳ２５８では、シフト反転部１０３から加算部１０７に対して－１倍被乗数のデータが出力され、加算部１０７により、－１倍被乗数が符号拡張されつつ加算され、図２３のステップＳ２３４に進む。なお、ここでは、－１倍被乗数の加算は、＋１倍被乗数の減算と同一の演算となる。

【０１８９】

以上のように、被乗数と乗数との乗算アルゴリズムをソフトウエアを用いて実現する演算装置１Ａについて説明したが、演算装置１Ａでは、ビットを反転させる負論理や所定の定数の加算などを行わない。これは、ソフトウエアによってＣＰＵで実現される演算では、ビットを反転させる演算のルーチンや、所定の定数の加算を行うための演算のルーチンが別途必要になるため、これらの演算については、行われない方が演算速度や演算による負荷低減を図る上で好ましい。したがって、演算装置１Ａでは、被乗数の符号拡張を行いつつ、被乗数に係る加算および減算が行われることで、演算の簡略化が図られて、被乗数と乗数との乗算結果が求められるように構成される。

【０１９０】

これに対して、上記実施形態に係る演算装置１の様に、例えば、論理回路を用いて演算を行うことを基本とする場合には、ビットを反転させる演算は、ＮＡＮＤ回路などによって容易に実現され、演算に対する論理圧縮も可能である。このため、所定の定数を算出して、加算する演算も容易に実現可能である。また、所定の定数を加算する方式では、加減算する被乗数の符号拡張が不要となり、論理回路の構成の複雑化も招かない。したがって、上記実施形態に係る演算装置１の様に、例えば、論理回路を用いて演算を行うことを基本とする場合には、ビットの反転や、所定の定数の加算を行う演算を採用することで、被乗数の符号拡張を行うことなく、乗算を行うことが可能となり、演算を行う部分の構成の簡略化が図られる。

【０１９１】

◎また、上記実施形態に係る画像表示装置３０では、明度および色差を示す信号であるＹ，Ｃｒ，Ｃｂの信号を、所定のルールに従ってＲＧＢの各色の階調を示す信号に変換する演算において、ＷＳ法の乗算アルゴリズムを適用する例を挙げて説明したが、これに限られない。例えば、温度の変化に対して、いわゆるγ変換のためのテーブルを補正する演算に、ＷＳ法の乗算アルゴリズムを適用するようにしても良い。

【０１９２】

◎また、上記実施形態に係る画像表示装置３０では、ＲＧＢの３原色の光の組合せによって各種画像が表現されたが、これに限られず、例えば、イエロー(Ｙ)、マゼンタ(Ｍ)、シアン(Ｃ)などのその他の複数の色の組合せによって各種画像が表現されても良い。なお、このような構成においても、上記実施形態と同様に、明度および色差を示す信号を所定のルールに従って各色の階調を示す信号に変換するための演算にＷＳ法の乗算アルゴリズムを適用すれば良い。

【図面の簡単な説明】

【０１９３】

【図１】本発明の実施形態に係る演算装置１の機能構成を示すブロック図である。

【図２】本発明の変形例に係る演算装置１ａの機能構成を示すブロック図である。

【図３】演算装置１における演算処理フローを示すフローチャートである。

【図４】演算装置１における演算処理フローを示すフローチャートである。

【図５】演算装置１における演算処理フローを示すフローチャートである。

【図６】演算装置１における計算例を説明するための図である。

【図７】本発明の実施形態のバリエーションに係る演算処理フローを示すフローチャートである。

【図８】本発明の実施形態のバリエーションに係る演算処理フローを示すフローチャートである。

【図９】本発明の実施形態のバリエーションに係る演算処理フローを示すフローチャートである。

【図１０】演算装置１における被乗数に係る加減算の回数と、従来技術における被乗数に係る加減算の回数とを比較するための図である。

【図１１】被乗数の加減算と定数加算とを行う演算態様を示すイメージ図である。

【図１２】映像システムを例示する図である。

【図１３】画像表示装置の機能構成を示すブロック図である。

【図１４】ＲＧＢ変換部におけるデータ変換について説明するための図である。

【図１５】Ｒ色に係るデータ変換を説明するための図である。

【図１６】Ｇ色に係るデータ変換を説明するための図である。

【図１７】Ｂ色に係るデータ変換を説明するための図である。

【図１８】本発明の変形例に係る演算装置１Ａの概略構成を示す図である。

【図１９】本発明の変形例に係る演算装置１Ａの機能構成を示すブロック図である。

【図２０】演算装置１Ａにおける演算処理フローを示すフローチャートである。

【図２１】演算装置１Ａにおける演算処理フローを示すフローチャートである。

【図２２】演算装置１Ａにおける演算処理フローを示すフローチャートである。

【図２３】演算装置１Ａにおける演算処理フローを示すフローチャートである。

【図２４】演算装置１Ａにおける演算処理フローを示すフローチャートである。

【図２５】演算装置１Ａにおける演算処理フローを示すフローチャートである。

【図２６】ブースの乗算アルゴリズムの演算処理フローを示すフローチャートである。

【図２７】十進法の数値に対応する二進法の４ビットの整数表記を示す図である。

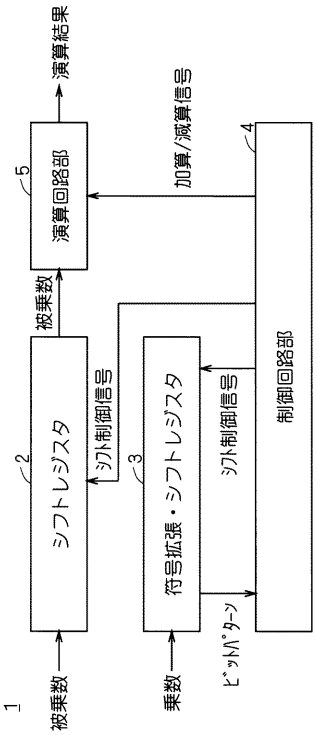
【図２８】ブースの乗算アルゴリズムによる計算例を説明するための図である。

【図２９】定数加算法の乗算アルゴリズムの演算処理フローを示すフローチャートである。

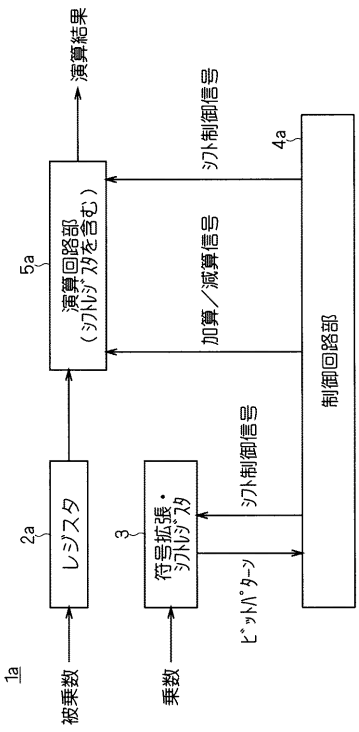
【図３０】定数加算法の乗算アルゴリズムの演算処理フローを示すフローチャートである。

【図３１】定数加算法の乗算アルゴリズムによる計算例を説明するための図である。
【図３２】２の補数の定数加算法の乗算アルゴリズムの演算処理フローを示すフローチャートである。
【図３３】２の補数の定数加算法の乗算アルゴリズムの演算処理フローを示すフローチャートである。
【図３４】２の補数の定数加算法の乗算アルゴリズムによる計算例を説明するための図である。
【符号の説明】
【０１９４】
１，１ａ，１Ａ　演算装置
２　シフトレジスタ
２ａ　レジスタ
３　符号拡張・シフトレジスタ
４，４ａ　制御回路部
５，５ａ　演算回路部
３０　画像表示装置
３１　信号変換部
３２　ＲＧＢ変換部
３３　タイミング制御部
１００　ＣＰＵ
１０１　被乗数保持部
１０２　乗数保持部
１０３　シフト反転部
１０４　デコーダ部
１０５　モードマシン部
１０６　タイミング制御部
１０７　加算部
２００　メモリ
３００　表示部
４００　操作部
ＢＬ　バスライン
ＤＰ　表示パネル
Ｘｄ　Ｘドライバ
Ｙｄ　Ｙドライバ

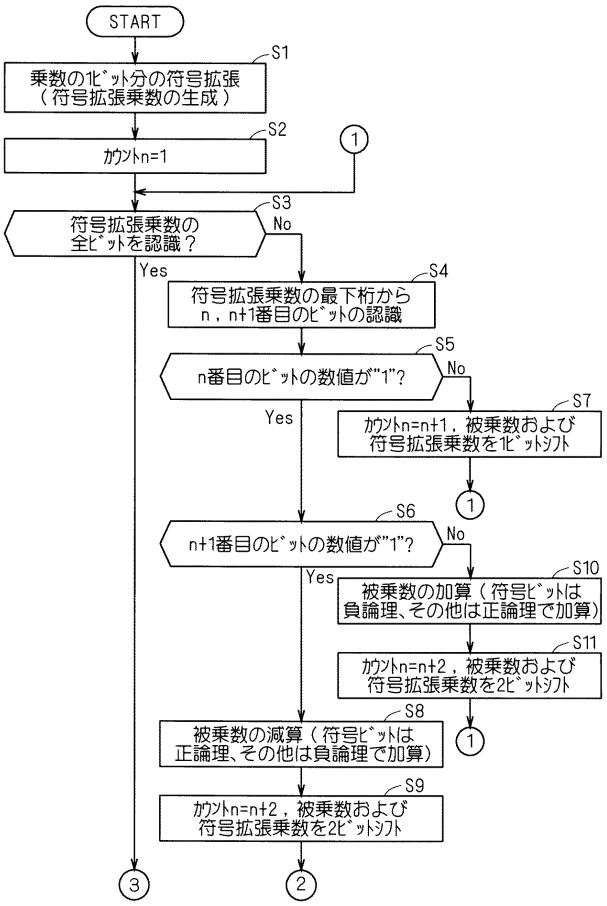
【図１】


演算結果
加算/減算信号
4
5
演算回路部
被乗数
シフトレジスタ
2
シフト制御信号
3
符号拡張・シフトレジスタ
シフト制御信号
制御回路部
1
被乗数
乗数
ビットパターン

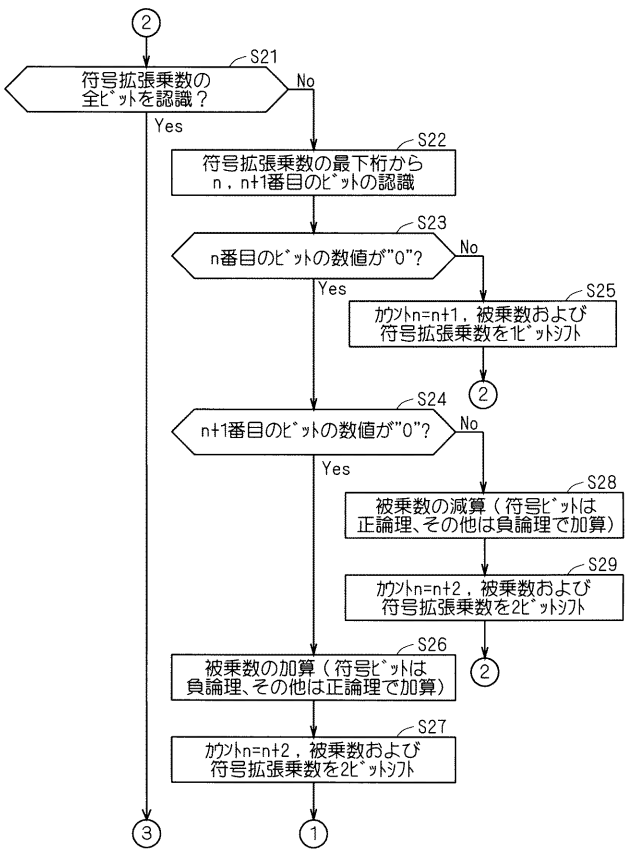
【図２】


演算結果
シフト制御信号
4a
5a
演算回路部
（シフトレジスタを含む）
加算／減算信号
制御回路部
2a
レジスタ
3
符号拡張・
シフトレジスタ
シフト制御信号
ビットパターン
1a
被乗数
乗数

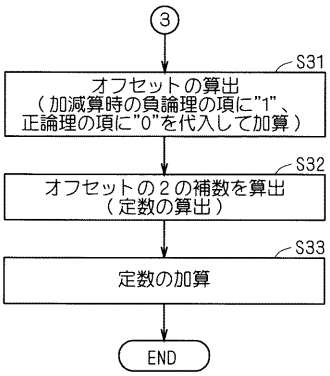
【図３】


START
S1
乗数のnビット分の符号拡張
（符号拡張乗数の生成）
S2
カウントn=1
S3
符号拡張乗数の
全ビットを認識？
No
Yes
S4
符号拡張乗数の最下桁から
n，n+1番目のビットの認識
S5
n番目のビットの数値が"1"?
No
Yes
S7
カウントn=n+1，被乗数および
符号拡張乗数をnビットシフト
S6
n+1番目のビットの数値が"1"?
No
Yes
S10
被乗数の加算（符号ビットは
負論理、その他は正論理で加算）
S11
カウントn=n+2，被乗数および
符号拡張乗数を2ビットシフト
S8
被乗数の減算（符号ビットは
正論理、その他は負論理で加算）
S9
カウントn=n+2，被乗数および
符号拡張乗数を2ビットシフト
1
2
3

【図４】


2
S21
符号拡張乗数の
全ビットを認識？
No
Yes
S22
符号拡張乗数の最下桁から
n，n+1番目のビットの認識
S23
n番目のビットの数値が"0"?
No
Yes
S25
カウントn=n+1，被乗数および
符号拡張乗数をnビットシフト
2
S24
n+1番目のビットの数値が"0"?
No
Yes
S28
被乗数の減算（符号ビットは
正論理、その他は負論理で加算）
S29
カウントn=n+2，被乗数および
符号拡張乗数を2ビットシフト
2
S26
被乗数の加算（符号ビットは
負論理、その他は正論理で加算）
S27
カウントn=n+2，被乗数および
符号拡張乗数を2ビットシフト
3
1

【図５】



【図６】

（ａ）

				0	1	0	1
			×	1	0	1	0
			1	1	0	1	
	0	0	1	0			
0	1	0	1	0	1	0	

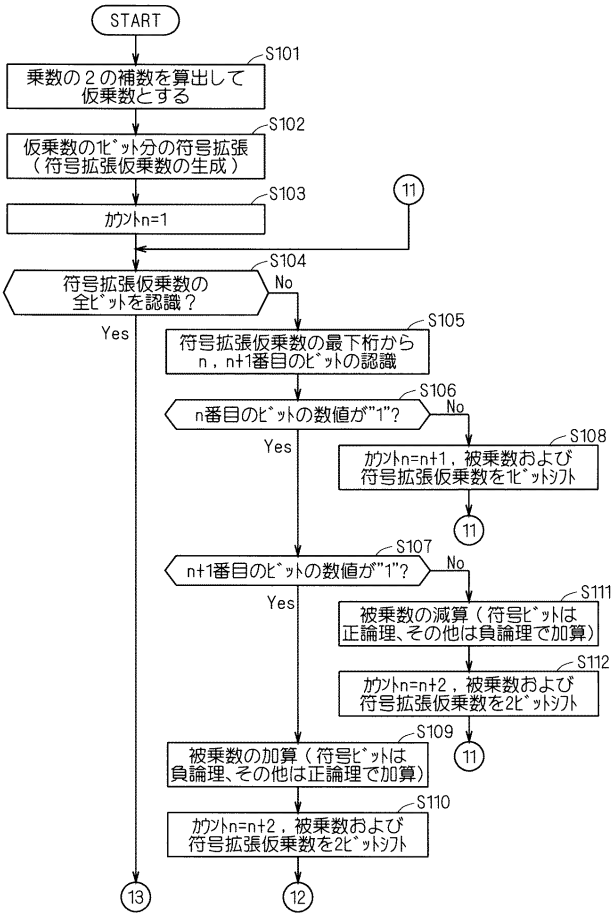
（ｂ）

				a3	a2	a1	a0
			×	b3	b2	b1	b0
			$\overline{a3}$	$\overline{a2}$	a1	a0	
+	a3	$\overline{a2}$	$\overline{a1}$	$\overline{a0}$			
			1	0	0	0	
	0	1	1	1			
1	0	0	1	0	0	0	

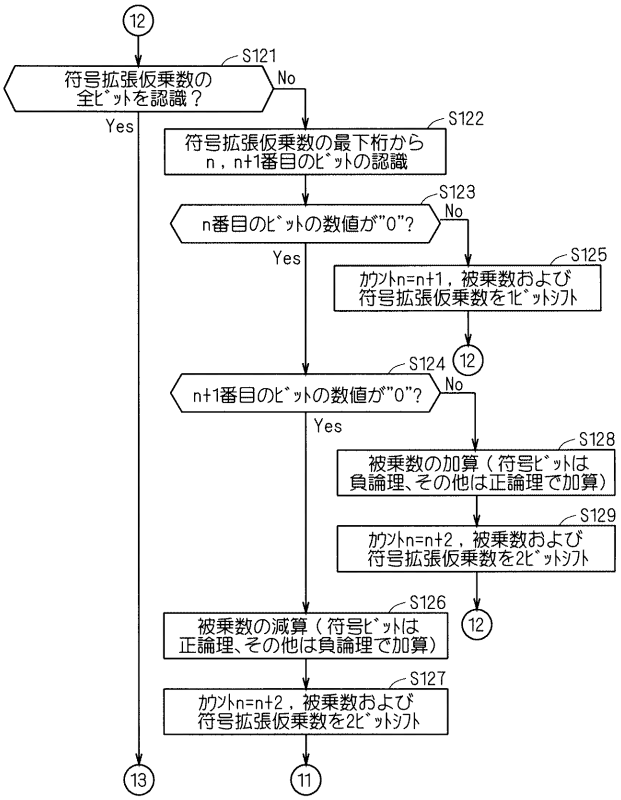
（ｃ）

		0	1	0	1	0	1	0
+	1	0	1	1	1	0	0	0
1	1	1	0	0	0	1	0	

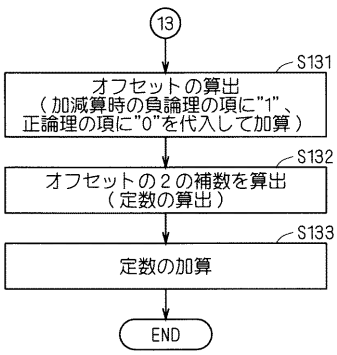
【図７】



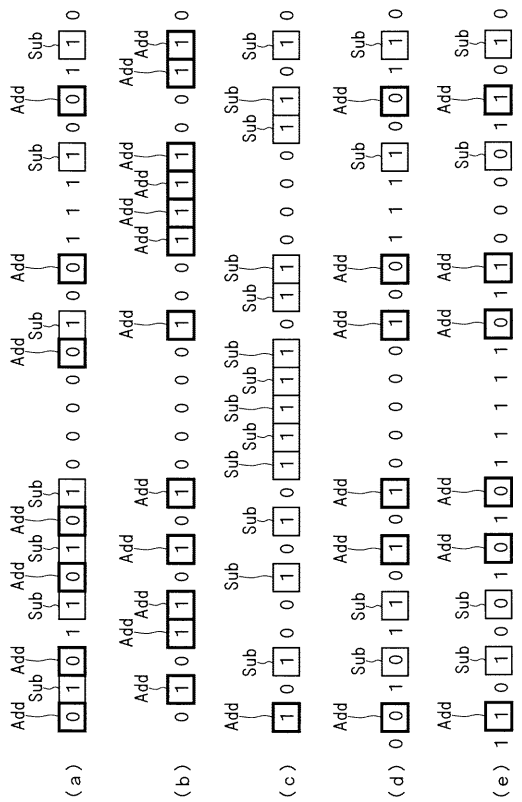
【図８】



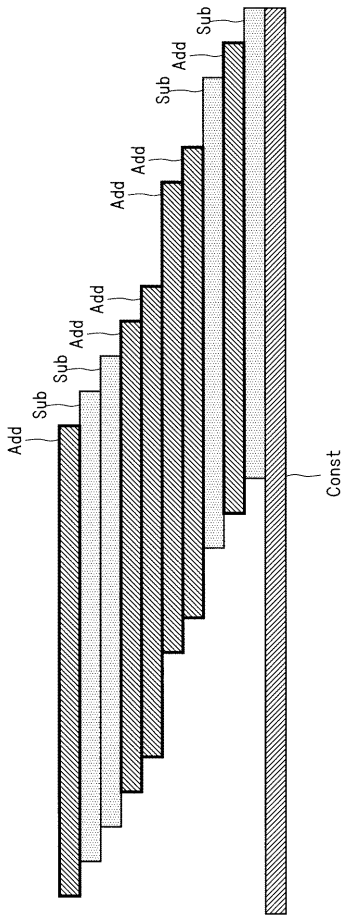
【図９】



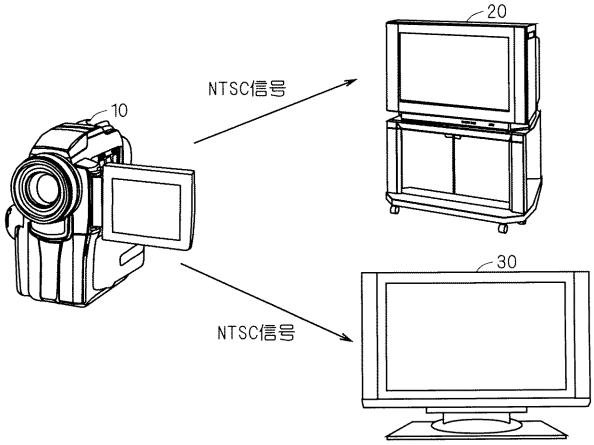
【図１０】



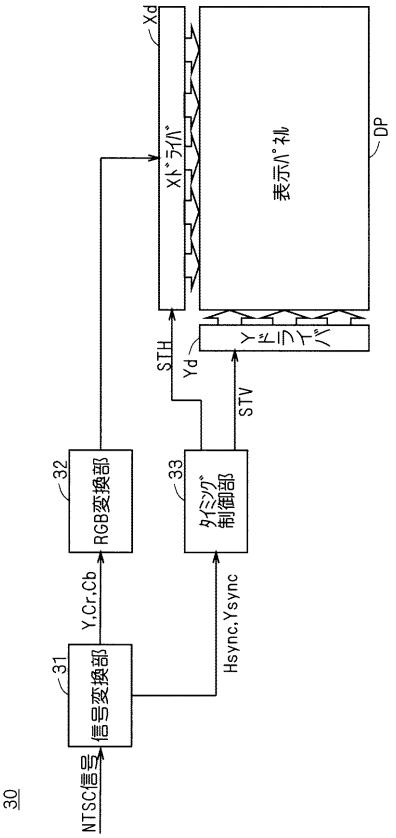
【図１１】



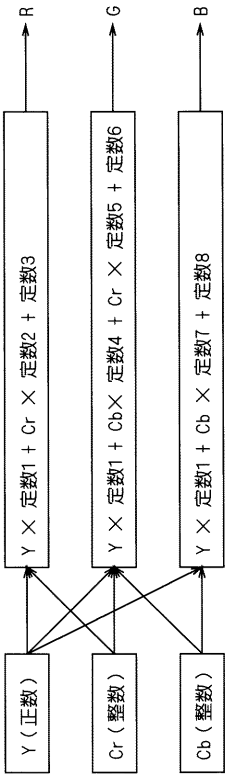
【図１２】



【図１３】



【図１４】



【図１５】

（ａ）

		Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0							
					Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0				
							Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0		
									Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0
	Cr7	Cr6	Cr5	Cr4	Cr3	Cr2	Cr1	Cr0								
					Cr7	Cr6	Cr5	Cr4	Cr3	Cr2	Cr1	Cr0				
							Cr7	Cr6	Cr5	Cr4	Cr3	Cr2	Cr1	Cr0		
									Cr7	Cr6	Cr5	Cr4	Cr3	Cr2	Cr1	Cr0
1	0	1	0	1	0	1	1	0	0	0	0	0	0	1		
1	1	1	1	1	0	1	1	0	1	0	1	1				
0	0	0	0	0	0	0	0	0	0	1						

（ｂ）

		Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0							
					Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0				
							Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0		
									Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0
	Cr7	Cr6	Cr5	Cr4	Cr3	Cr2	Cr1	Cr0								
					Cr7	Cr6	Cr5	Cr4	Cr3	Cr2	Cr1	Cr0				
							Cr7	Cr6	Cr5	Cr4	Cr3	Cr2	Cr1	Cr0		
									Cr7	Cr6	Cr5	Cr4	Cr3	Cr2	Cr1	Cr0
1	0	1	0	0	1	1	0	0	1	1	1	1	0	1		

【図１６】

（ａ）

		Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0							
					Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0				
							Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0		
									Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0
		Cb7	Cb6	Cb5	Cb4	Cb3	Cb2	Cb1	Cb0							
					Cb7	Cb6	Cb5	Cb4	Cb3	Cb2	Cb1	Cb0				
							Cb7	Cb6	Cb5	Cb4	Cb3	Cb2	Cb1	Cb0		
	Cr7	Cr6	Cr5	Cr4	Cr3	Cr2	Cr1	Cr0								
			Cr7	Cr6	Cr5	Cr4	Cr3	Cr2	Cr1	Cr0						
					Cr7	Cr6	Cr5	Cr4	Cr3	Cr2	Cr1	Cr0				
1	1	0	0	0	0	0	1	1	1	1	0	0	1	0	1	
1	1	1	1	1	0	1	1	0	1	0	1	1				
0	0	0	0	0	0	0	0	0	0	1						

（ｂ）

		Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0							
					Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0				
							Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0		
									Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0
		Cb7	Cb6	Cb5	Cb4	Cb3	Cb2	Cb1	Cb0							
					Cb7	Cb6	Cb5	Cb4	Cb3	Cb2	Cb1	Cb0				
							Cb7	Cb6	Cb5	Cb4	Cb3	Cb2	Cb1	Cb0		
	Cr7	Cr6	Cr5	Cr4	Cr3	Cr2	Cr1	Cr0								
			Cr7	Cr6	Cr5	Cr4	Cr3	Cr2	Cr1	Cr0						
					Cr7	Cr6	Cr5	Cr4	Cr3	Cr2	Cr1	Cr0				
1	0	1	1	1	1	0	1	0	1	0	1	1	1	0	1	

【図１７】

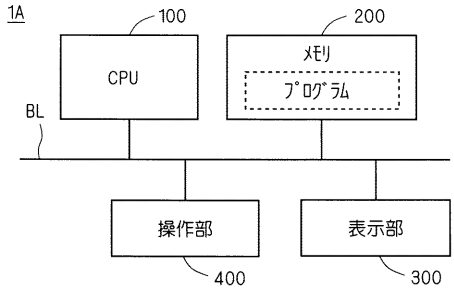
（ａ）

		Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0							
					Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0				
							Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0		
									Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0
	Cb7	Cb6	Cb5	Cb4	Cb3	Cb2	Cb1	Cb0								
							Cb7	Cb6	Cb5	Cb4	Cb3	Cb2	Cb1	Cb0		
									Cb7	Cb6	Cb5	Cb4	Cb3	Cb2	Cb1	Cb0
1	0	1	1	1	0	1	1	0	1	1	1	1				
1	1	1	1	1	0	1	1	0	1	0	1	1				
0	0	0	0	0	0	0	0	0	0	1						

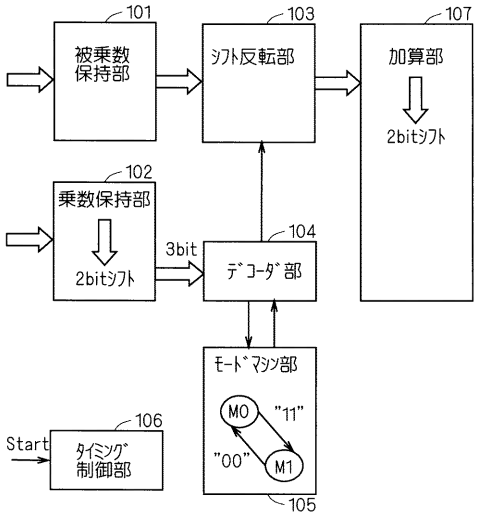
（ｂ）

		Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0							
					Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0				
							Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0		
									Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0
	Cb7	Cb6	Cb5	Cb4	Cb3	Cb2	Cb1	Cb0								
							Cb7	Cb6	Cb5	Cb4	Cb3	Cb2	Cb1	Cb0		
									Cb7	Cb6	Cb5	Cb4	Cb3	Cb2	Cb1	Cb0
1	0	1	1	1	0	1	0	1	1	0	1	1				

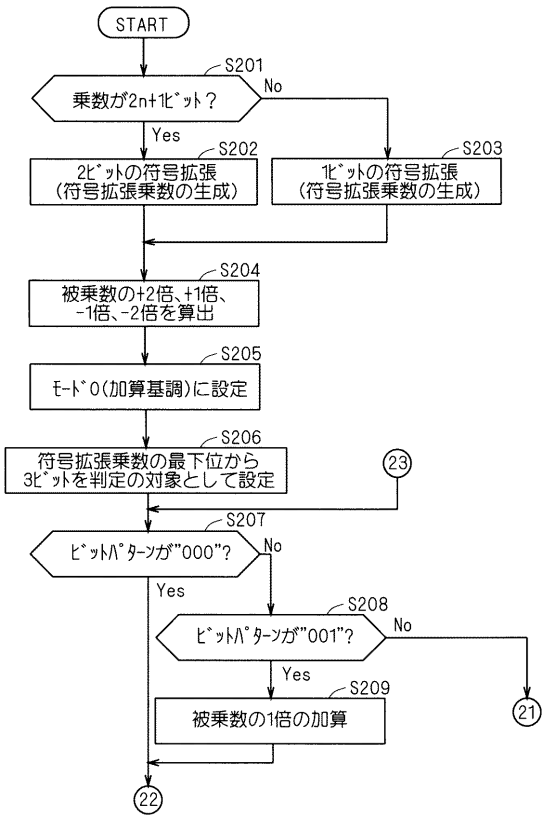
【図１８】



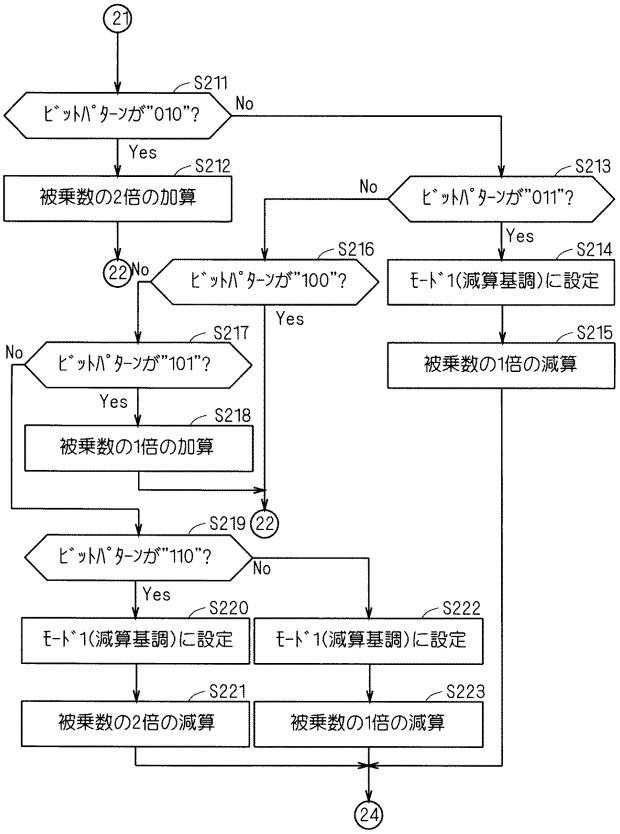
【図１９】



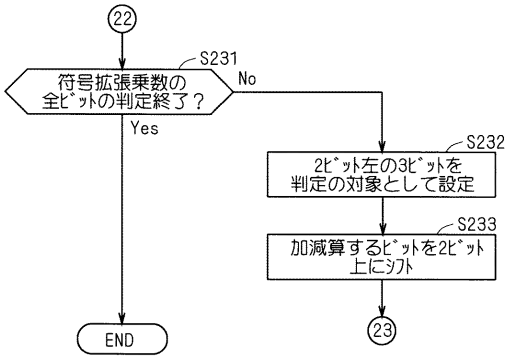
【図２０】



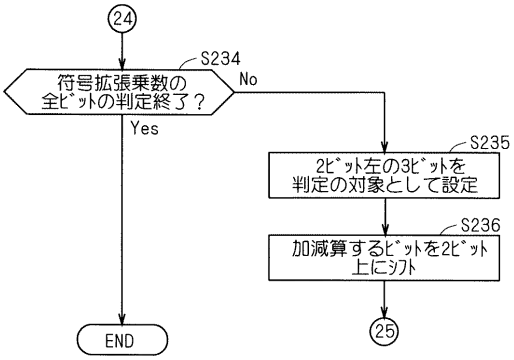
【図２１】



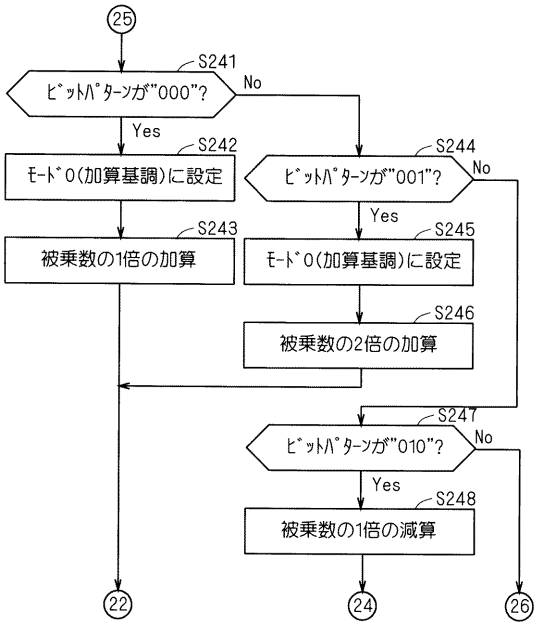
【図２２】



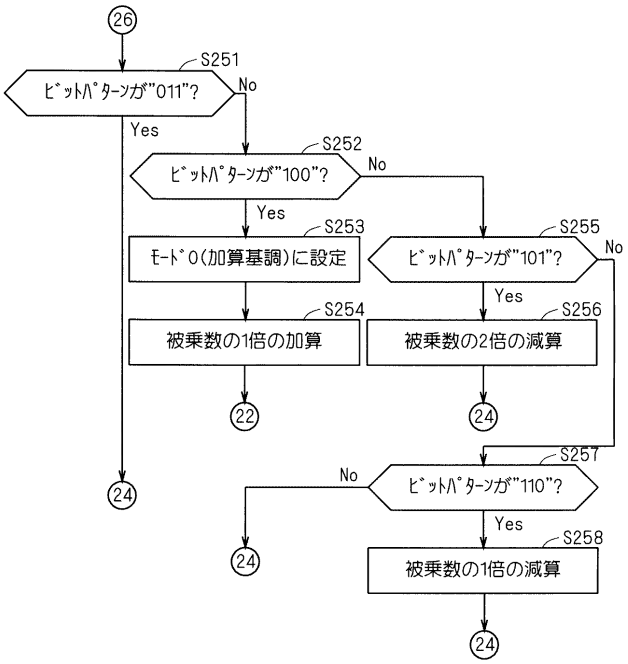
【図２３】



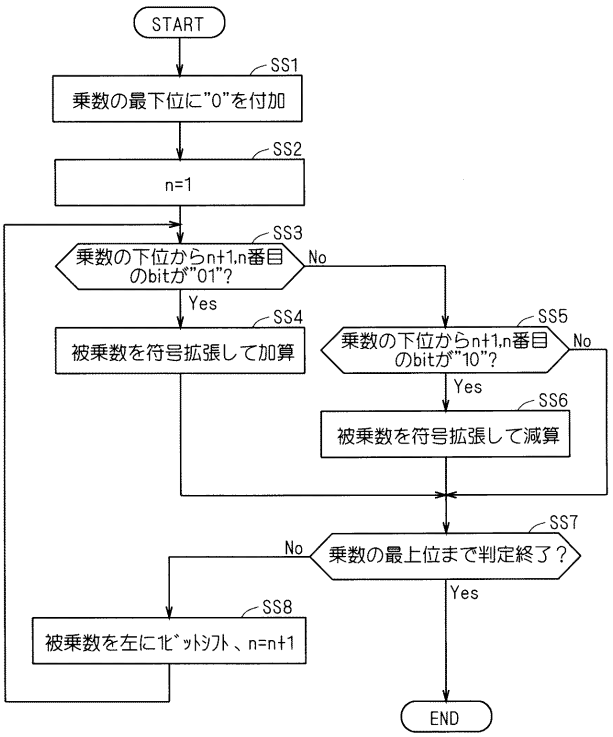
【図２４】



【図２５】



【図２６】



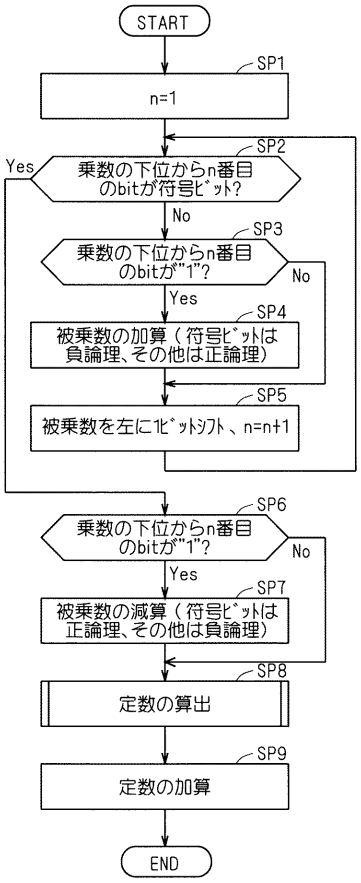
【図２７】

十進数表記	4ﾋﾞｯﾄ整数表記
-8	1000
-7	1001
-6	1010
-5	1011
-4	1100
-3	1101
-2	1110
-1	1111
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111

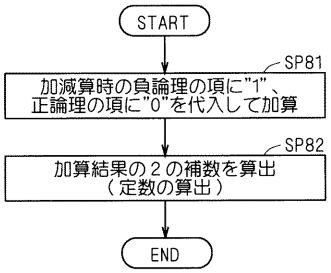
【図２８】

```
          0 1 0 1
        × 1 0 1 0
  -------------------
    1 1 1 1 0 1 1
  0 0 0 1 0 1
+ 1 1 0 1 1
  -------------------
  1 1 1 0 0 0 1 0
```

【図２９】



【図３０】



【図３１】

（ａ）

				×	0	1	0	1
					1	0	1	0
		1	1	0	1			
	0	0	1	0				
	0	1	0	1	0	1	0	

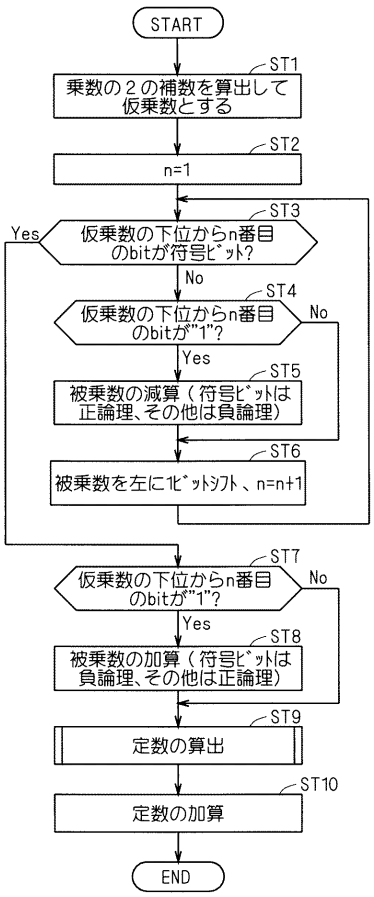
（ｂ）

				×	a3	a2	a1	a0
					b3	b2	b1	b0
			$\overline{a3b1}$	$\overline{a2b1}$	a1b1	a0b1		
+	a3b3	$\overline{a2b3}$	$\overline{a1b3}$	$\overline{a0b3}$				
			1	0	0	0		
	0	1	1	1				
	1	0	0	1	0	0	0	

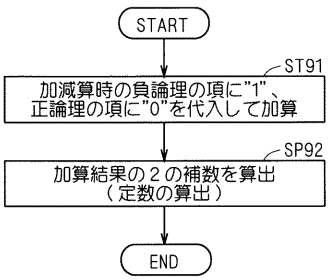
（ｃ）

		0	1	0	1	0	1	0
+	1	0	1	1	1	0	0	0
	1	1	1	0	0	0	1	0

【図３２】



【図３３】



【図３４】

（ａ）

				×	0	1	0	1
					0	1	1	0
		0	0	1	0			
	0	0	1	0				
	0	0	1	1	0	0		

（ｂ）

				×	a3	a2	a1	a0
					b3	b2	b1	b0
		$\overline{a3b3}$	$\overline{a2b3}$	$\overline{a1b3}$	$\overline{a0b3}$			
+	a3b3	$\overline{a2b3}$	$\overline{a1b3}$	$\overline{a0b3}$				
		0	1	1	1			
	0	1	1	1				
	1	0	1	0	1	0		

（ｃ）

			0	0	1	1	0	0
+	1	1	0	1	0	1	1	0
	1	1	1	0	0	0	1	0