卒業研究報告

題　　目

ＶＨＤＬによるＣＤＭＡ方式受信機の設計

指導教員

矢野　政顕　教授

報告者

山崎　慎太郎

平成 14 年　2 月　7 日

高知工科大学　電子･光システム工学科

目次

第１章　はじめに

昨今、携帯電話やコードレス電話、無線ＬＡＮなどの移動体通信の急速な普及に伴い、無線通信技術は急速な発展を遂げている。しかし、無線通信に使用できる周波数帯は限られているため、今までの無線通信方式では高速なデータ・サービスや、多くのユーザ数を確保できない。そこで、新たに登場した通信方式がスペクトル拡散技術を用いたCDMA（Code Division Multiple Access）である。本研究ではこのCDMAに着目し、CDMAの受信機をハードウェア記述言語であるVHDL（Very high speed integrated circuit Hardware Description Language）を用いて設計することを目的とする。

第１章では研究目的を述べている。第２章ではFDMA（Frequency Division Multiple Access）、TDMA(Time Division Multiple Access)、CDMA(Code Division Multiple Access)について述べる。第３章では設計にあたって必要なCDMA方式の基礎について述べる。第４章では今回設計した受信機について述べる。第５章では今回設計した受信機のシミュレーション結果、また論理合成結果を示す。第6章で、まとめとして設計した回路の評価について述べる。またVHDL記述、論理合成結果については、巻末の付録で示す。送信機側の仕様についてはシノプスデザインコンテスト用に作成されたものを引用している。［1］

第２章　多元接続方式

2.1　ＣＤＭＡの必要性

１９９６年代から急速に普及してきた携帯電話などの、移動体通信市場は爆発的な成長を遂げている。携帯電話の契約数は、社団法人電気通信事業者協会の発表によると、１９９６年１月には携帯電話の契約数は８６７万台、PHS３９５万３０００台で、２００１年１１月末の携帯電話の加入台数は６６３９万２００台、ＰＨＳは５６８万３０００台となっている。PHS の増加はそれ程大きくはないが、携帯電話はこの６年で約５８００万台増加している。これは、日本で約現在２人に１台という事になる。

携帯電話の場合も使用できる周波数帯が決められているが、それにも限りがある。携帯電話の利用者は相当な数に上るので、複数の人が同時に通話できるようにする必要がある。このようにすることを「多元接続 (Multiple Access)」と言う。多元接続の主な方法には次の３つの方法がある。すなわち、第１世代として登場した「ＦＤＭＡ」、第２世代の「ＴＤＭＡ」、そして以上の方式でも周波数帯の確保が難しくなってきたので第３世代「ＣＤＭＡ」である。

2.2　多元接続

多元接続とは、同じ地域（場所）で複数組の通信を行うとき、両者が互いに混信を起こさずにそれぞれの相手と通信できるようにすることである。[2]

2.2.1　ＦＤＭＡの概要

ＦＤＭＡ方式では、システムに与えられた周波数帯域を分割して各局に割り当てる。信号は狭帯域で伝送され、受信側での信号対雑音比を高くすることが比較的容易である。ＦＤＭＡ方式は、それほど技術的困難を伴わずに実現できるという特徴のために、これまでアナログ自動車・携帯電話システムに採用されてきた。各ユーザが使用するチャネル間にガードバンドを設けることにより、ユーザ間の相互干渉を避けている。信号は連続的であるので、送信に当たって時間的な同期を必要としない。また受信に際しても、信号の復調における高速な同期を必要としない。図 2.1 にＦＤＭＡ方式の概念図を示す。

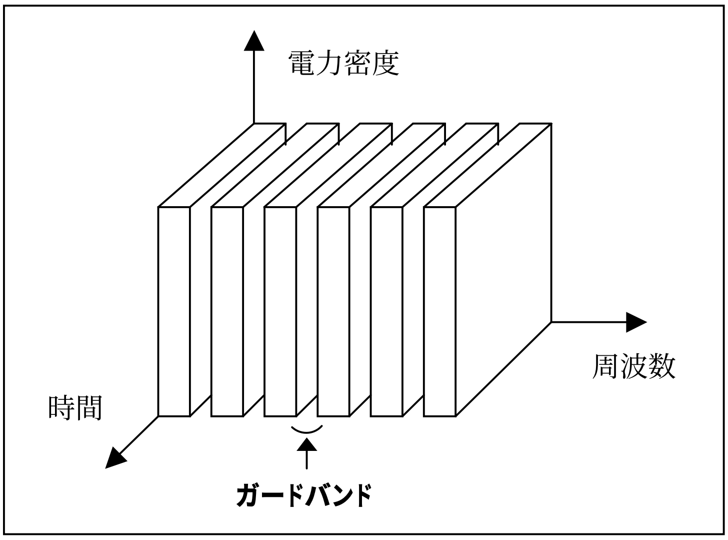


図 2.1ＦＤＭＡの概念

2.2.2　ＴＤＭＡの概要

ＴＤＭＡ方式では、時間を分割して各局に割り当てる。割り当てられた時間中はユーザ通信路の全体帯域幅を使用できる。各局は、信号を断続的かつ周期的（この周期をフレーム周期という）に送信する。また、各チャネル間には、各移動局と基地局間の距離の差や送信タイミング誤差などにより、各移動局からのバーストが重なるのを防ぐためにガードタイム が設けられる。そのため、送信タイミングの制御が必要である。従って、ＴＤＭＡは同期技術の向上してきた最近になってデジタル自動車電話や携帯電話、ＰＨＳ、衛星通信の分野で広く用いられるようになってきた。ＴＤＭＡ方式の概念図を図 2.2 に示す。

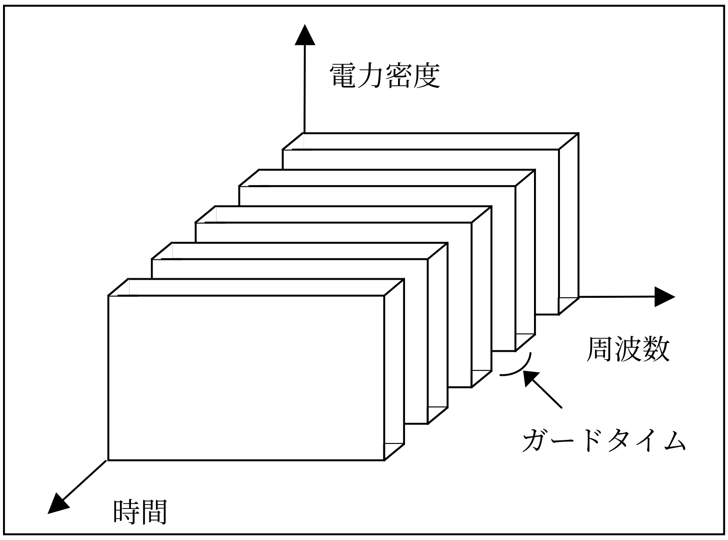


図 2.2　ＴＤＭＡの概念

また、システムに与えられた周波数帯域すべてを使ってＴＤＭＡを行うと、1フレーム中にアクセスすべき局数を多くせざるを得なくなる。このようにすると、各局から送信される断続信号（これをバーストという）の時間長が短くなり、信号対雑音比を維持するためには、各局の送信電力を高く しなければならない。そのため、消費電力の面では FDMA との差はない。

2.2.3　ＣＤＭＡの概要

ＣＤＭＡ方式では、同一の無線周波数においてユーザごとに異なる符号を割り当て、多元接続を行う。この符号を用いた方法はスペクトル拡散として古くから知られており、その秘匿性を利用して主として軍事用に利用されていたが、最近では送信電力制御の進歩により公衆用システムにも利用されるようになってきた。図 2.3 にＣＤＭＡ方式の概念図を示す。

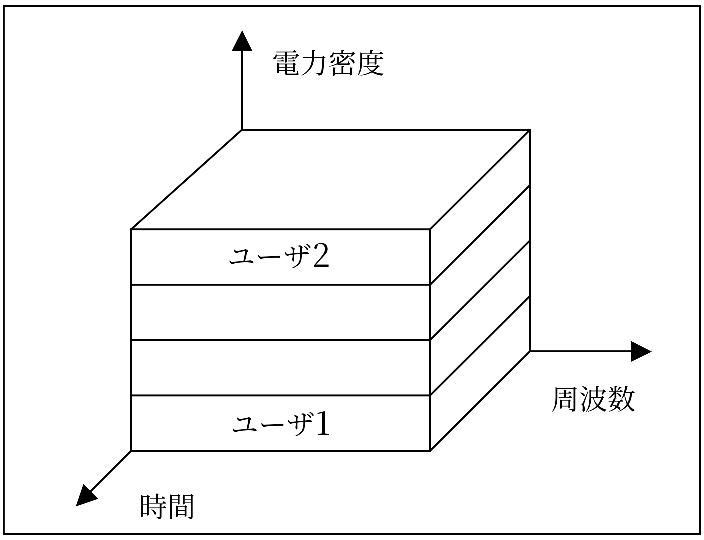


図 2.3　ＣＤＭＡの概要

ＣＤＭＡには、ＤＳ（直接拡散）方式、ＦＨ（周波数ホッピング）方式、これらを組み合わせたハイブリッド方式がある。ＤＳ-ＣＤＭＡ方式は、情報信号に１次変調を施した後に情報伝送速度よりも十分に速い拡散符号で２次変調をかけ、広帯域に拡散し、情報 1 ビットあたりの電力密度を小さくして送信するものである。受信側では、同じ拡散符号をかけて逆拡散することにより元のデータを復元する。通常、ユーザは通信のために与えられた全帯域と時間を共有

する。ユーザごとにＰＮ（擬似雑音）符号系列などの拡散符号を割り当てれば、複数のユーザが同時に接続できる。このため、伝送路において干渉成分や雑音が重畳されても、受信側で復調時に逆スペクトル拡散を行うことによって、復調された信号成分に比べて絶対レベルが十分小さくなるため、干渉や雑音の影響を受けにくいという特徴がある。

一方、ＦＨ-ＣＤＭＡは、２次変調として複数の周波数をある規則に従って順次切り換える（ホッピング）することにより、周波数相関の低い信号を得る。この方法では、ホッピングする周波数の順序がユーザごとに異なり、これを各ユーザに割り当てれば、複数のユーザが同時に接続できる。従ってＣＤＭＡでは、拡散符号を使って受信信号を逆拡散して信号を取り出すので、傍受されにくいという特徴を持つ。

しかし、多重するユーザ数が多くなってくると他ユーザからの干渉信号が大きくなるので、希望する信号を再生できなくなるという欠点を持つ。

第３章　CDMA 方式の基礎

3.1　スペクトラム拡散の概要（DS 方式）

スペクトラム拡散とは、前章の 2.2.3 節でも述べたように情報信号に１次変調を施した後、情報伝送速度よりも十分に速い拡散符号で２次変調をかけ、広帯域に拡散し、情報 1 ビットあたりの電力密度を小さくして送信方式である。CDMA 通信の概念図を図 3.1 に示す。[3]

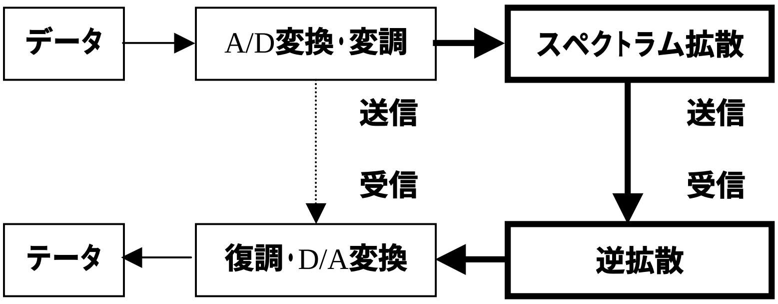


図 3.1　CDMA 通信の流れ

受信側では復調する前にスペクトラム拡散で拡散された信号を「逆拡散」を施しその送信データを受け取る（図 3.1 参照）。「拡散」「逆拡散」は PN 符号という特殊な符号を掛け合わせることにより生成される。PN 符号とは前に述べた、情報伝送速度よりも十分に速い拡散符号のことである。（図 3.2 参照）

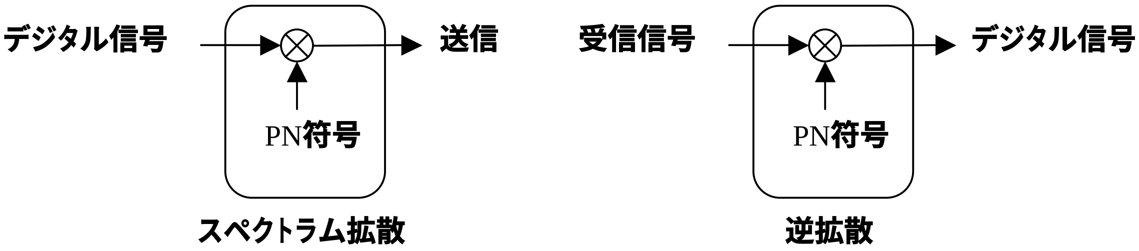


図 3.2　PM 符号による拡散・逆拡散の概念

3.2　PN 符号

ＰＮ符号（Pseudorandom Noise：擬似ランダム雑音）は、ランダム雑音を模擬するビット列で表す。「擬似ランダム」であるので真のランダムではなく、あらかじめ決められたビット列を繰り返します。理想的な PN 符号は、まったく無秩序に真の乱数として取り出されるものが望ましい。例えば熱雑音を利用して PN 符号を作れるが､熱雑音は再現性がないため送信側と受信側で同じ PN 符号を作り出すことは不可能である。

送信側と受信側で同じ PN 符号が同時に作るためには、次々に発生される系列がある状態変数に応じて決まる関数で表せる必要がある。このような PN 符号発生器として M 系列（maximum length code）を用いた発生器がある。次節 M 系列で述べる。

3.3　M 系列

M 系列はシフトレジスタと加算器（XOR）で簡単に作れる（図 3.3 参照)。ただし、図でシフトレジスタの中間部分から抜き出されているタップ（帰還タップ）はどの位置から取り出してもよいものではなく、特定の少数の組み合わせを使った場合にのみ取り出されるデータ系列を M 系列と呼ぶ。

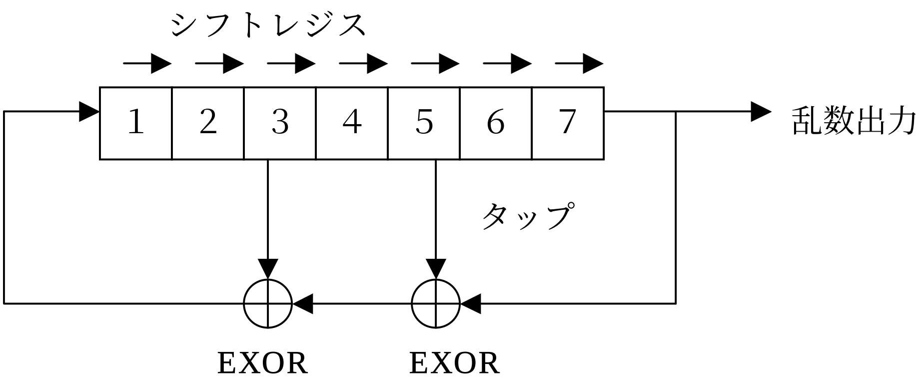


図 3.3　７段の M 系列発生器

図 3.3 のシフトレジスタは７ビットのデータを記憶するものであり、最大 2^7 ＝１２８通りの状態が記憶できる。また１回シフトすると１２８通りのどれかに移行する。初期値がどんな値であっても最大１２８回シフトすればレジスタの内容は初期状態に戻るのである。しかし、初期値がオールゼロの場合、初段

に帰還される値が０になって以降レジスタの内容が０を繰り返す。そのため、レジスタの初期値をオールゼロの値を除くと１２７が実現可能な最大周期となる。

M 系列は、以下のような特徴をもっている。

- どんな周期の M 系列でも１周期中に含まれる「０」と「１」の数の割合が一定で、段のシフトレジスタを使ったとすれば、１の数が2^{k-1}個、０の数が（2^{k-1}）－１になる。これはレジスタの状態がオールゼロであった場合を除くからである。０と１のバランスがとれているので、０を－１に置き換える PN 系列（PN 符号の並び）では、直流成分が少なくなって好都合になる。詳しくは次節の相関値で述べる。

- 系列中に０または１が連続して現れるとき､その連続の長さを run と呼ぶ。系列中１周期中、長さｍの run の発生頻度は長さm+1 の run の頻度のちょうど２倍になっている。これは真の乱数がもつ性質と同じである。

- 発生された M 系列と、レジスタの初期値だけが違う同 M 系列のデータを加算すると、もとの M 系列を時間的にずらしただけの同一の系列ができる（シフト加法性）ここでの加算とは桁上げのない特別な加算で XOR（排他的論理和）と同じと考えればよい。

3.4　拡散・逆拡散の例

ここで４段の M 系列発生器を使い、送信する目的データの拡散・逆拡散をおこなう。PN 符号が送信側と受信側で同じ場合を図 3.4（回路図)、図 3.5（動作波形)、図 3.6（シミュレーション仕様）を示す。次に PN 符号が送信側と受信側で異なる場合を図 3.7（回路図)、図 3.8（動作波形)、図 3.9（シミュレーション仕様）を示す。また、図 3.12 にＰＮ符号の時間波形の例を示す。

動作波形の結果、PN Code が送信側と受信側で一致しないと復調できないとが確認できる。また、PN Code が同じでも周期が送信側と受信側で合わなければ復調できない（図 3.10，3.11 参照)。これは次節「相関関数」で説明する。

ここでの回路図作成とシミュレーションには ALTERA 社の Max+plusⅡを使用した。

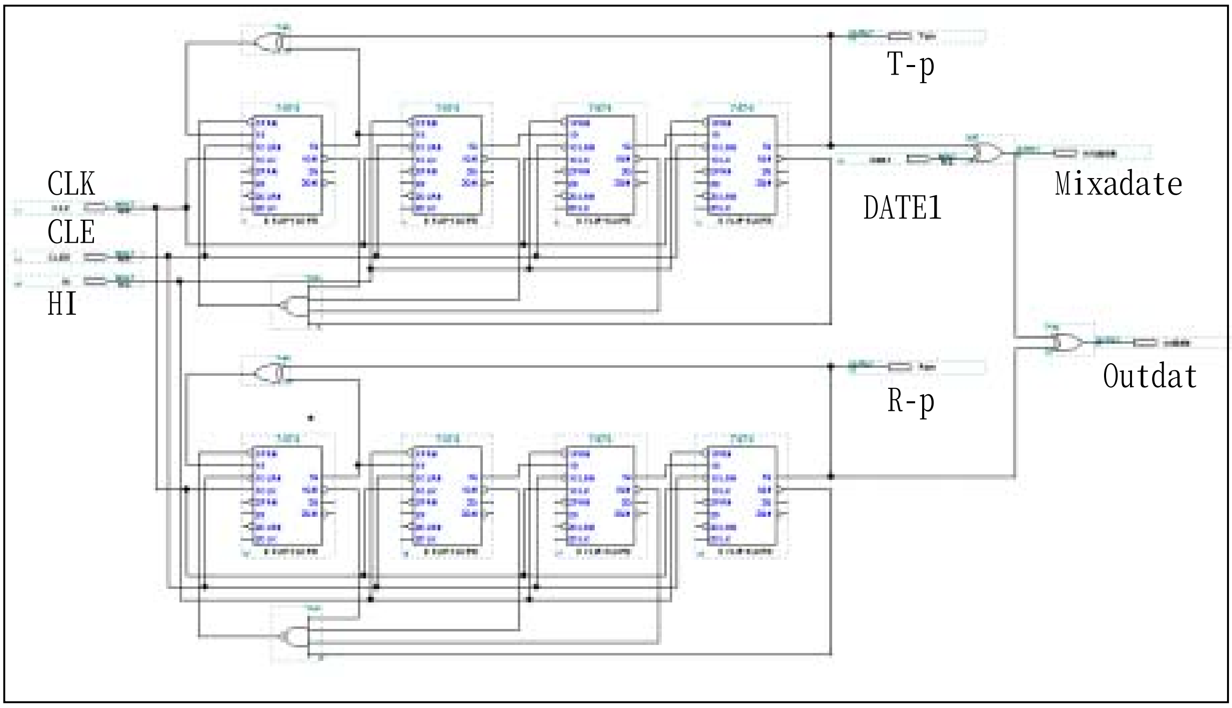


図 3.4　4bit M 系列回路図（送受信機とも 1,4 段目タップ）

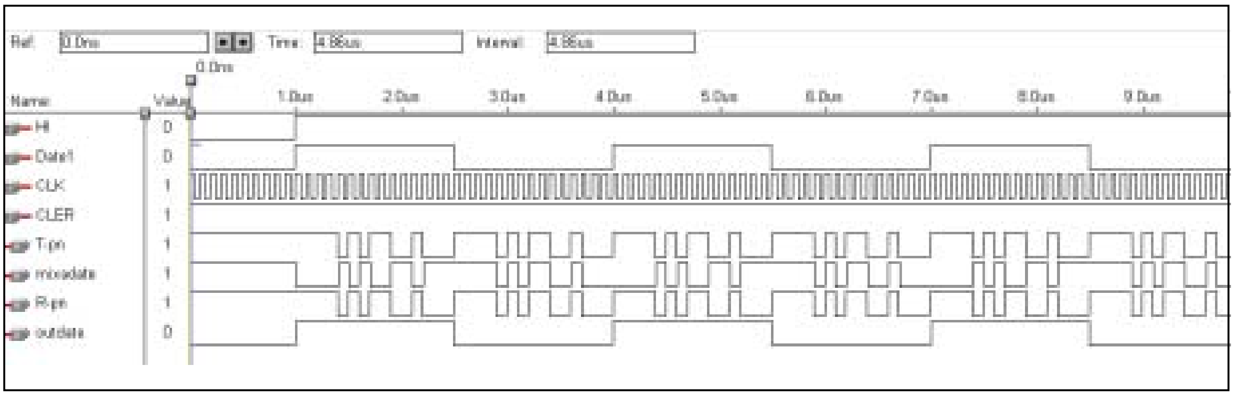


図 3.5　拡散・逆拡散波形

HI　　　：リセット（1us）

Date1　　　：送信データ　　（ビットレート 1.5us）

CLK　　：クロック　100ns

T-pn　　：送信機　pn 符号　（チップレート 100ns）

111101011001000

R-pn　　：受信機　pn 符号

111101011001000

図 3.6　シミュレーション仕様

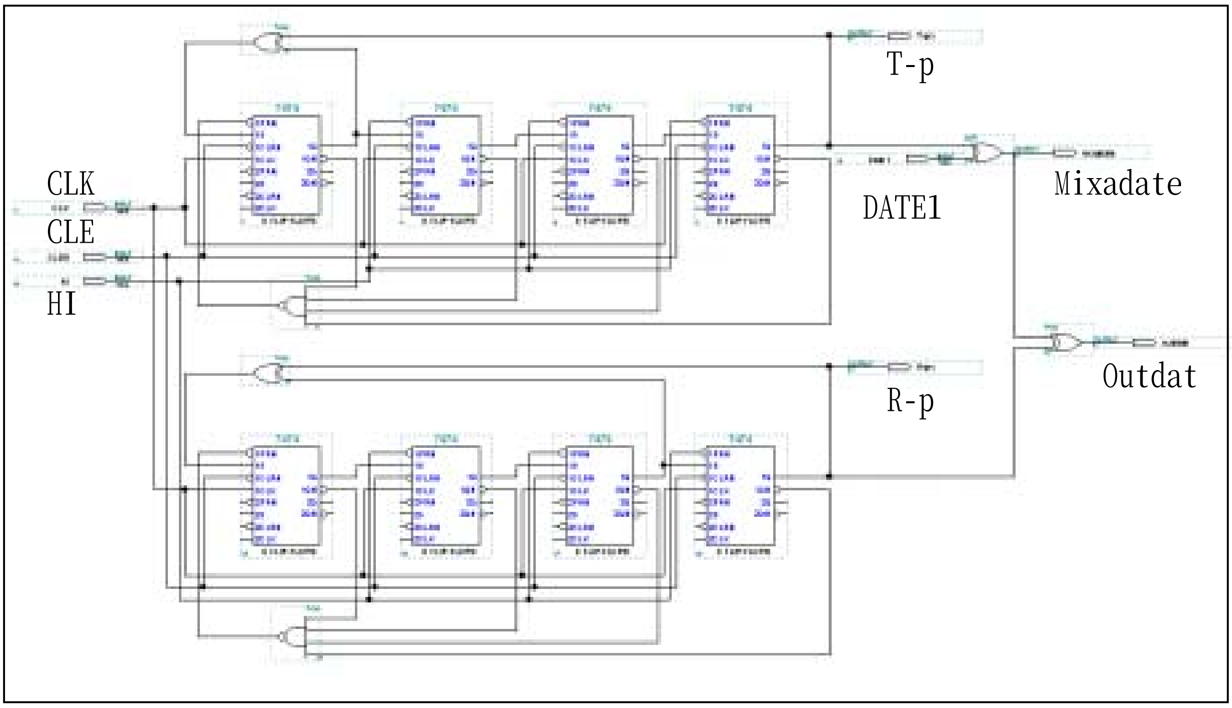


図 3.7　4bit M 系列回路図（受信機 3,4 段目タップ）

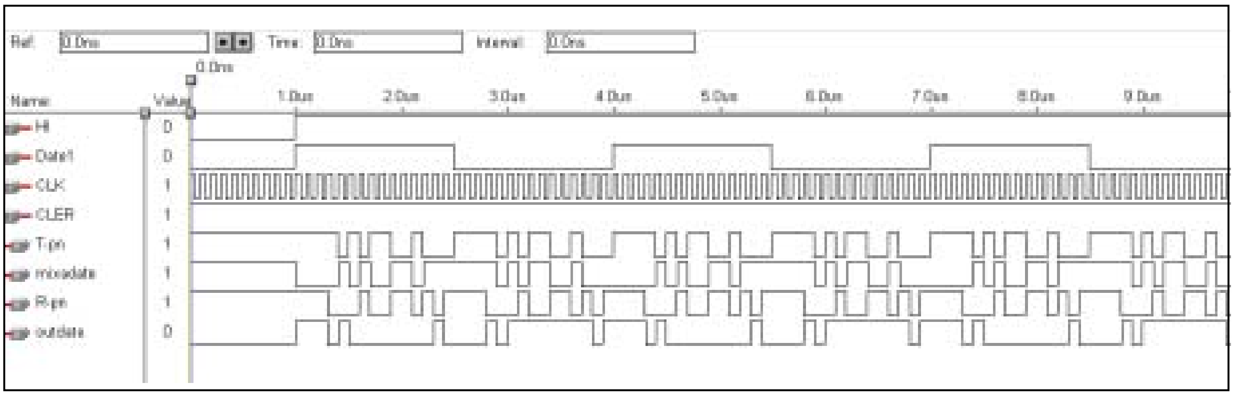


図 3.8　拡散・逆拡散波形

HI　：リセット（1us）

Date1　：送信データ　（ビットレート 1.5us）

CLK　：クロック　100ns

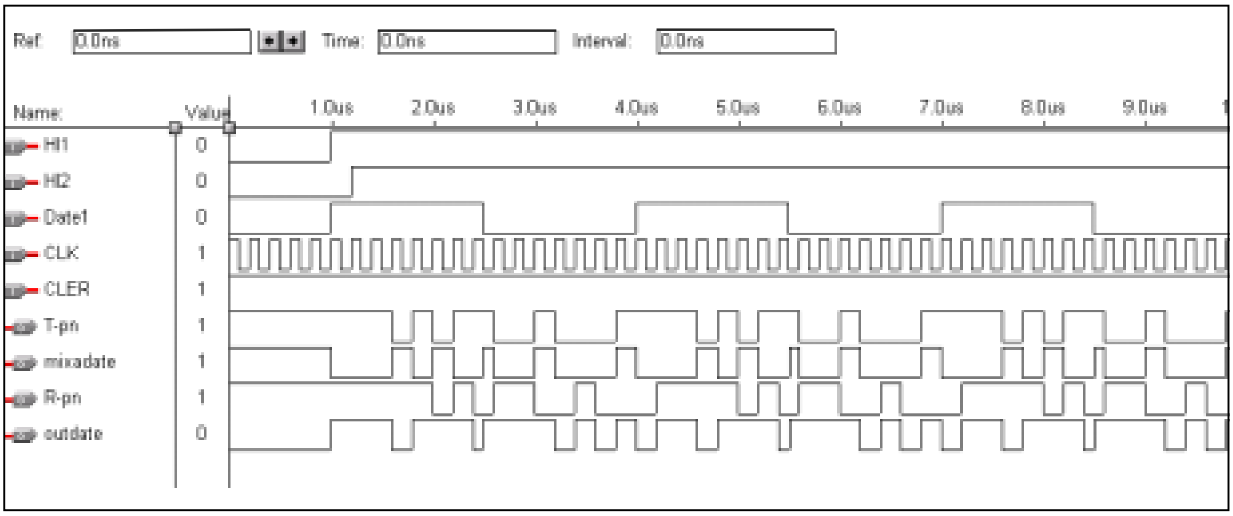
T-pn　：送信機　pn 符号　（チップレート 100ns）

111101011001000

Mixadate :送信データ（拡散後）

R-pn　：受信機　pn 符号

図 3.9　シミュレーション仕様



Ref: 0.0ns Time: 0.0ns Interval: 0.0ns

図 3.10　拡散・逆拡散波形

HI1　　：リセット（1us）
HI2　　：リセット（1.1us）
Date1　　：送信データ　　（ビットレート 1.5us）
CLK　：クロック　100ns
T-pn　：送信機　pn 符号　（チップレート 100ns）
　　111101011001000
Mixadate :送信データ（拡散後）
R-pn　　:受信機　pn 符号

図 3.11　シミュレーション仕様

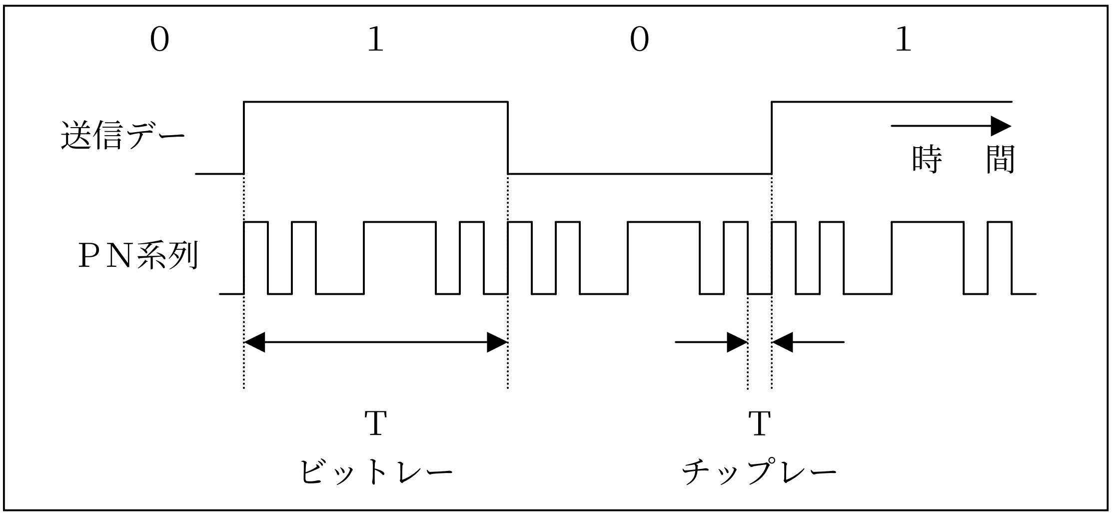


図 3.12　ＰＮ符号の時間波形

3.5　相関関数

スペクトラム拡散を理解する上で重要な鍵となるのが相関値である。相関知は「関数波形の類似性」をあらわす値である。厳密には「２つの関数波形の、瞬間値の現れ方の類似性」ともいう。相関値を求めるには、「対象の２つの関数を掛け合わせて」から「一定の区間について積分する」だけでよい。２つの関数ａ（ｔ），ｂ（ｔ）について相関関数（絶対値が１以内に収まるようにした相関値）を計算した場合

- １ならばａ（ｔ）＝ｂ（ｔ）であり
- －１ならばａ（ｔ）＝－ｂ（ｔ）となる
- ０の時は「波形の類似性」がない

このことを指す。互いの相関値が０である符号で拡散された信号は、一度混ざり合っても逆拡散時に完全に分離できる。相関値・相関係数は通常信号の波形について積分して求めるのだが、「ビット間の同期がとれたデジタル信号波形」に限り、「一致しているビット数と不一致ビット数」から簡単に相関値・相関係数が求めることができる。

3.4 節で述べた拡散・逆拡散の例（図 3.5 参照）を用いて時間シフト量によるＰＮ符号の変化を表 3.1 に示す。時間シフト量０と１のＰＮ符号（４段のM系列・1、4 段目タップ）で相関係数を見た場合、

相関係数＝（一致したビット数－不一致ビット数）／ビット長

で求められるので、それぞれ代入した場合

相関係数＝（７－８）／１５＝－０．０６６６…

となり、限りなく０に近いため波形は類似していないと言える。これは図 3.10 でも確認できる。このことから２つの波形の時間的ずれがないときは同一の波形なので相関値は１である。式であらわすと自己相関関数は、

$$R(\tau) = 1,\ \tau = 0,\ 2^{k}-1,\ 2\,(^{2k-1}),\ 3\,(^{2k-1}),\ \cdots$$

$$= \frac{-1}{2^{k}-1},\ \tau\text{が上記以外のとき}$$

になる。時間的ずれ量を離散的ではなく連続的にとると、図 3.12 のグラフが描ける。ここではＴｃは発生させたＰＮ系列１チップの時間幅である。

表 3.1　ＰＮ符号時間シフト表

時間シフト量	PN符号															元のコードと一致する数	不一致の数
0	1	1	1	1	0	1	0	1	1	0	0	1	0	0	0	15	0
1	1	1	1	0	1	0	1	1	0	0	1	0	0	0	1	7	8
2	1	1	0	1	0	1	1	0	0	1	0	0	0	1	1	7	8
3	1	0	1	0	1	1	0	0	1	0	0	0	1	1	1	7	8
4	0	1	0	1	1	0	0	1	0	0	0	1	1	1	1	7	8
5	1	0	1	1	0	0	1	0	0	0	1	1	1	1	0	7	8
6	0	1	1	0	0	1	0	0	0	1	1	1	1	0	1	7	8
7	1	1	0	0	1	0	0	0	1	1	1	1	0	1	0	7	8
8	1	0	0	1	0	0	0	1	1	1	1	0	1	0	1	7	8
9	0	0	1	0	0	0	1	1	1	1	0	1	0	1	1	7	8
10	0	1	0	0	0	1	1	1	1	0	1	0	1	1	0	7	8
11	1	0	0	0	1	1	1	1	0	1	0	1	1	0	0	7	8
12	0	0	0	1	1	1	1	0	1	0	1	1	0	0	1	7	8
13	0	0	1	1	1	1	0	1	0	1	1	0	0	1	0	7	8
14	0	1	1	1	1	0	1	0	1	1	0	0	1	0	0	7	8
15	1	1	1	1	0	1	0	1	1	0	0	1	0	0	0	15	0

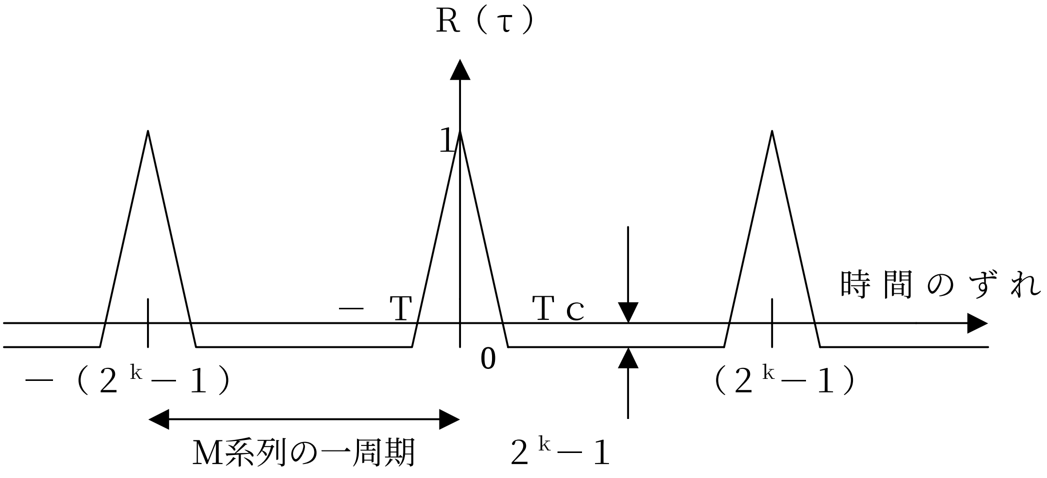


図 3.13　M系列の自己相関関数

3.6　具体的なCDMA通信の例

まず、CDMA通信の具体的な例として、系列長７のPN符号３種類を用いてデータを送信する場合について説明する。

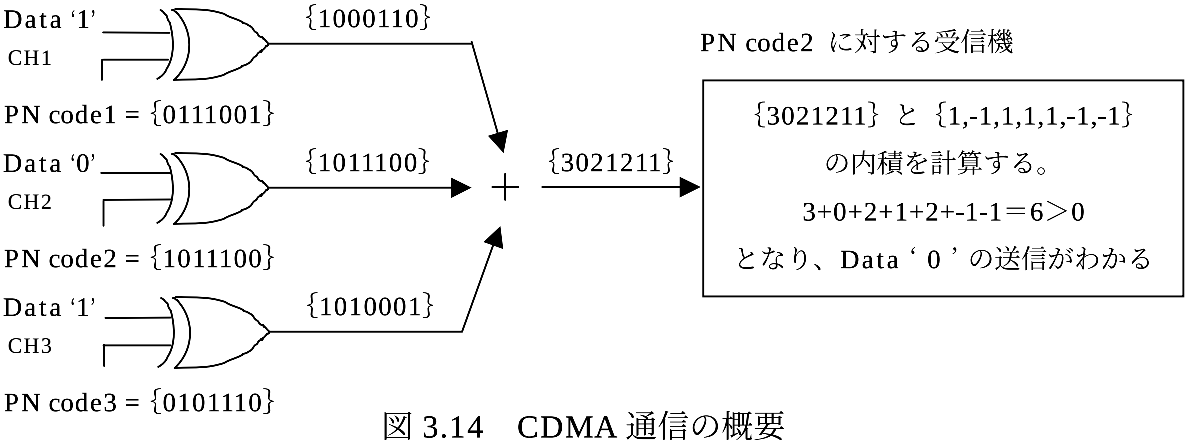


図3.14　CDMA通信の概要

図3.14に示すように、PN code１= ｛０１１１００１｝、PN code２＝｛１０１１１００｝、PN code３＝｛０１０１１１０｝なる３種類のPN codeを用いて、それぞれのPN Codeに対してCH1のData’１’、CH2のData’０’、CH3のData’１’を送信する。各PN codeはそれぞれのＣＨのDateとXOR（排他的論理和）され、CHのDateが’１’の時には各PN codeとは反転し、CHのDateが’０’では反転しない。これをデータの拡散という。次に各PN codeで拡散したデータを加算して２bitの情報とする。加算結果を｛３０２１１２１１｝の系列で表しているが、実際は｛１１、００、１０、０１、０１、１０、０１、０１｝の信号として送信される。

受信機では｛３０２１１２１１｝となる系列が受信され、PN code２に対してのDataを受け取る場合は、受信した系列とPN code２の相関をとる。具体的にはPN codeの中の’０’を’－１’と読み直して、系列をベクトルと考えて内積をとることになる。この結果が正の大きめの値であれば送信データは’０’であり、負の大き目の値であれば送信データは’１’と判断する。

図3.15では各DATAの拡散の波形を示す。図3.16では加算・逆拡散・復調を示す。図3.14では１つのデータの拡散しか表していないが、状態の変化がわかるようにここでは続けてPN code 2 に対してDATA’０’の信号を送信している。ここでは各PN符号は初期の状態を繰り返すものとする。

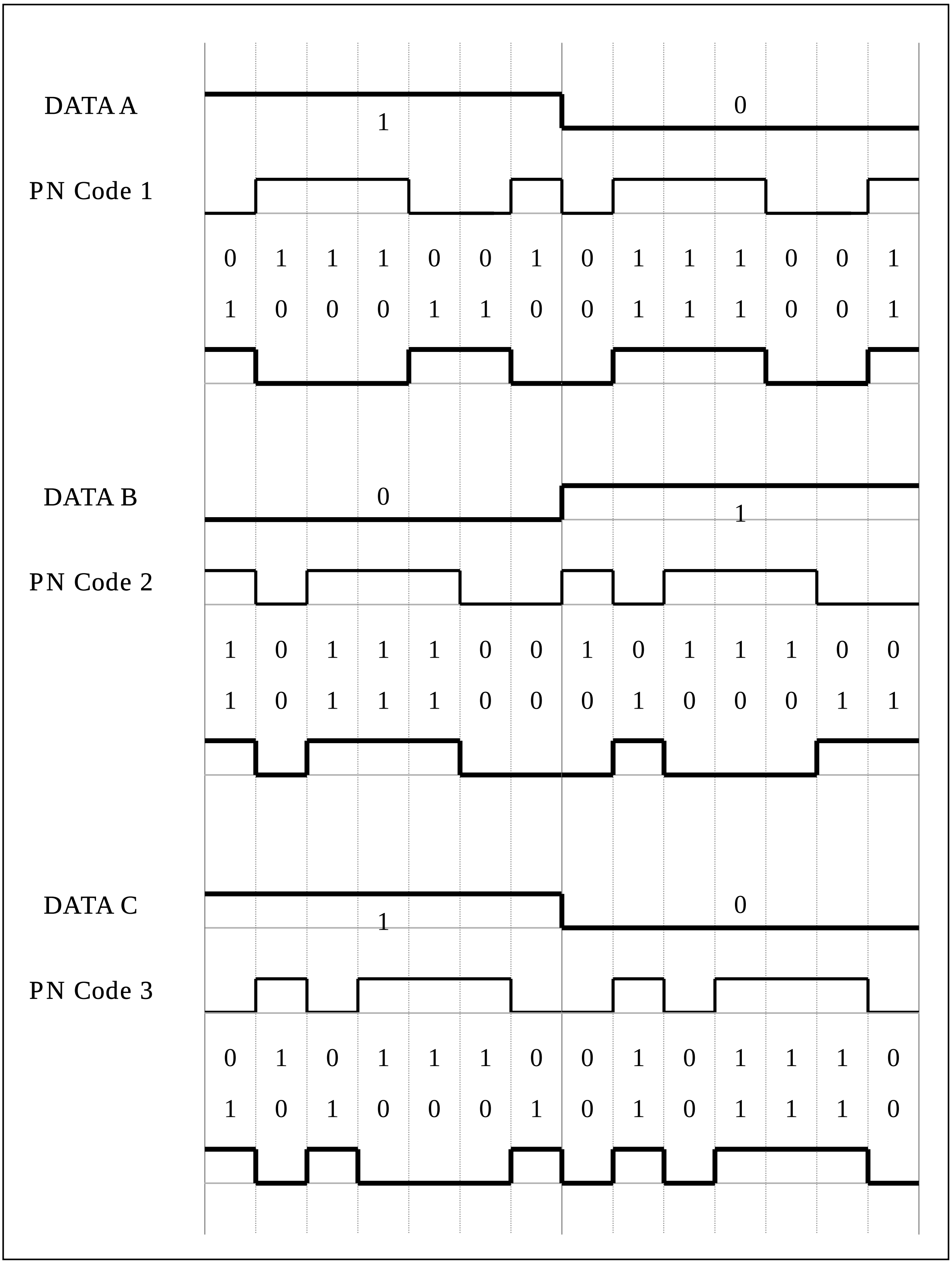

DATA A
1
0
PN Code 1
0 1 1 1 0 0 1 0 1 1 1 0 0 1
1 0 0 0 1 1 0 0 1 1 1 0 0 1
DATA B
0
1
PN Code 2
1 0 1 1 1 0 0 1 0 1 1 1 0 0
1 0 1 1 1 0 0 0 1 0 0 0 1 1
DATA C
1
0
PN Code 3
0 1 0 1 1 1 0 0 1 0 1 1 1 0
1 0 1 0 0 0 1 0 1 0 1 1 1 0

図 3.15　拡散波形

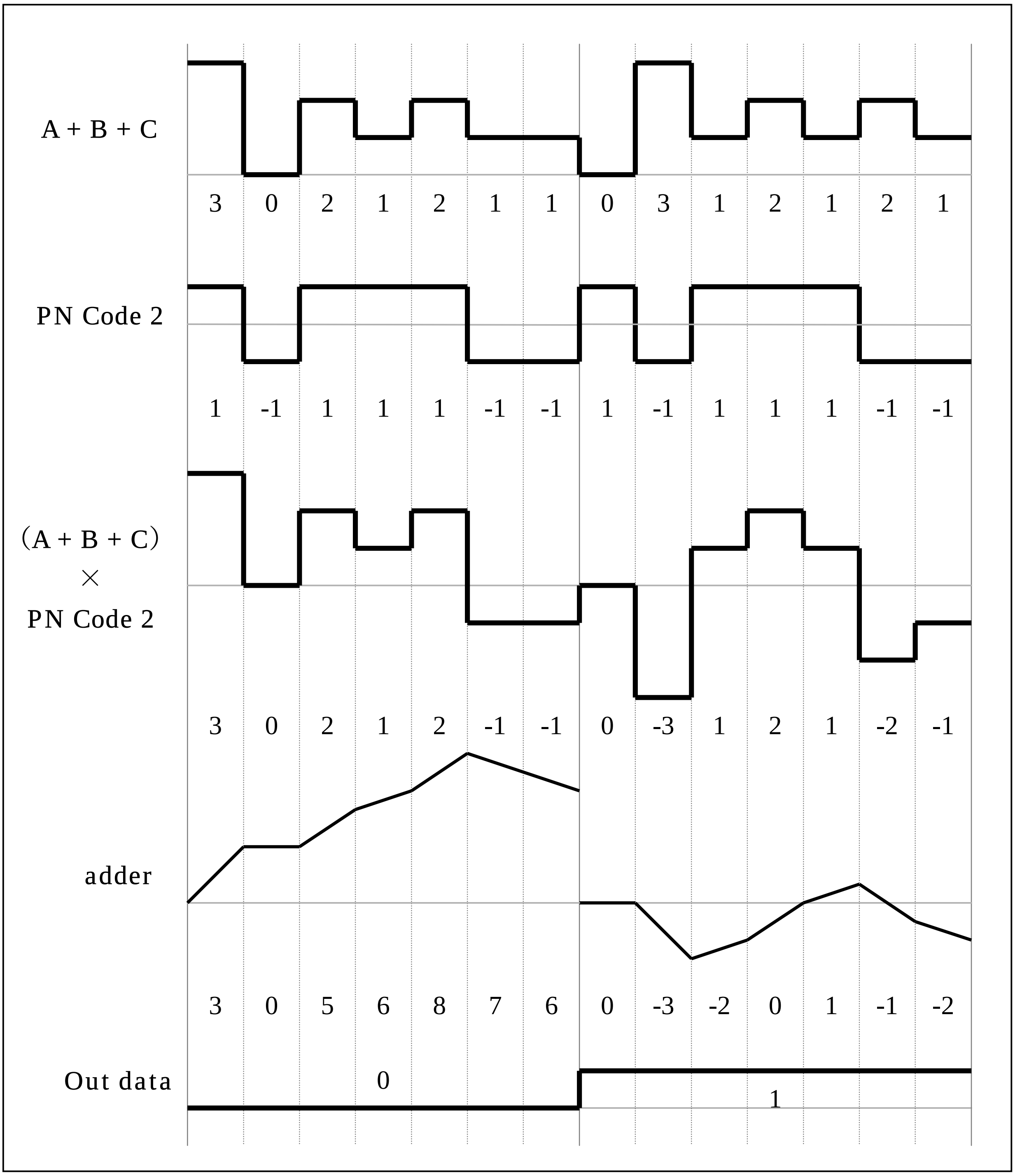


図 3.16　加算・逆拡散・復調波形

第４章　VHDLによるCDMA受信機の設計

この章では全体のシステム構成と実際に設計した受信機について述べる。送信機については、「Design wave 2001 デザインコンテスト」で使用された仕様を使用した。[1]

設計にあたって、ハードウェア記述言語のVHDLはXILINX社のweb pack「Modelsim」、ALTERA社の「max+plusⅡ」を使用し、シミュレーションを行った。送信機、受信機のVHDL記述は付録として示している。

4.1　システム構成

CDMA通信のシステム全体の構成を図4.1に示す。大きく２つのユニットに分かれ、送信機（transmitter）と受信機（receiver）から成る。送信機の仕様は次節4.2で述べ、受信機の仕様は4.3節で述べる。

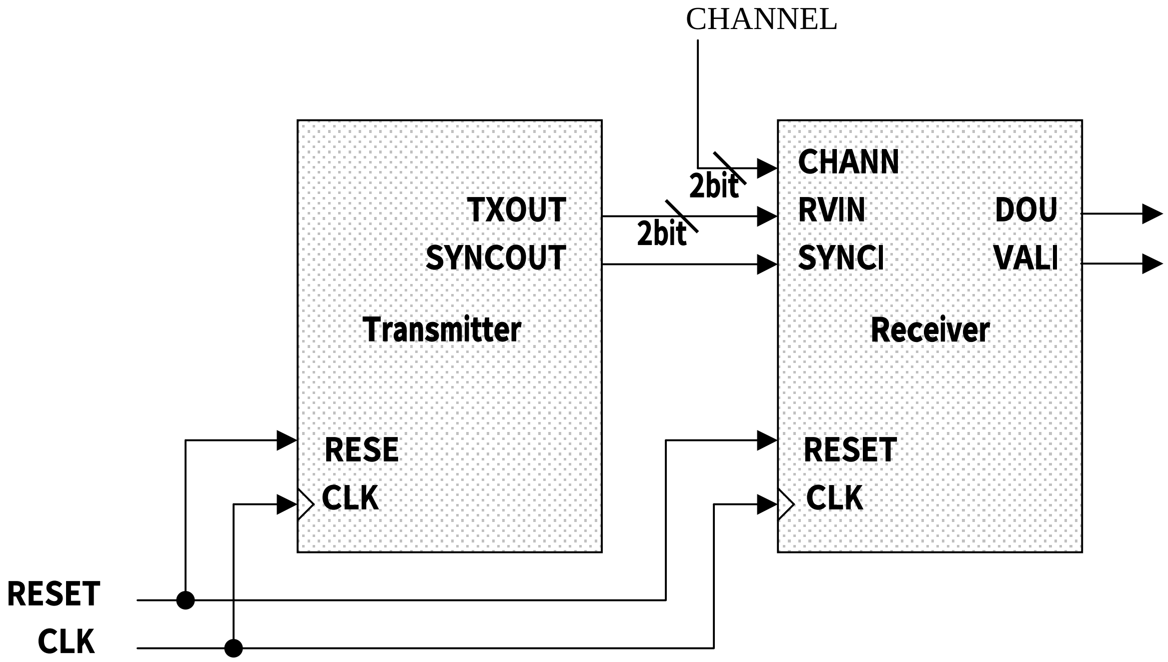


図4.1　システム構成

4.2　送信機

この節では送信機（Transmitter）の仕様と動作を述べる。ここで用いられる送信機は前章？節で述べた、CDMA 通信例の送信側と基本的に動作は同じである。例では系列長 7 の PN Code であったが、今回使用する送信機は系列長 127 の PN Code である。また受信側で PN Code の開始がわかるように同期信号を付加している。

表 4.1　送信機の仕様

Transmitter			
信号名	入出力	ビット幅	説明
CLK	IN	1	クロック信号
RESET	IN	1	‘1’でリセット
TXOUT	OUT	2	送信信号
SYNCOUT	OUT	1	同期信号

表 4.1 に送信機の仕様を示す。送信機は大きく分けると 4 つのユニットから成る。系列長１２７で 3 種類の PN 符号を生成するユニット。各チャンネル（CH１/CH2/CH3）に送信するデータを生成するデータ生成ユニット。送信データを拡散するユニット。送信データを加算するユニットに分けられる。送信機の構成を図 4.2 に示す。

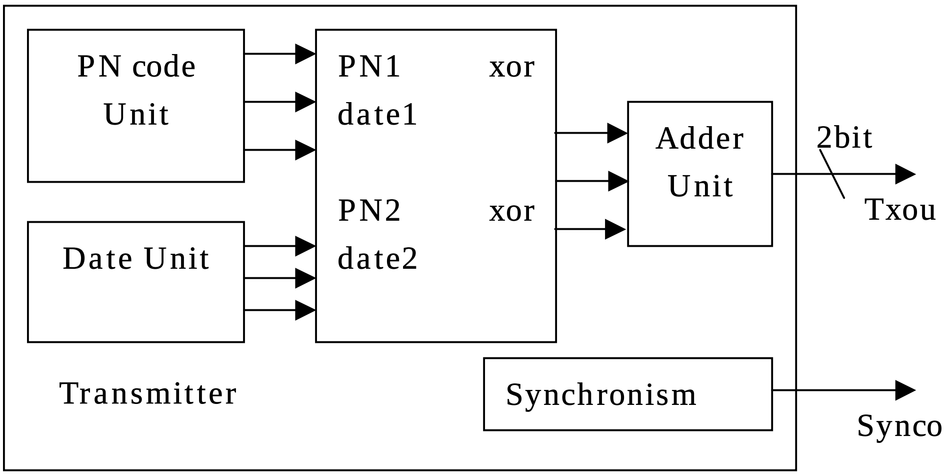


図 4.2　送信機の構成

4.2.1　PN 符号発生ユニット（送信機）

まず、ＰＮ符号を発生するユニットを説明する。前章で「相関関数」で説明した通り、相関値が０近い場合、拡散された信号（ここでは生成されるＰＮ系列のこと）は、波形が類似していないので一度混ざり合っても完全に分離できると説明した。ＰＮ符号は他のＰＮ符号と無相関でなくてはならない。系列長１２７のＰＮcode を７つの D－FF を使って無相関な PN 符号を生成する（図 4.3 参照）。

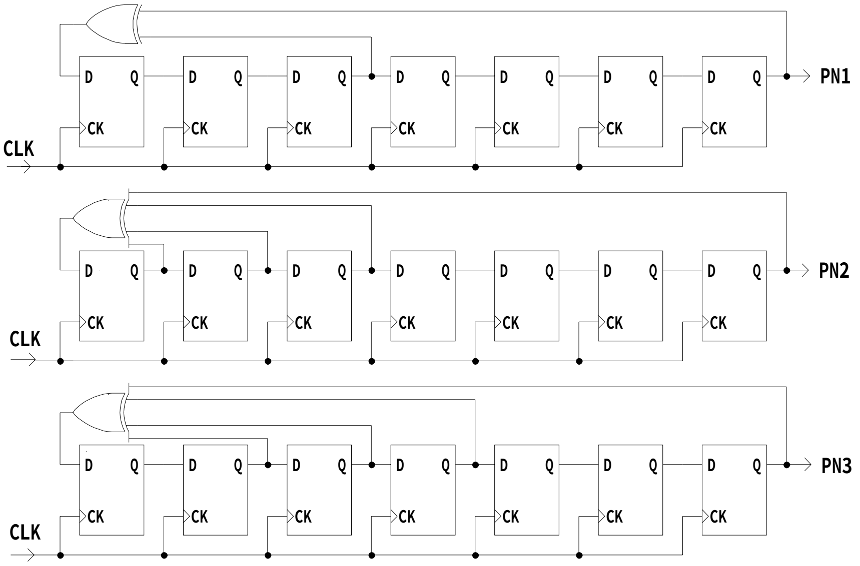


図 4.3　PN 符号発生器

前章 3.3 節の M 系列で述べたように、各 D－FF（D フリップフロップ）の帰還タップの位置を変えることによって互いに無相関な PN 符号を生成している。また、この PN 符号生成器で実際に生成した PN 符号チャンネルごとに示す（図 4.4 参照）。ただし、D－FF の初期値は ALL=1 としているため最初の出力の７つは１が出力されている。

PN1＝1111111 0001110 1100010 1001011 1110101 0100001 0110111 1001110 0101011 0011000 0011011 0101110 1000110 0100010 0000010 0100110 1001111 0111000 0

PN2＝1111111 0111011 0111101 0001011 0010111 1100010 0000011 00110110 0011100 1110101 1100001 0011000 0010101 0110100 1001010 0111100 1000110 101000 0

PN3＝1111111 0010000 0010110 0000111 0100001 0011100 0110100 1001011 1011011 1001101 1001010 1101011 1110111 1000011 0001000 1010011 0011110 1010100 0

図 4.4　系列長１２７の PN 符号

4.2.2　送信データ生成ユニット

次にデータ生成ユニットについて説明する。送信データは以下の信号を繰り返し送信する。

CHANNEL ‘０１’ 「DATA1 ＝ ０１０１０１０１」
CHANNEL ‘１０’ 「DATA2 ＝ ００１１００１１」
CHANNEL ‘１１’ 「DATA3 ＝ ００００１１１１」

4.2.3　拡散・加算ユニット

最後に拡散ユニットと加算ユニットについて説明する。拡散については前章の 3.6 節 CDMA 通信例で述べたが､各 PN code はそれぞれの送信データと XOR（排他的論理和）され、Data が’１’の時には各 PN code とは反転し、Data が ’０’では反転しない状態で送信される｡加算ユニットも前章の 3.6 節 CDMA 通信例で述べた通りである。

4.2.4　シミュレーション（送信機）

今回使用した送信機のシミュレーション結果を図4.5 に示す｡図4.6 に RESET 解除後の波形を拡大して示し、また各 PN Code 発生器の１クロックごとの状態を示す。

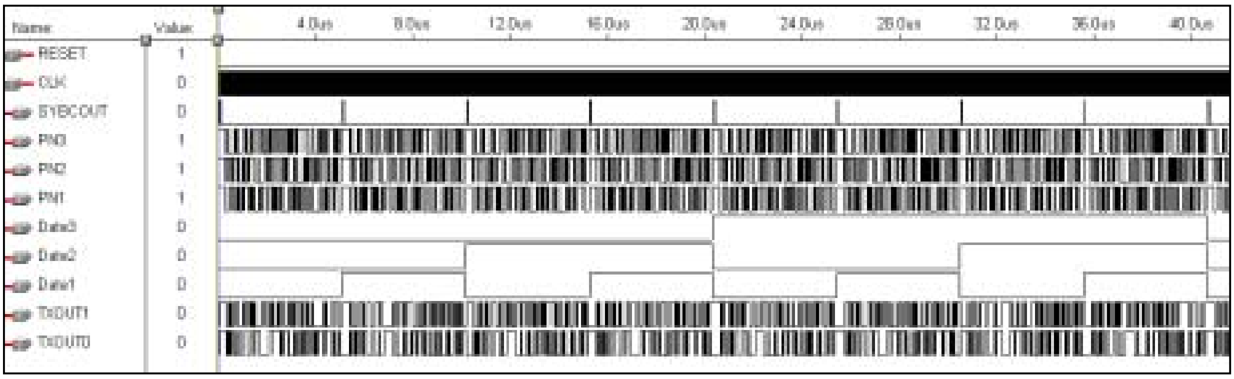


図 4.5　送信機動作波形

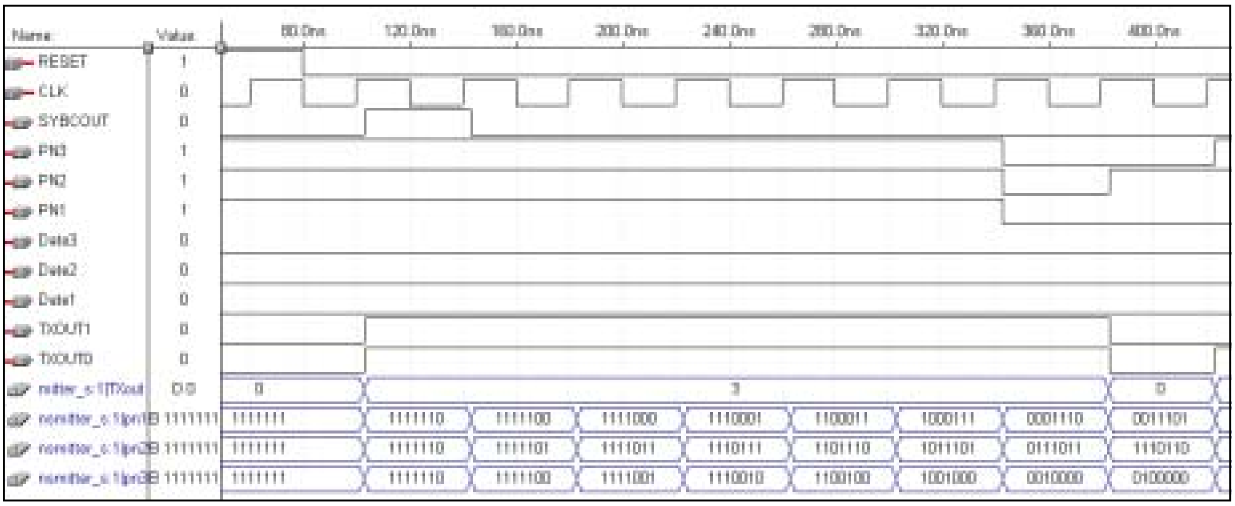


図 4.6　拡大波形（RESET 解除後）

4.3　受信機

この節では今回設計した CDMA 受信機（Receiver）の仕様と動作を述べる。以下の表 4.2 に受信機の仕様を示す。

表 4.2　受信機の仕様

Receiver			
信号名	入出力	ビット幅	説明
CLK	IN	1	クロック信号
RESET	IN	1	‘1’でリセット
SYNCIN	IN	1	同期信号
RVIN	IN	2	送信機からの出力
CHANNEL	IN	2	チャンネル選択
DOUT	OUT	1	復調されたデータ
VALID	OUT	1	値が確定した時に’1’を出力

受信機の動作は前章 3.6 節 CDMA 通信例の受信側と基本的には同じであるが、PN 符号開始のタイミングに送信機から同期信号を使用する。これは、前章の 3.5 節「相関関数」で説明したように、M 系列の特徴として周期が合わなければ相関値が小さくなり、望ましい結果が得られない。そこで同期信号で M 系列の周期を合わせて送信機側からの信号との相関を取ることで、自己相関値に近い値が得られる。

受信機は大きく分けると３つのユニットから成る。CHANNEL（チャンネル）の選択により必要な PN Code を生成し、CLK（クロック）ごとに PN Code を出力する PN Code ユニット，PN Code ユニットで生成された PN Code と送信機からの送信データと内積をとり出力する Addition ユニット，および Addition ユニットで求められた内積が閾値を越えた場合、０または１の判断して復調するユニットである。受信機の構成を図 4.7 に示す。

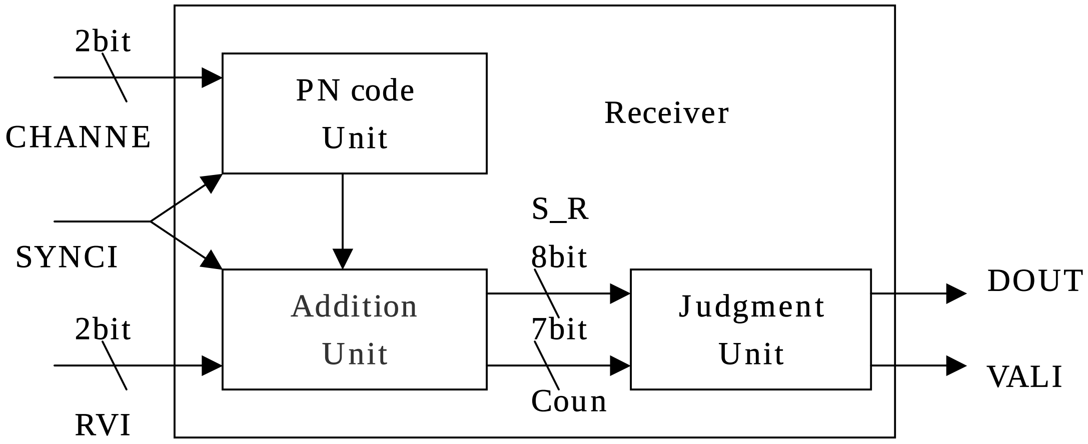


図 4.7　受信機の構成

4.3.1　PN Code 発生器

この節では、PN Code ユニットについて説明する。このユニットでは送信機側と同じ PN 符号発生器を使用する（図４.3 参照）。CHANNEL を選択することで目的の PN Code を生成する。また、SYNCIN（同期信号）により PN Code の開始する。図 4.8 に CHANNEL の選択が ’０１’ の場合の動作波形を示す。

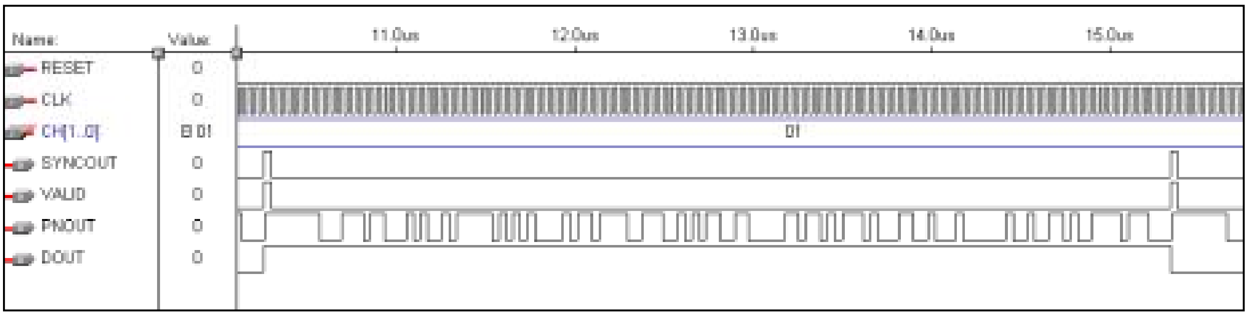


図 4.8　ＰＮCode 動作波形

4.3.2　Addition ユニット（逆拡散）

この節では、逆拡散にあたる Addition ユニットについて説明する。このユニットは PN code ユニットにより生成された信号と送信側からの送信データ RVIN との内積をクロックの立ち上がりごとに取る。この作業により逆拡散の動作を行っている。このユニットの構成を図 4.9 に示す。

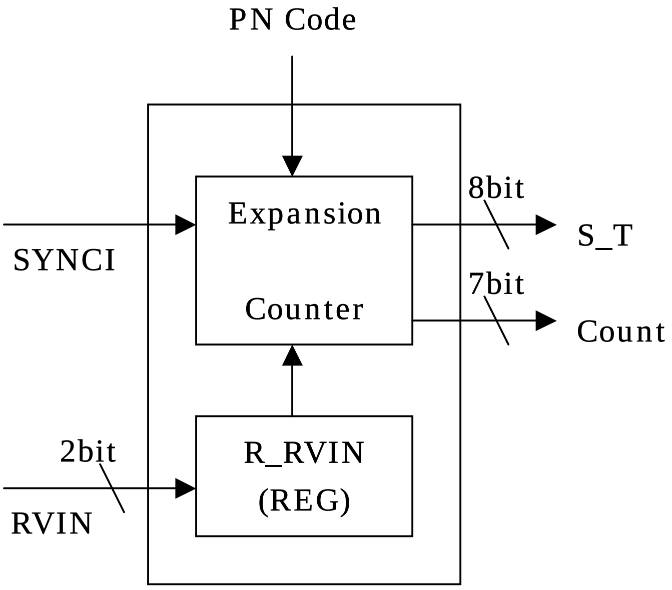


図 4.9　Addition 構成

PN Code は SYNCIN に同期して生成開始されるため Addition ユニットに送られてくるのが、１CLK 遅れる。送信データとの内積のタイミングを合わせるため一旦レジスタ（R_RVIN）に格納する。

Expansion では系列長１２７PN Code の分、内積の計算をおこなう。計算の動作は前章の 3.6 節 CDMA 通信例の受信機の計算と同じである。しかし、ここでは１２７回加減算をおこなうため送信されてきたデータを 8bit に拡張する。これは送信データの値は 0,1,2,3 の 4 種類であり、その平均は 1.5 ある。前章 3.5 節 M 系列の特徴で述べたように PN Code ‘1’ ,’0’ の発生の確率は約 1/2 なので系列長 128 で考えればプラスの数は 64 でありデータの平均値を掛け合わせると 80 になる。80 を 2 進で表すには 7bit 必要である。今回、2 の補数でマイナスを表現するため符号ビットとして MSB に 1bit 付加し 8bit としている。図 4.10 に Expansion のフローチャートを示し、図 4.11 に Addition の動作波形を示す。

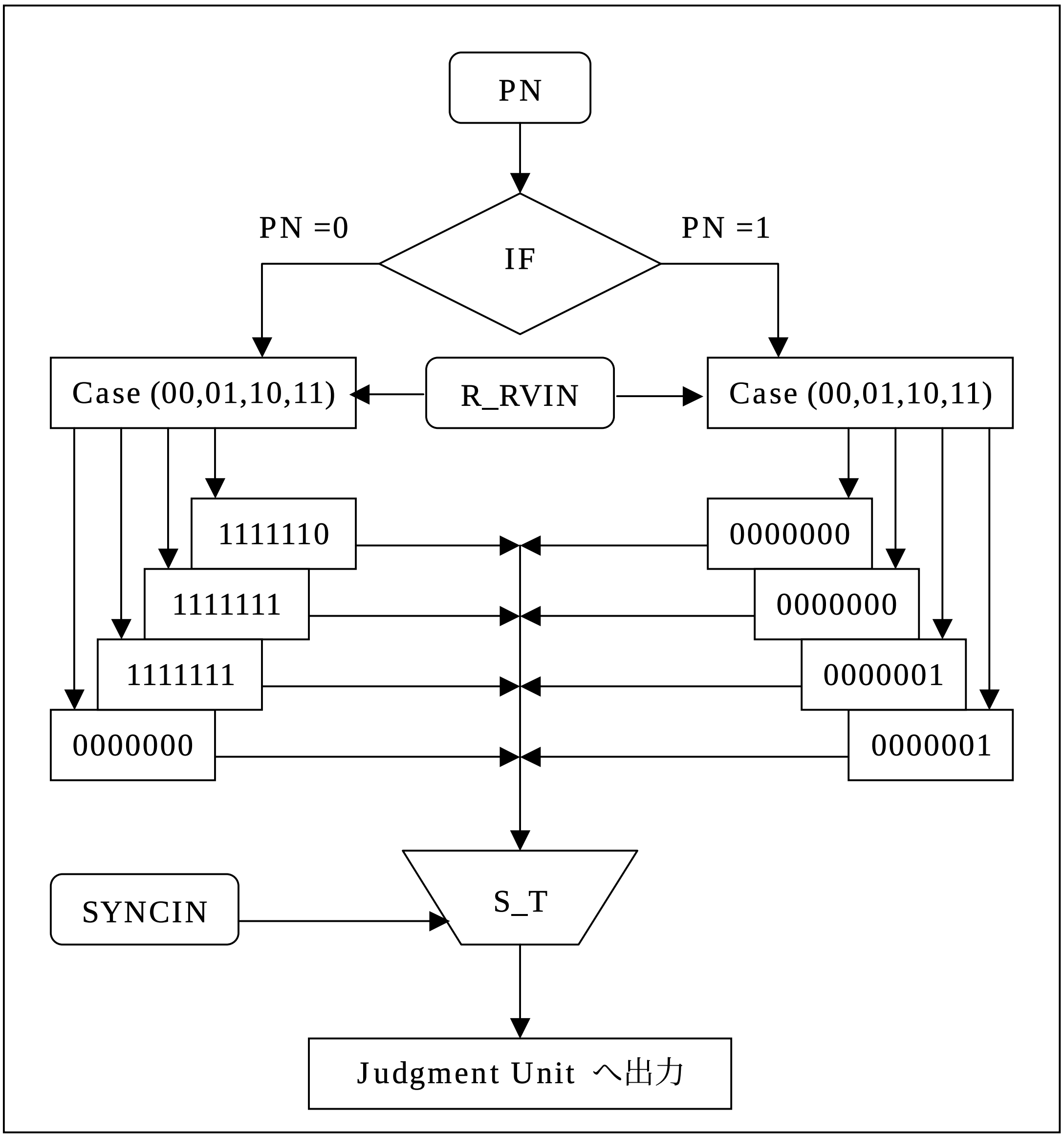


図 4.9　Expansion フローチャート

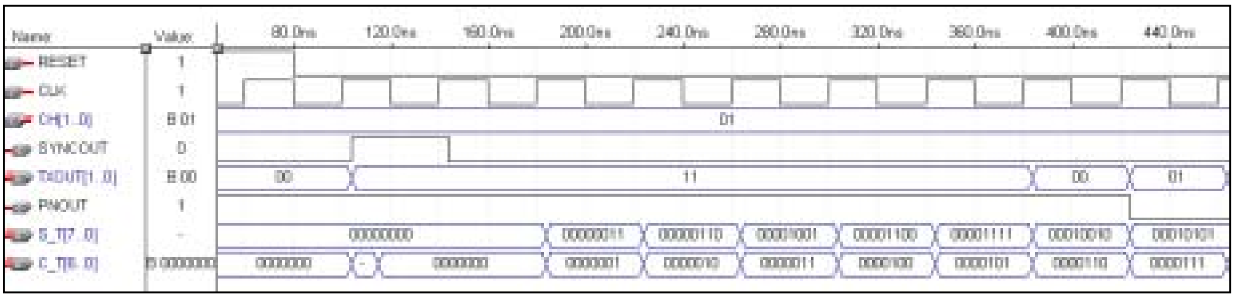


図 4.10　Addition 動作波形

4.3.3　Judgment ユニット（判定）

最後に Judgment ユニットについて説明する。このユニットは Addition ユニットから内積値（S_R）を受けり、復調をおこなう。復調は、Addition ユニットの加算回数が１２５回の時（Count＝１２５）の内積値を閾値にて判断する。ここで閾値の値は±40 としている。図 4.12 に動作波形を示す。

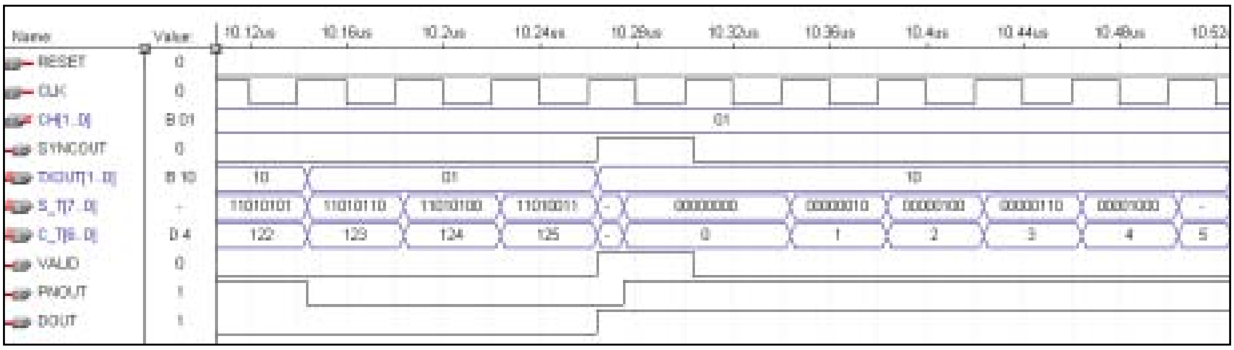


図 4.12　Judgment 動作波形

第５章　設計回路のシミュレーション

この章では第４章で述べた、CDMA 受信機のシミュレーション結果を示す。また ALTERA 社「max+plusⅡ」で製作した全体の回路図（シンボル）を図 5.1 に示し、Receiver（受信機）のユニット構成（シンボル）を図 5.2 に示す。

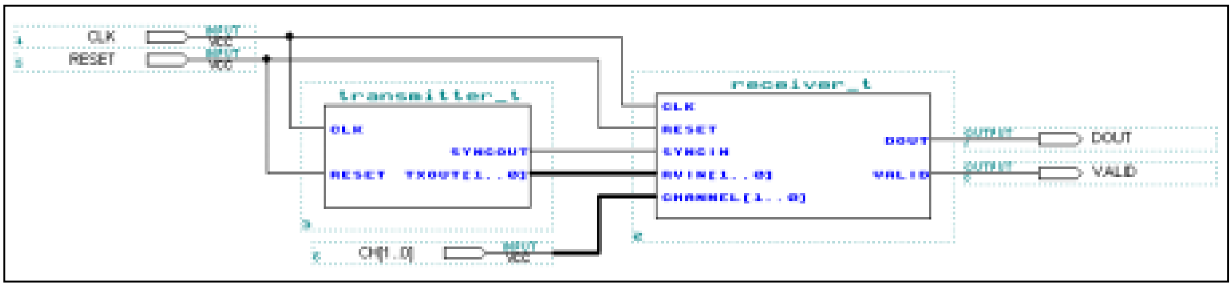


図 5.1　全体回路

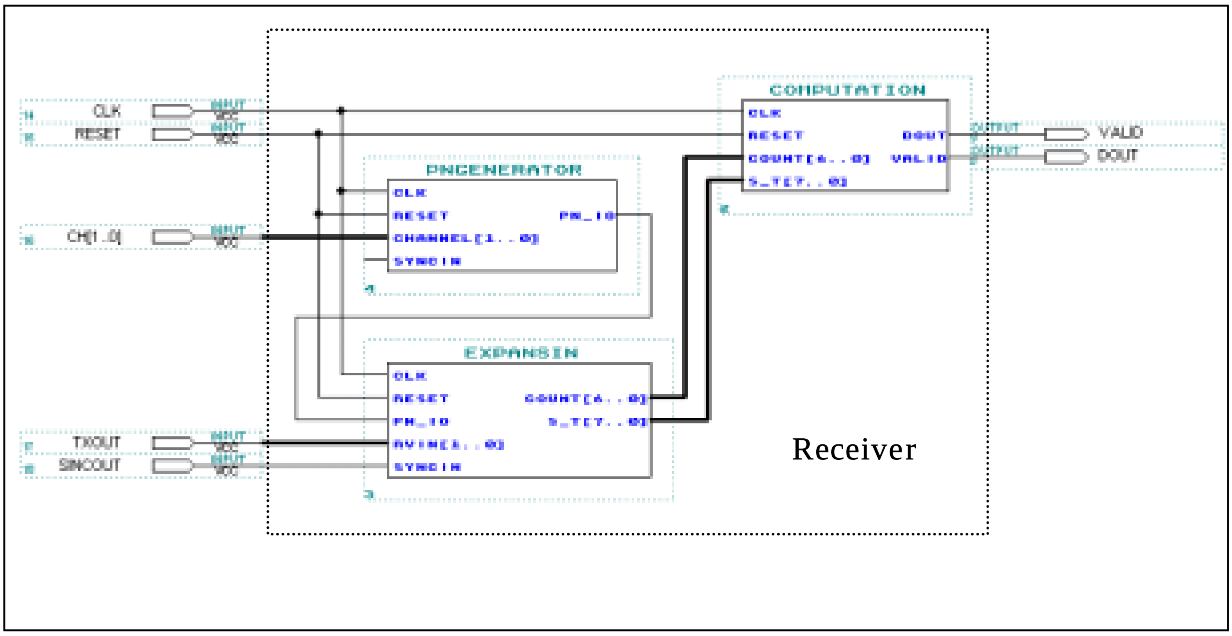


図 5.2　Receiver 回路構成

5.1　シミュレーション

この節では設計した回路を ALTERA 社「max+plusⅡ」を用いてシミュレーションした結果に示す。シミュレーションにあたっての仕様を表 5.1 に示す。各 CHANNEL での動作波形を図 5.3，5.4，5.5 に示す。

表 5.1　シミュレーション仕様

信号名	設定値
CLK	１周期　40ns
RESET	最初の 80ns　‘1’を出力
CHANNEL	それぞれの CH 信号

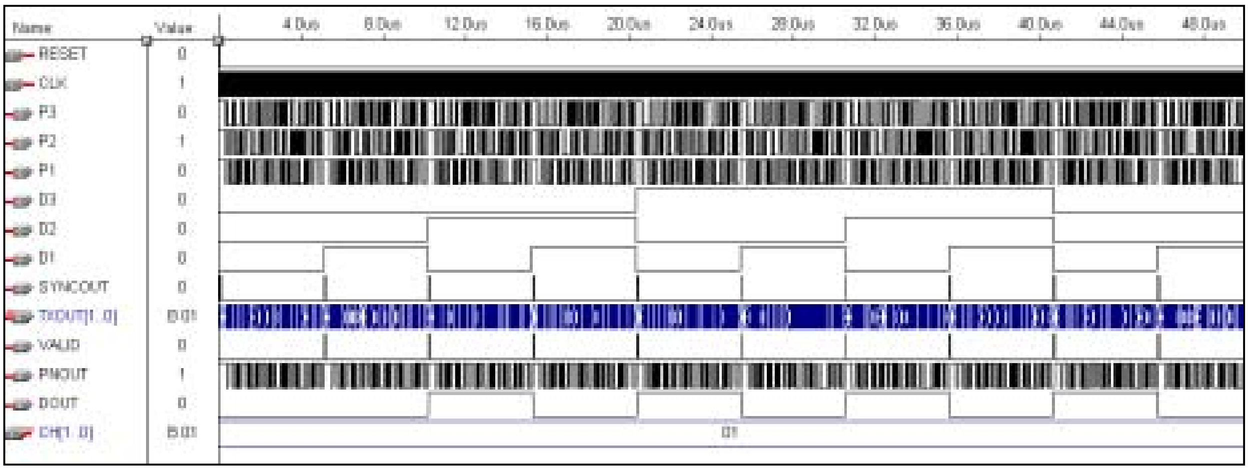


図 5.5　CH＝’01’の時の動作波形

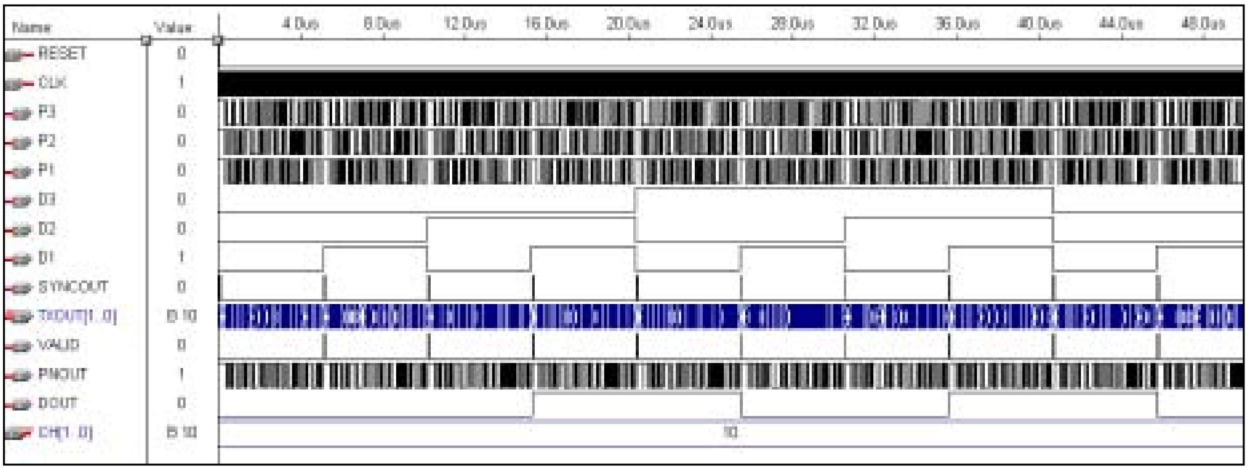


図 5.6　CH＝’10’の時の動作波形

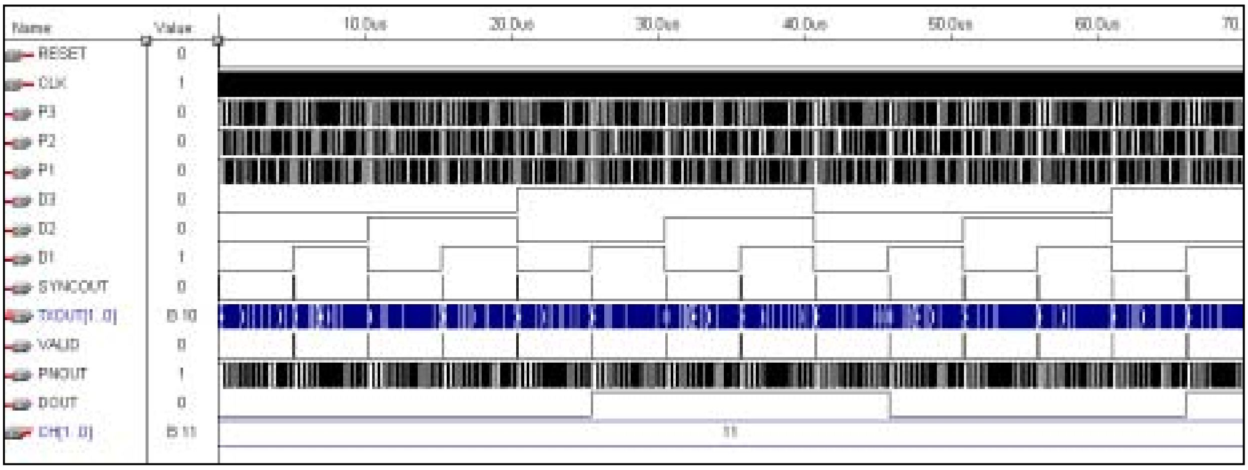


図 5.7　CH＝’11’の時の動作波形

CHANNEL ごとに送信データが復調されているのが、以上の波形で確認できた。また、内積の計算結果を確認するため図 5.8 に CH＝’01’の拡大図を示す。図 5.8 からわかるように計算結果が’80’となり、閾値の 40 を超えているため送信データが’0’であると確定し、VALID が出力されている。

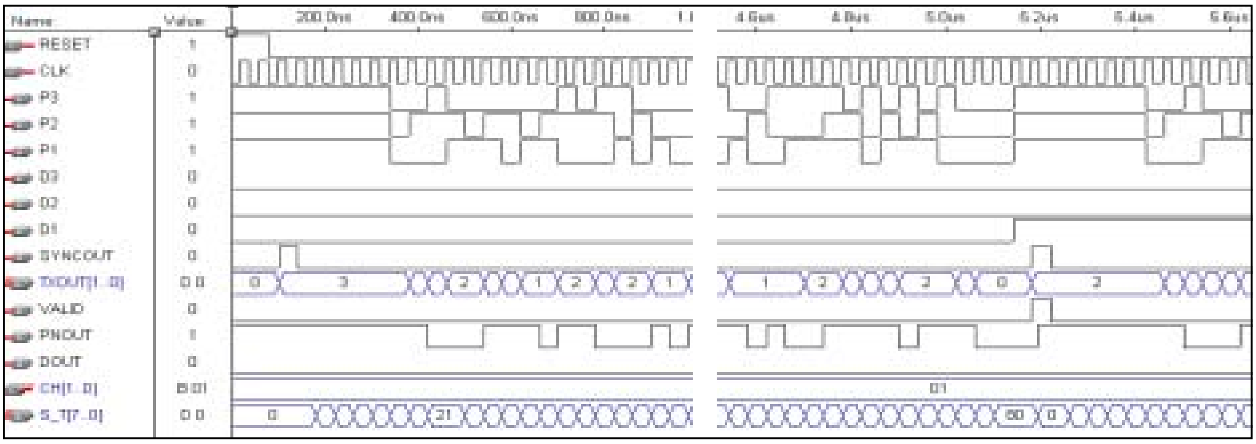


図 5.8　CH＝’01’の拡大図

5.2　論理合成

ここでは今回設計した回路（ユニット）ごとに論理合成した結果を表 5.2 に示す。今回、論理合成ツールとして Exemplar Logic 社「LeonardoSpectrum」を使用した。

表 5.2　論理合成結果

Unit　name	ゲート数	遅延時間
Transmitter	480	3.60
Receiver	661	5.09
PN_generator	110	2.46
Expansion	487	5.04
Computation	64	1.48

第6章　おわりに

卒業研究の目的は VHDL によるハードウェア設計をおこなうことであった。その題材としてCDMA受信機を選び設計結果について所望の動作を確認することができた。しかし、同期信号の処理や内積値の計算方法の良いアイデアがなかった為、設計した回路規模は大きくなり反省点も多く残った。また、今回は初期段階での設計が不十分だった為、幾度なく設計の見直し、時間のロスが出てしまった。設計をおこなうにあたって、きちんと順序だてて物事を考えていく事の大切さがわかった。

謝辞

本研究を進めるにあたり、日頃より懇切丁寧なご指導してくださいました、本学科電子光システム工学科 矢野政顕 教授に心より感謝いたします。また、日頃から多くの助言をしていただきお世話になりました電子光システム工学科の原央 教授、河津哲 教授ならびに橘昌良 助教授、他各先生方に厚くお礼申し上げます。同研究室の木村知史さん、新妻研作さん、中村基継さん、同学部生の仲間たちにも感謝いたします。

参考文献

［1］ デザインシノプスコンテスト 2001
http://bw-www.ie.u-ryukyu.ac.jp/~wada/design00/spec_j.html

［2］ 曽田 敏弘：中川研究室：多元接続方式
http://www.nkgw.ics.keio.ac.jp/jap/basic.html

［3］ 山内雪路：スペクトラム拡散通信　第２版 高性能ディジタル通信方式に向けて＊東京電気出版局 PP40~104（2001）

［4］ CDMAwow：http://www.ki.rim.or.jp/~fuji/cdma/

付録 1　VHDL プログラム

----送信機----

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

entity transmitter_s is
      Port ( CLK                    in std_logic;
                 RESET      in std_logic;
                 SYNCout   out std_logic;
                 P1          :           out std_logic; --PN1 code
                 P2          :           out std_logic; --PN2 code
                 P3          :           out std_logic; --PN3 code
                 D1          :           out std_logic; --PN1-DATE1
                 D2          :           out std_logic; --PN2-DATE2
                 D3          :           out std_logic; --PN3-DATE3
                 TXout      : out unsigned(1 downto 0)
             );
end transmitter_s;

architecture RTL of transmitter_s is


          signal counter    : unsigned(6 downto 0);
          signal pn1         : unsigned(6 downto 0);
          signal pn2         : unsigned(6 downto 0);
          signal pn3         : unsigned(6 downto 0);
          signal state      : unsigned(2 downto 0);
          signal date        : unsigned(2 downto 0);

begin
```

```
-- 127 counter & 8 counter --

cnt: process(CLK,RESET)
    begin
         if (RESET='1') then
                  counter <= (others => '0');
                  state     <= "000";
         elsif rising_edge(CLK) then
         if (counter = "1111110") then
                  counter <= (others => '0');
                  state     <= state+1;
         else
                  counter <= counter+1;
         end if;
         end if;
 end process cnt;

-- PN code generator --

pc: process(CLK,RESET)
begin
         if (RESET='1') then
                  pn1        <= (others => '1');
                  pn2        <= (others => '1');
                  pn3        <= (others => '1');

         elsif rising_edge(CLK) then
                  pn1(6 downto 1) <= pn1(5 downto 0);
                  pn1(0) <= pn1(6) xor pn1(2);
                  pn2(6 downto 1) <= pn2(5 downto 0);
                  pn2(0) <= pn2(6) xor pn2(2) xor pn2(1) xor pn2(0);
                  pn3(6 downto 1) <= pn3(5 downto 0);
                  pn3(0) <= pn3(6) xor pn3(3) xor pn3(2) xor pn3(1);
         end if;
end process pc;
```

```
-- DATE generator --

dt: process(state)
        begin
        case state is
                when "000"   => date <= "000";
                when "001"   => date <= "001";
                when "010"   => date <= "010";
                when "011"   => date <= "011";
                when "100"   => date <= "100";
                when "101"   => date <= "101";
                when "110"   => date <= "110";
                when "111"   => date <= "111";
                when others => date <= "XXX";
        end case;
end process dt;

-- TXout & SYNCout --

ts: process (CLK,RESET)
        begin
        if (RESET='1') then
                TXout    <= "00";
                SYNCout <= '0';

        elsif rising_edge(CLK) then
                TXout    <= ('0' & (pn1(6) xor date(0)))
                            +('0' & (pn2(6) xor date(1)))
                            +('0' & (pn3(6) xor date(2)));
        if (counter="0000000") then
                SYNCout <= '1';
        else
                SYNCout <= '0';
        end if;
        end if;
end process ts;
```

```
P1 <= pn1(6);
                P2 <= pn2(6);
                P3 <= pn3(6);
                D1 <= date(0);
                D2 <= date(1);
                D3 <= date(2);
end RTL;
```

----受信機----

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

entity receiver_s is
        Port (
                CLK             in    std_logic;
                RESET           in    std_logic;
                CHANNEL         in   unsigned(1 downto 0);
                RVIN            in   unsigned(1 downto 0);
                SYNCIN          in    std_logic;
                DOUT            out   std_logic;
                VALID           out   std_logic
                );
end receiver_s;

architecture RTL of receiver_s is

component pn_generator
        Port (
                CLK             in   std_logic;
                RESET           in   std_logic;
                CHANNEL         in   unsigned(1 downto 0);
                SYNCIN          in   std_logic;
```

```
                    PN_IO           out   std_logic
                    );
end component;

component expansin
          Port (
                    CLK             in   std_logic;
                    RESET           in   std_logic;
                    PN_IO           in   std_logic;
                    RVIN            in   unsigned(1 downto 0);
                    SYNCIN          in   std_logic;
                    COUNT           out unsigned(6 downto 0);
                    S_T             out unsigned(7 downto 0)
                    );
end component;

component computation
          Port (
                    CLK             in   std_logic;
                    RESET                   in   std_logic;
                    COUNT           in   unsigned(6 downto 0);
                    S_T             in   unsigned(7 downto 0);
                    DOUT            out std_logic;
                    VALID           out std_logic
                    );
end component;

          signal PN_IO              std_logic;
          signal S_T                unsigned(7 downto 0);
          signal COUNT              unsigned(6 downto 0);

begin

Z1 : pn_generator
       port map (CLK,RESET,CHANNEL,SYNCIN,PN_IO);
```

```
Z2 : expansin
        port map (CLK,RESET,PN_IO,RVIN,SYNCIN,COUNT,S_T);

Z3 : computation
        port map (CLK,RESET,COUNT,S_T,DOUT,VALID);

end RTL;
```

---- PN Code generator ----

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

entity pn_generator is
        Port (
                CLK             in   std_logic;
                RESET           in   std_logic;
                CHANNEL         in   unsigned(1 downto 0);
                SYNCIN          in   std_logic;
                PN_IO           out std_logic
                );
end pn_generator;

architecture RTL of pn_generator is

        signal pn       : unsigned(6 downto 0);

begin

--- PN Generator ---
process(CLK,RESET,SYNCIN)
        begin
```

```
			if(RESET='1' or SYNCIN='1') then
				pn <= (others => '1');
			elsif rising_edge(CLK) then
				pn(6 downto 1) <= pn(5 downto 0);
			case CHANNEL is
			when "01" => pn(0) <= pn(6) xor pn(2);
			when "10" => pn(0) <= pn(6) xor pn(2) xor pn(1) xor pn(0);
			when "11" => pn(0) <= pn(6) xor pn(3) xor pn(2) xor pn(1);
			when others => pn(0) <= 'X';
			end case;
			end if;
end process;

			PN_IO <= pn(6);

end RTL;
```

---- expansion ----

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

entity expansin is
		Port (
			CLK		in   std_logic;
			RESET		in   std_logic;
			PN_IO		in   std_logic;
			RVIN		in   unsigned(1 downto 0);
			SYNCIN		in   std_logic;
			COUNT		out unsigned(6 downto 0);
			S_T		out unsigned(7 downto 0)
			);
end expansin;

architecture RTL of expansin is
```

```
signal S_R              unsigned(7 downto 0);
signal R_RVIN           unsigned(1 downto 0);
signal counter          unsigned(6 downto 0);

begin

process(CLK,RESET)
        begin
                if(RESET ='1') then
                        R_RVIN <= "00";
                elsif rising_edge(CLK) then
                        R_RVIN <= RVIN;
                else
                        R_RVIN <= R_RVIN;
                end if;
end process;

process (CLK,RESET,SYNCIN)
        begin
                if(RESET='1' or SYNCIN='1') then
                        S_R      <= (others => '0');
                        counter <= (others => '0');
                elsif rising_edge(CLK) then
                if (PN_IO='1') then
                case R_RVIN is
                        when "00"    => S_R <= S_R + "00000000";
                        when "01"    => S_R <= S_R + "00000001";
                        when "10"    => S_R <= S_R + "00000010";
                        when "11"    => S_R <= S_R + "00000011";
                        when others => S_R <= S_R + "XXXXXXXX";
                end case;
                        counter <= counter + 1;
                else
                case R_RVIN is
                        when "00"    => S_R <= S_R + "00000000";
                        when "01"    => S_R <= S_R + "11111111";
```

```
                        when "10"      => S_R <= S_R + "11111110";
                        when "11"      => S_R <= S_R + "11111101";
                        when others => S_R <= S_R + "XXXXXXXX";
                end case;
                        counter <= counter + 1;
                end if;
                end if;
end process;

                S_T     <= S_R;
                COUNT <= counter;

end RTL;
```

----- Computation -----

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

entity computation is
        Port (
                CLK             in   std_logic;
                RESET           in   std_logic;
                COUNT           in   unsigned(6 downto 0);
                S_T             in   unsigned(7 downto 0);
                DOUT            out std_logic;
                VALID           out std_logic
                );
end computation;

architecture RTL of computation is

begin
```

```
process(CLK,RESET,COUNT)
        begin
                if (RESET='1')then
                        DOUT   <= '0';
                        VALID <= '0';
                elsif rising_edge(CLK) then
                if (COUNT ="1111101" and
                        S_T(7 downto 0) >= "10101000")then
                        DOUT   <= '1';
                        VALID <= '1';
                elsif (COUNT ="1111101" and
                        S_T(7 downto 0) >= "00101000")then
                        DOUT   <= '0';
                        VALID <= '1';
                else
                        VALID <= '0';
                end if;
                end if;
end process;

end RTL;
```

---- Test Bench-----

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;

ENTITY testbench IS
END testbench;

ARCHITECTURE cdma_testbench OF testbench IS

COMPONENT transmitter_s
        Port (
                CLK             in   std_logic;
                RESET                   in   std_logic;
                SYNCout         out std_logic;
                P1              out std_logic; --PN1 code
                P2              out std_logic; --PN2 code
                P3              out std_logic; --PN3 code
                D1              out std_logic; --PN1-DATE1
                D2              out std_logic; --PN2-DATE2
                D3              out std_logic; --PN3-DATE3
                Txout           out unsigned(1 downto 0)
                );
END COMPONENT;

COMPONENT receiver_s
        Port (
                CLK             in   std_logic;
                RESET           in   std_logic;
                CHANNEL         in   unsigned(1 downto 0);
                RVIN            in   unsigned(1 downto 0);
                SYNCIN          in   std_logic;
                DOUT            out std_logic;
                VALID           out std_logic
                );
END COMPONENT;
```

```
SIGNAL CLK           std_logic :='0';
SIGNAL CYCLES        integer :=0;
SIGNAL RESET         std_logic :='1';
SIGNAL P1            std_logic;
SIGNAL P2            std_logic;
SIGNAL P3            std_logic;
SIGNAL D1            std_logic;
SIGNAL D2            std_logic;
SIGNAL D3            std_logic;
SIGNAL SINCout       std_logic;
SIGNAL TXOUT         unsigned(1 downto 0);
SIGNAL CHANNEL       unsigned(1 downto 0) :="10";
SIGNAL DOUT          std_logic;
SIGNAL VALID         std_logic;

BEGIN

uut1: transmitter_s
        PORT MAP
        (CLK,RESET,SINCout,P1,P2,P3,D1,D2,D3,TXOUT);
uut2: receiver_s
        PORT MAP
        (CLK,RESET,CHANNEL,TXOUT,SINCout,DOUT,VALID);

CK : PROCESS
        BEGIN
        if (cycles < 6500) then
                cycles <= cycles + 1;
                wait for 10 ns;
                CLK <= not CLK;
        else wait;
        end if;
END PROCESS CK;
```

```
RT : PROCESS
        BEGIN
        loop1 : for n in 0 to 5 loop
                wait until falling_edge(CLK);
        end loop loop1;
                RESET <= '0';
END PROCESS RT;
```

付録２　論理合成リスト

----送信機----

```
*****************************************************
Cell: transmitter_s      View: RTL      Library: work
------------------------------------------------------------------------

Cell            Library    References        Total Area

AN2T0           scl05u      7 x      5       33 gates
AN4T0           scl05u      1 x      8        8 gates
FD1B0           scl05u     10 x      9       86 gates
FD1C0           scl05u     21 x      9      181 gates
FD1I0           scl05u      3 x     11       32 gates
HA1A0           scl05u      5 x      7       33 gates
IV1N0           scl05u      1 x      3        3 gates
IV1NP           scl05u      1 x      4        4 gates
MX2L0           scl05u      1 x      6        6 gates
ND2N0           scl05u      1 x      5        5 gates
ND4N1           scl05u      1 x      8        8 gates
NR4R1           scl05u      1 x      8        8 gates
OR4T0           scl05u      1 x      8        8 gates
XN2R0           scl05u      4 x      5       20 gates
XN3R0           scl05u      2 x      7       14 gates
XR2T0           scl05u      4 x      5       20 gates
XR3T0           scl05u      2 x      7       14 gates

 Number of ports                       :11
 Number of nets                        :75
 Number of instances                   :66
 Number of references to this view     :0

Total accumulated area         :
 Number of gates               :480
```

```
Info, Command 'report_area' finished successfully
->report_delay   -num_paths 1 -critical_paths -clock_frequency
Using default wire table: SCL_CORE_4K

Clock Frequency Report

Clock：Frequency
-----------------------------------
CLK  ：256.4 MHz

Critical Path Report

Critical path #1, (unconstrained path)
NAME            GATE            ARRIVAL                 LOAD
----------------------------------------------------------------------------------
clock information not specified
delay thru clock network                                                 0.00 (ideal)

ix55/Q                               FD1B0        0.00    0.60 up        0.27
counter_ix32_D0_I0_dup_231/CO        HA1A0        0.55    1.14up         0.17
counter_ix32_D0_I0_dup_232/CO        HA1A0        0.43    1.57up         0.17
counter_ix32_D0_I0_dup_233/CO        HA1A0        0.43    2.00up         0.17
counter_ix32_D0_I0_dup_234/CO        HA1A0        0.43    2.42up         0.17
counter_ix32_D0_I0_dup_235/CO        HA1A0        0.41    2.83up         0.14
ix1/X                                XR2T0        0.36    3.32dn         0.06
ix5/X                                AN2T0        0.28    3.60dn         0.11
counter(6)/D                         FD1B0        0.00    3.60dn         0.00
data arrival time                                                        3.60

data required time                          not specified
----------------------------------------------------------------------------------
data required time                          not specified
data arrival time                           3.60
                                            ----------
                                            unconstrained path
----------------------------------------------------------------------------------
```

----受信機----

```
****************************************************
Cell: receiver_s     View: RTL     Library: work
****************************************************


Cell              Library    References       Total Area

AN2T0             scl05u        2 x       5        9 gates
AN3T0             scl05u        1 x       6        6 gates
AO1A0             scl05u        1 x       6        6 gates
AO1I0             scl05u        1 x       6        6 gates
AO2A0             scl05u        7 x       8       55 gates
AO2L0             scl05u        1 x       8        8 gates
CONZ0             scl05u        1 x       1        1 CONZ0
FA2A0             scl05u        4 x      15       60 gates
FD1B0             scl05u        9 x       9       77 gates
FD1B1             scl05u        1 x       9        9 gates
FD1C0             scl05u        7 x       9       60 gates
FD1I0             scl05u        9 x      11       95 gates
HA1A0             scl05u        5 x       7       33 gates
IV1N0             scl05u        5 x       3       16 gates
IV1N2             scl05u        1 x       3        3 gates
MX2L0             scl05u        7 x       6       43 gates
ND2N0             scl05u        3 x       5       14 gates
ND4N0             scl05u        1 x       8        8 gates
NR2R0             scl05u        2 x       5        9 gates
NR2R1             scl05u        1 x       5        5 gates
NR2R2             scl05u        3 x       5       15 gates
NR3R0             scl05u        1 x       6        6 gates
OA1R0             scl05u        1 x       6        6 gates
OAI3R0            scl05u        1 x       8        8 gates
OAOI0             scl05u        1 x       8        8 gates
XN2R0             scl05u        3 x       5       15 gates
XN3R0             scl05u        3 x       7       20 gates
```

```
XR2T0         scl05u        11 x       5       54 gates
XR3T0         scl05u        1 x        7        7 gates

 Number of ports                   :9
 Number of nets                    :115
 Number of instances               :94
 Number of references to this view :0

Total accumulated area             :
 Number of CONZ0                   :1
 Number of gates                   :661

Info, Command 'report_area' finished successfully
->report_delay   -num_paths 1 -critical_paths -clock_frequency
Using default wire table: SCL_CORE_4K

Clock Frequency Report

Clock         :Frequency
-----------------------------------
CLK           :172.6 MHz

Critical Path Report

Critical path #1, (unconstrained path)
NAME          GATE          ARRIVAL             LOAD
------------------------------------------------------------------------------------
clock information not specified
delay thru clock network                                              0.00 (ideal)

Z2_reg_R_RVIN(0)/Q            FD1B0         0.00    0.53dn                  0.36
Z2_ix607/X                    NR2R2         0.95    1.47up                  1.39
Z2_ix617/X                    XR2T0         0.82    2.29 up                 0.22
Z2_ix125/X                    MX2L0         0.66    2.95 up                 0.19
Z2_ix630/X                    MX2L0         0.32    3.27 dn                 0.18
Z2_ix169/X                    MX2L0         0.39    3.66 up                 0.19
```

```
Z2_ix645/X              MX2L0      0.32   3.98 dn        0.18
Z2_ix213/X              MX2L0      0.31   4.29 up        0.08
Z2_ix217/X              XN3R0      0.32   4.65 dn        0.06
Z2_ix223/X              AO2A0      0.45   5.09 dn        0.06
Z2_ix225/D              FD1I0      0.00   5.09 dn        0.00
data arrival time                                        5.09

data required time           not specified
-----------------------------------------------------------------------------------------
data required time           not specified
data arrival time            5.09
                             ----------
                             unconstrained path
-----------------------------------------------------------------------------------------
```

----PN - generator----

```
*****************************************************
Cell: pn_generator     View: RTL     Library: work
*****************************************************


Cell          Library   References       Total Area


FD1C0         scl05u      7 x      9      60 gates
IV1N0         scl05u      1 x      3       3 gates
MX2L0         scl05u      1 x      6       6 gates
ND2N0         scl05u      1 x      5       5 gates
NR2R2         scl05u      1 x      5       5 gates
OAI3R0        scl05u      1 x      8       8 gates
XN2R0         scl05u      1 x      5       5 gates
XN3R0         scl05u      1 x      7       7 gates
XR2T0         scl05u      1 x      5       5 gates
XR3T0         scl05u      1 x      7       7 gates
```

```
 Number of ports :                         6
 Number of nets :                         21
 Number of instances :                    16
 Number of references to this view :       0

Total accumulated area :
 Number of gates :                       110

Info, Command 'report_area' finished successfully
->report_delay   -num_paths 1 -critical_paths -clock_frequency
Using default wire table: SCL_CORE_4K

                         Clock Frequency Report

Clock                  : Frequency
-----------------------------------
CLK                     : 361.9 MHz

Critical Path Report

Critical path #1, (unconstrained path)
NAME          GATE          ARRIVAL              LOAD
---------------------------------------------------------------------------
clock information not specified
delay thru clock network                                          0.00 (ideal)

reg_pn(2)/Q          FD1C0              0.00   0.57 up              0.33
ix131/X              XN2R0              0.61   1.43 dn              0.14
ix141/X              XR3T0              0.39   1.82 dn              0.11
ix25/X               MX2L0              0.28   2.09 up              0.06
ix135/X              ND2N0              0.14   2.23 dn              0.06
ix29/X               OAI3R0             0.23   2.46 up              0.11
reg_pn(0)/D          FD1C0              0.00   2.46 up              0.00
data arrival time                                                   2.46
```

```
data required time                  not specified
------------------------------------------------------------------------
data required time                  not specified
data arrival time                   2.46
                                    ----------
                                    unconstrained path
------------------------------------------------------------------------
```

----Expansion----

```
**********************************************************
Cell: expansion      View: RTL      Library: work
**********************************************************


Cell             Library    References       Total Area


AN2T0            scl05u        2 x         5         9 gates
AO1A0            scl05u        1 x         6         6 gates
AO1I0            scl05u        1 x         6         6 gates
AO2A0            scl05u        7 x         8        55 gates
AO2L0            scl05u        1 x         8         8 gates
CONZ0            scl05u        1 x         1         1 CONZ0
FA2A0            scl05u        4 x        15        60 gates
FD1B0            scl05u        8 x         9        69 gates
FD1B1            scl05u        1 x         9         9 gates
FD1I0            scl05u        8 x        11        85 gates
HA1A0            scl05u        5 x         7        33 gates
IV1N0            scl05u        2 x         3         6 gates
IV1N2            scl05u        1 x         3         3 gates
MX2L0            scl05u        6 x         6        37 gates
ND2N0            scl05u        1 x         5         5 gates
NR2R0            scl05u        2 x         5         9 gates
NR2R1            scl05u        1 x         5         5 gates
NR2R2            scl05u        2 x         5        10 gates
XN2R0            scl05u        2 x         5        10 gates
```

```
XN3R0          scl05u      1 x       7        7 gates
XR2T0          scl05u     10 x       5       49 gates
XR3T0          scl05u      1 x       7        7 gates

 Number of ports :                          21
 Number of nets :                           88
 Number of instances :                      68
 Number of references to this view :         0

Total accumulated area :
 Number of CONZ0 :                             1
 Number of gates :                           487

Info, Command 'report_area' finished successfully
->report_delay   -num_paths 1 -critical_paths -clock_frequency
Using default wire table: SCL_CORE_4K

Clock Frequency Report

Clock                  : Frequency
------------------------------------
CLK                     : 174.1 MHz

Critical Path Report

Critical path #1, (unconstrained path)
NAME                   GATE          ARRIVAL              LOAD
------------------------------------------------------------------------------------
clock information not specified
delay thru clock network                                                 0.00 (ideal)

ix1/Q                       FD1B1          0.00     0.53 dn                  0.53
ix607/X                     NR2R2          0.89     1.42 up                  1.39
ix617/X                     XR2T0          0.82     2.24 up                  0.22
ix125/X                     MX2L0          0.66     2.90 up                  0.19
ix630/X                     MX2L0          0.32     3.22 dn                  0.18
```

```
ix169/X                          MX2L0        0.39    3.61 up              0.19
ix645/X                          MX2L0        0.32    3.92 dn              0.18
ix213/X                          MX2L0        0.31    4.24 up              0.08
ix217/X                          XN3R0        0.32    4.60 dn              0.06
ix223/X                          AO2A0        0.45    5.04 dn              0.06
ix225/D                          FD1I0        0.00    5.04 dn              0.00
data arrival time                                                          5.04

data required time                       not specified
---------------------------------------------------------------------------------
data required time                       not specified
data arrival time                        5.04
                                         ----------
                                         unconstrained path
---------------------------------------------------------------------------------
```

----Computation----

```
*******************************************************
Cell: computation     View: RTL     Library: work
*******************************************************


Cell            Library   References        Total Area


AN3T0           scl05u       1 x       6       6 gates
FD1B0           scl05u       1 x       9       9 gates
FD1I0           scl05u       1 x      11      11 gates
IV1N0           scl05u       2 x       3       6 gates
ND2N0           scl05u       1 x       5       5 gates
ND4N0           scl05u       1 x       8       8 gates
NR3R0           scl05u       1 x       6       6 gates
OA1R0           scl05u       1 x       6       6 gates
OAOI0           scl05u       1 x       8       8 gates
```

```
 Number of ports :                        19
 Number of nets :                         24
 Number of instances :                    10
 Number of references to this view :       0

Total accumulated area :
 Number of gates :                        64

Info, Command 'report_area' finished successfully
->set report_delay_slack_threshold 0
0
->report_delay   -num_paths 1 -critical_paths -clock_frequency
Using default wire table: SCL_CORE_4K

Clock Frequency Report

Clock                    : Frequency
------------------------------------
CLK                       : 477.8 MHz

Critical Path Report

Critical path #1, (unconstrained path)
NAME                     GATE         ARRIVAL               LOAD
------------------------------------------------------------------------------
COUNT(0)/                                  0.00    0.00 up                 0.06
ix13/XND4N0                                0.16    0.16 dn                 0.06
ix141/X                  NR3R0             0.56    0.72 up                 0.12
ix45/X                   OA1R0             0.76    1.48 up                 0.19
ix55/EN                  FD1I0             0.00    1.48 up                 0.00
data arrival time                                                          1.48

data required time                    not specified
```

```
--------------------------------------------------------------------------------
data required time                    not specified
data arrival time                     1.48
                                      ----------
                                      unconstrained path
--------------------------------------------------------------------------------
```