２．ディジタル通信

1. 目的

本実験では，簡単なデータ伝送 (送信・受信) 回路の設計を行い，ディジタル通信における基本的な伝送手法について理解する．

2. データ通信

2.1. 並列伝送と直列伝送

パソコンなど各種計算機では，通常ビットデータをバイト (8 ビット) やワード (16 ビット)，ダブルワード (32 ビット) 単位で扱う．そのためデータ伝送についても，この単位で行われることが普通である．

図 1 のように，扱うデータ単位分の通信線を並列に並べて一度に伝送する方式を**並列 (パラレル) 伝送**と呼ぶ．この方式では回路は簡単になり一度に多くのデータを伝送できるが，多数の通信線が必要になるため，通常は装置内あるいは建物内など近距離のデータ伝送に利用される．

一方図 2 のように，データを一度バッファに蓄え，1 本の通信線を使い順次 1 ビットずつ伝送する方式を**直列 (シリアル) 伝送**と呼ぶ．直列伝送では，シフトレジスタを使って直並列の変換を行わねならないため，回路は複雑となる．しかし，遠隔地との通信も 1 本の通信路で行えるという特徴がある．

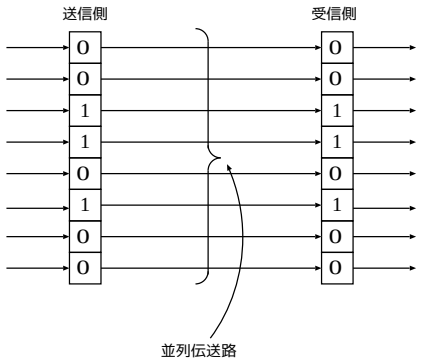

図 1: 並列伝送

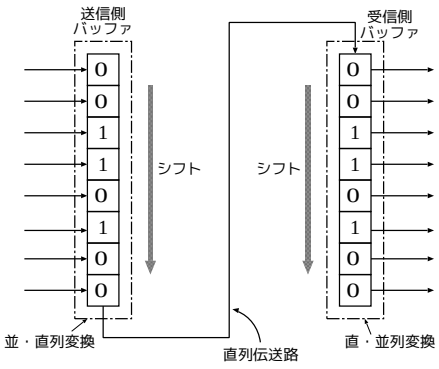

図 2: 直列伝送

2.2. 通信方式

データ通信システムにおける通信方式には，通信路の形態に関係なく，情報の流れる方向から次の 3 種類に分類することができる．

- **単向通信 (simplex)**
  情報の流れが A から B というように常に一定している通信方式で，逆方向には情報を送ることができない．(但し，誤り制御のためのアンサ・バックのような監視信号を B から A に送る場合の含める．)
- **半二重通信 (half-duplex)**
  情報の流れが，あるときは A から B，あるときは B から A というように変化する通信方式．この方式では同時に両方向の伝送はできない．
- **全二重通信 (full-duplex)**
  A から B の方向と，B から A の方向の情報伝送を同時に行うことができる通信方式．

2.3. 回線の構成

特に直列伝送の場合は，その構成方法により次のように 2 方式に分類される．

【1】2 線式 (2W)

信号線と帰路 (グラウンド) の 2 本の信号線で構成される．双方向の通信の場合は上り/下りとも同一伝送路を使用する．

【2】4 線式 (4W)

2 線式の伝送路を 2 組使用する．上り/下りの伝送路は別々である．

2.4. 同期方式

シリアル伝送では，データは一連の 2 進符号で表現されるが，受信側ではビットの区切りを時間的に識別しないと，送られてきたデータを受けとることができない．したがって，送信側では，ビットの区切りを識別するための情報を送る必要がある．これを**同期信号**と呼ぶが，この同期情報を得る方法としては，次の 2 方式がある．

【1】調歩式 (非同期式)

各キャラクタの先頭にスタート・ビット，最後にストップ・ビットを付加し，1 キャラクタごとにビット同期をとる方式である．

この方式では，キャラクタの間隔を任意にとることができる利点があるが，反面各キャラクタごとにスタート・ビット，ストップ・ビットを付加する必要があるため，例えば 8 ビットのデータを送信するときには最低 10 ビットのデータを送信しなければならない．

【2】同期式

一定のクロックに同期させてデータの送受信を行う方式で，同期を確立する方法として，ビット同期，キャラクタ同期，フレーム同期がある．

- **ビット同期**：ビットの位置合わせは，PAD_L(リーディング・パッド：ビット同期確立のために同期キャラクタ SYN の前に挿入する伝送制御キャラクタ) で行う．
- **キャラクタ同期**：PAD_L に続く 2 文字以上の SYN キャラクタでとられる．
  キャラクタ同期が確立した後は，一定間隔ごとに同期信号を挿入する．
- **フレーム同期**：ブロック同期，テキスト同期とも呼ばれ，同期方法はキャラクタ同期と同じである．フレーム同期では，あらかじめ定められたフレーム同期パターンを見い出し，その後，キャラクタの同期をとる．

2.5. 回線形態

データ通信システムは，用途によってタイム・シェアリング方式，リアルタイム方式，リモート・バッチ方式など色々な形態に分けられる．また，データ処理装置同士の接続形態で分類すると，次に示すような非交換回線と交換回線に分けることができる．

● 非交換回線 (図 3)

(1) **直通回路**

ポイント・トゥ・ポイントあるいは専用回線と呼ばれるもので，二つのデータ処理装置が一つの通信路を専有し，常時接続されている形態．

(2) **分岐回路**

マルチ・ドロップ回路と呼ばれ，一つの通信路に複数の端末装置が接続される形態．

(3) **分割多重型**

伝送路を時分割あるいは周波数分割して用いる形態.

(4) **集信型**

多数の端末からのデータ集信を目的としたシステム形態.

(5) **配信型**

多数の端末へのデータの配信を目的としたシステム形態.

(6) **ループ型**

ひとつ の伝送路をループとして用いる形態.

● **交換回線 (図 4)**

(1) **回線交換型**

電話網と同様な構成で，トラフィック量に応じて伝送路の低減を目的としたもの.

(2) **蓄積交換型**

端末と基幹コンピュータ間の情報をメッセージまたはパケットと呼ばれる単位で，一旦メモリに蓄積したあと宛先に応じて交換するシステム.

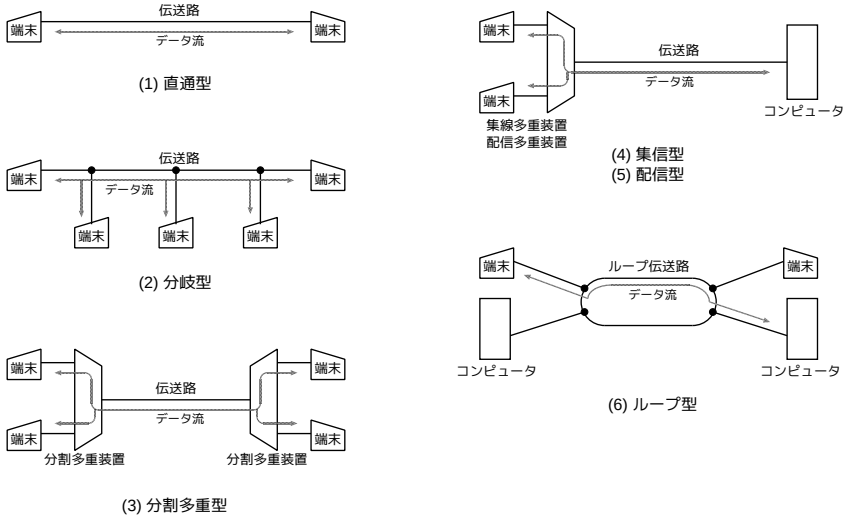

図 3: データ通信方式 (非交換型)

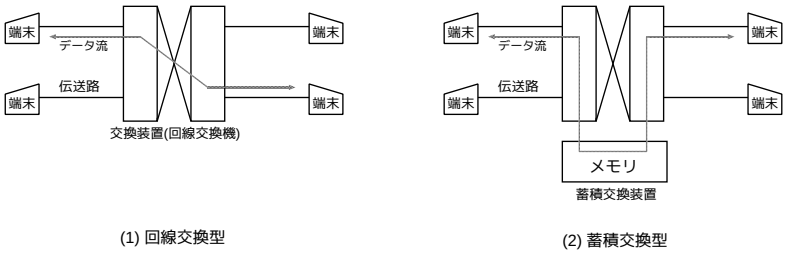

図 4: データ通信方式 (交換型)

3. 実験

本実験では，Quartus II を用いて単一の 8 ビットデータのパラレル伝送を行う送信回路，受信回路の設計，論理シミュレーションを行い，その後 DE0 ボードを用いて実際の動作検証を行う．

3.1. データ送受信方式

本実験で設計するデータ送受信方式の模式図を図 5 に示す．この方式は，以前よりパソコン通信等で用いられていたプロトコルである XMODEM と呼ばれる非同期伝送方式を基本としたものである．本実験では，1 つの 8 ビットデータの送受信を行うパラレル伝送回路を設計するが，図 5 の① 〜 ③ を繰り返すことにより，多数の 8 ビットデータを伝送することができる．なお，今回はストップビットやパリティビットは実装していない．

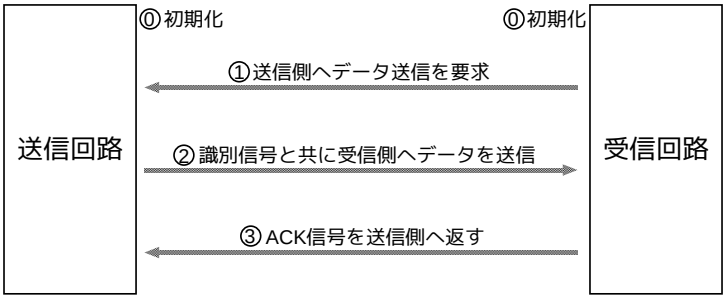

図 5: データ送受信の流れ

設計する伝送方式では，一回に８つの１ビットデータを一度に伝送します．送受信部の動作を各々フローチャートで表すと図 6 のようになる．

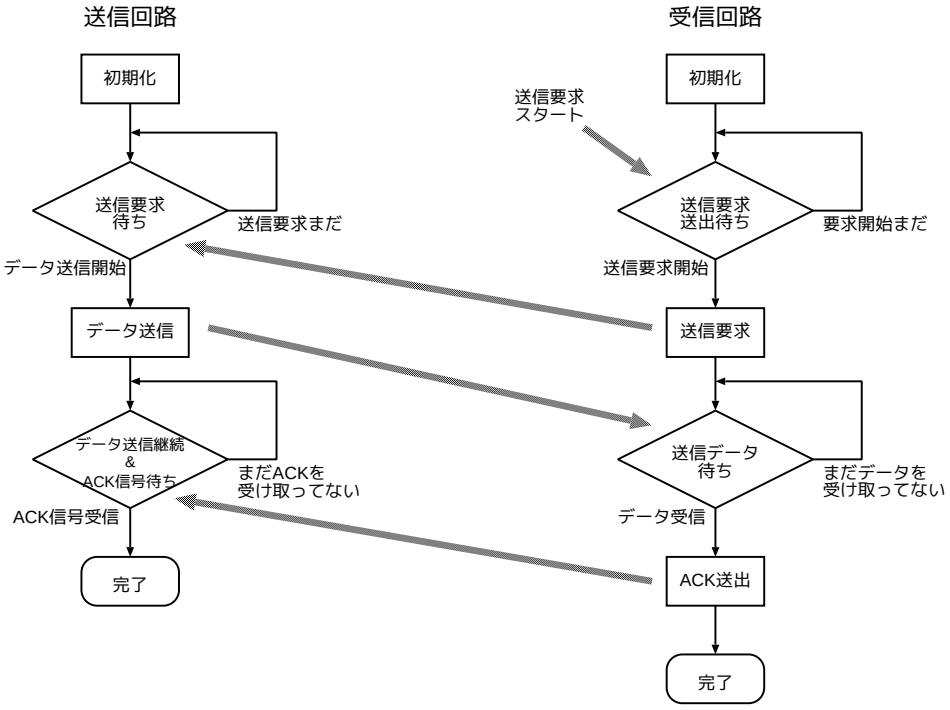

図 6: データ送受信の流れ (フローチャート)

この実験で設計する回路は複数の**状態**を持ち，ある状態にある送信回路・受信回路は

a) 現在の自分の状態（データ待ち中か、データ送信中か、など）

b) 現在の状態で入力される信号

など，現在の入力だけでなくその前の時点での状態などの条件によって、次にどのような状態になるか、または次の状態でどのような信号が出力されるのかが決定される．このような回路は**順序回路**とよばれ、**状態遷移図**で表現することができる．[1]．VHDL のようなハードウェア記述言語では，状態遷移図で表現される回路の動作を，容易にディジタル回路として実現することができる．本実験で設計する送信回路，受信回路の状態遷移図を図 7，図 8 に示す.

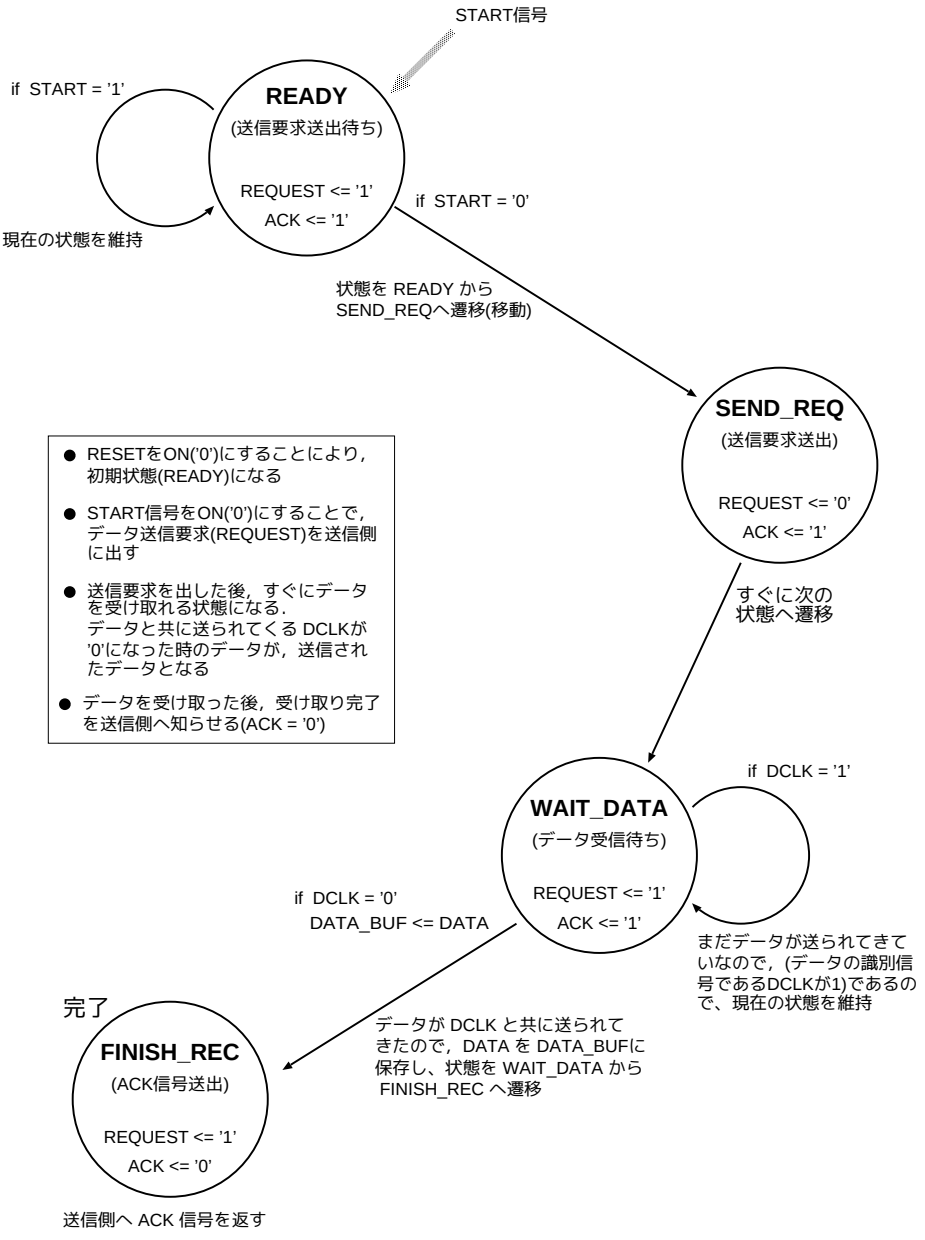

図 7: 受信回路の状態遷移図

[1]これに対し，ハードウェア実験 - 「2．論理設計の基礎」で設計した全加算器など，出力が入力のみにより決定される回路は**組み合せ回路**と呼ばれる

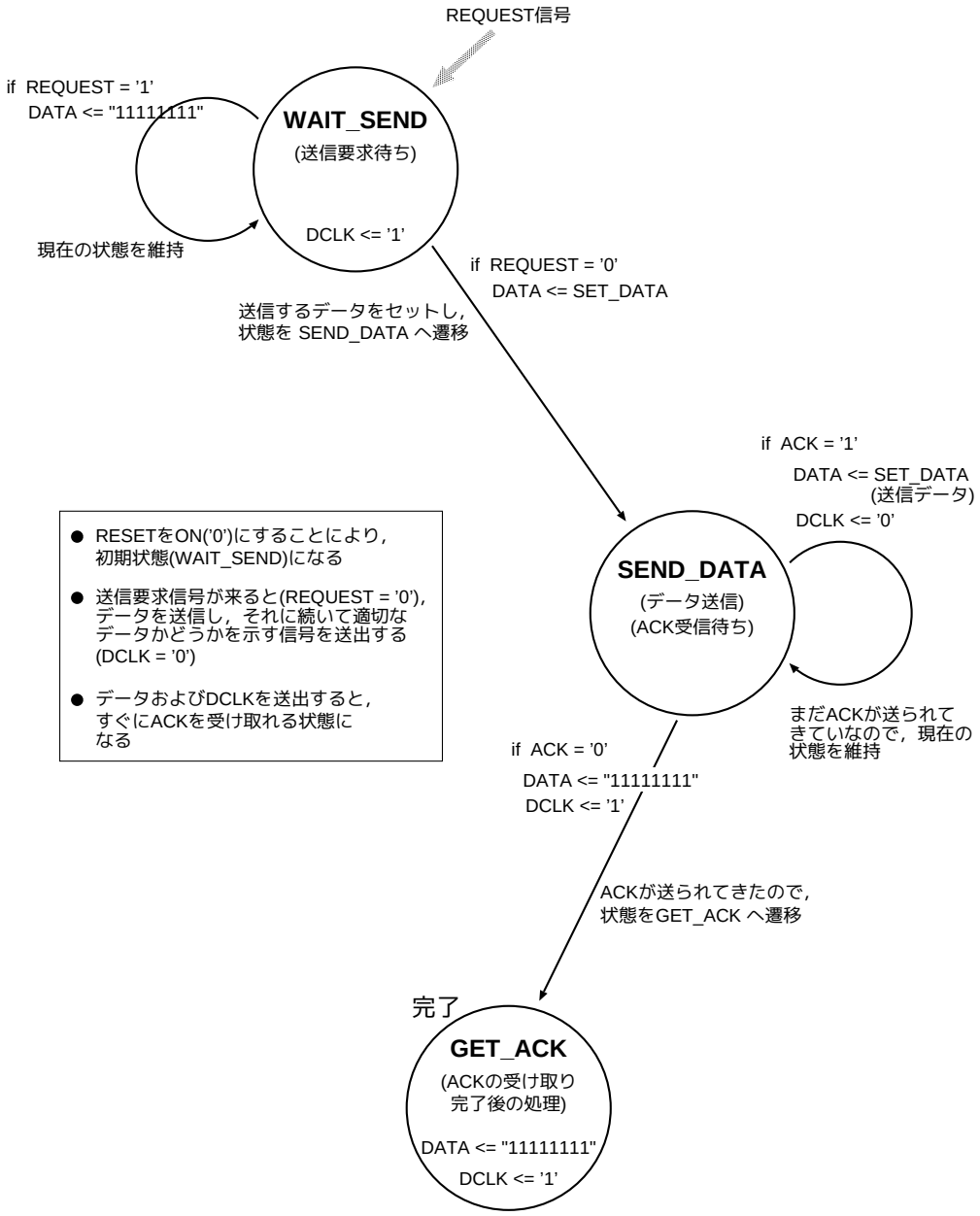

図 8: 送信回路の状態遷移図

3.2. 状態遷移図について

○は各回路の持つ状態、矢印が状態が変化する向き

I. ○内の上側は各状態の名前

II. ○内の下側は各状態での出力信号

III. ２本以上の矢印が出ている状態は，入力信号によって次の状態が決定される．

3.3. 状態遷移図の VHDL による回路設計 (プログラム設計)

受信回路を例に，状態遷移図と VHDL 記述の対応を図 9 に示す．このプログラム例では，状態を表す変数 **STATE** をキーとする **case 文**により状態遷移を実現している．各状態は 「**when XXXXX =>**」 で定義され，各々にその状態の動作を記述する．状態遷移図で各円内で示しているのがその状態での出力信号である．

図 9 で READY の状態にあるとき，出力信号は REQUEST, ACK ともに '1' である．この状態で，外部からの START 信号が '0' になったら次の状態 (SEND_REQ) に遷移することになるので，これを if 文を用いて，次の状態を STATE に代入することで実現する．他の状態についても同じく記述する．送信回路についても同様である．

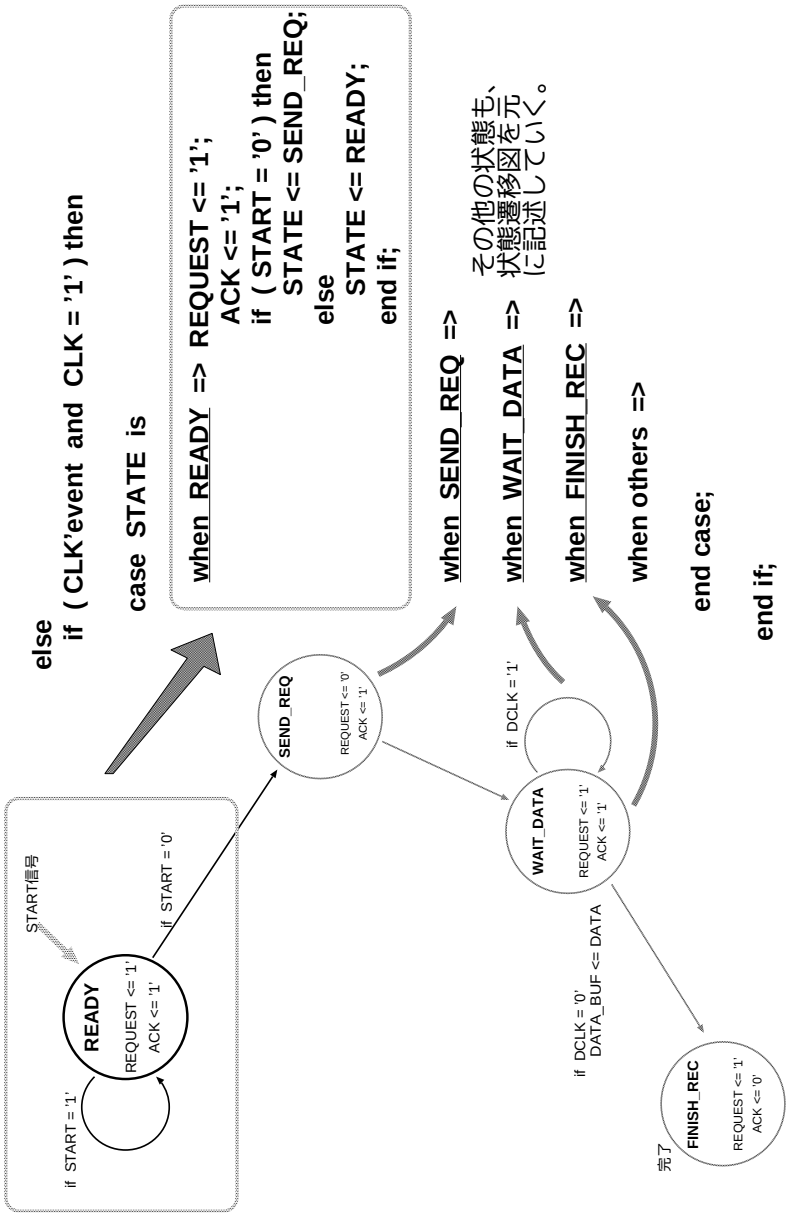

図 9: 状態遷移図の VHDL 記述例

3.4. 実験概要

3.4.1. 受信回路・送信回路の設計

【1】受信回路・受信回路の状態遷移図を用い、コピーした input_core.vhd に加筆して受信回路を設計する。(作成したプログラムは レポートに添付)

【2】受信回路の論理シミュレーション入力を用いて，同じ出力結果になるかどうかを Quartus II のシミュレータにより確認する。(シミュレーショ結果もレポートに添付)

(重要) 本年度は全員「受信回路」を設計して下さい。

3.4.2. FPGA へ転送する回路の生成および DE0 ボード上での動作検証

【1】既に用意してある回路ファイル (ＬＥＤ表示回路と、受信回路~~または送信回路~~とＬＥＤ表示回路を動作させるためのベース回路) を組み合わせて、DE0 ボードで実際に動作させるための受信回路~~、送信回路~~を各々コンパイル→ピン配置→再コンパイルする。

【2】コンパイルした受信回路、送信回路を各々DE0 ボードにダウンロードし、ボード間を平ケーブルで接続し動作検証を行う。

※ 受信回路、送信回路双方をリセットし，その後スタートボタンを押して送信側で設定したデータ (LED 上の値) が受信側に転送されればＯＫ。

この段階で、実験担当からシミュレーション結果の確認と FPGA ボードでの動作確認をしてもらって下さい。(送信回路側は用意してあります。)

3.5. 実験装置

- **パソコン：2 台以上**
- **FPGA ボード (DE0 ボード)：2 台以上**
- **DE0 ボード間のデータ伝送用の平ケーブル**
- **実験参考文書等**

~~各班で送信回路・受信回路それぞれを設計する人を決めて，各回路同時進行で作業して下さい。~~

3.6. (実験１) 受信回路・送信回路の設計

実験で使用するソフトウェア **Quartus II**、FPGA ボードの使い方、使用上の注意については『ハードウェア実験 - ２．論理設計の基礎』のときと同様である.

3.6.1. ＜ Part 1 ＞ 〜 受信・送信回路の設計 〜

各班にある「Quartus II の使い方」を参考にして設計して下さい。

【1】作業用フォルダの作成

コンピュータの画面の左下の丸い部分 (スタート) から「コンピュータ」を開き、さらにその中の「ネットワークの場所」にある windows ディスクを開き、その中に班のアルファベット名のフォルダを作成して下さい。**日本語等の全角文字は使用できません。**

【2】以下の URL から受信回路~~または送信回路~~のファイルを作成した作業フォルダにコピーします。

http://wiki.cis.iwate-u.ac.jp/~toy/EXP/network/

- 受信回路設計側
  - input_core.vhd (受信回路のサンプルファイル)
  - input_base.vhd (受信回路のベースファイル)
  - disp_led.vhd (送受信データ表示用回路ファイル)
- ~~送信回路設計側~~
  - ~~output_core.vhd (送信回路のサンプルファイル)~~
  - ~~output_base.vhd (送信回路のベースファイル)~~
  - ~~disp_led.vhd (送受信データ表示用回路ファイル)~~

 # 本実験での送受信回路の設計は、input_core.vhd、~~output_core.vhd~~ のファイルを修正することで行います。以下からは受信回路、送信回路とも同じ作業です。

【3】Quartus II の起動 (重要！)

デスクトップ上には Quartus II のアイコンが２つありますが、必ず Quartus II 9.1 の方を使用して下さい。

【4】プロジェクト名の設定 (「Quartus II の使い方」 P5〜P10)

- (使い方 P6)：プロジェクト名は，受信回路では **input_core**、~~送信回路では **output_core**~~ として下さい。
- (使い方 P7)：画面右上にある「Add」をクリックして，**input_core**~~または **output_core**~~ を登録します。そして「Next」をクリック。
- (使い方 P8)：ウインドウ上の Family: は **Cyclone III** を選択し、その後下の Available devices: で **EP3C16F484C6** を選択して「Next」をクリック。
- (使い方 P9)：そのまま「Next」をクリック。

【5】サンプルファイルのオープン

- メイン画面左上の「Entry」と書かれてあるところのピラミッドの下にある **input_core**~~または **output_core**~~ をクリックすると，登録したファイルが開きます。

【6】回路設計 (記述)：開いたファイル内に必要な記述を追加します。

- **注)entity 名や入出力ポート名は変更しないで下さい。**

【7】**設計回路の合成** (「Quartus II の使い方」 P13〜P15)

【8】**論理シミュレーション** (「Quartus II の使い方」 P19〜P27、P32〜P36)

- **Node Finder** の **File:** を **Design Entry(all names)」**に設定して **「List」**をクリック (使い方 P23 上図)
- シミュレーション例にある信号名を選択し「>」ボタンをクリックして登録 (使い方 P23 下図)
- **Grid Size: 10ns、End Time: 400ns**
- 多ビット信号 (DATA, REC_DATA, DATA_BUF, SET_DATA) は、マウスで信号名を選択した後、**Propaties** で **Radix:** を **Hexadecimal** に変更する (使い方 P25)

【9】正しい論理シミュレーション結果が得られたら、そのシミュレーション結果を実験担当の方に確認してもらい、ＯＫが出たら設計した回路のソースをシミュレーション結果をプリントアウトして下さい (これらはレポートに添付)

.....

3.6.2. ＜ Part 2 ＞ 〜 設計回路のＦＰＧＡへの実装・動作検証 〜

※ ここでは、Part 1 で設計した受信回路~~・送信回路~~を用いて DE0 ボードで動作させるための回路を生成し、2 台の DE0 ボードの各々に受信回路、送信回路を実装し平ケーブルにより接続の後、データ転送を検証します。

(必ず始めに作った作業フォルダで設計を行って下さい)

【1】Quartus II の起動

【2】受信~~、送信~~ベース回路のコンパイル

- ベース回路は、前に設計した受信回路~~・送信回路~~と、ＬＥＤ表示回路 (送受信データを表示するための回路) を組み合わせるための回路です (ファイル名 **input_base、~~output_base~~**)。ＦＰＧＡには、この回路が実装されます。
- プロジェクト名の設定 (「Quartus II の使い方」 P5〜P10)
  - (使い方 P6)：プロジェクト名は，受信回路では **input_base**、~~送信回路では **output_base**~~ として下さい。
  - (使い方 P7)：画面右上にある「Add」をクリックして，**input_core**~~または **output_core**~~ を登録します。そして「Next」をクリック。
  - (使い方 P8)：ウインドウ上の Family: は **Cyclone III** を選択し、その後下の Available devices: で **EP3C16F484C6** を選択して「Next」をクリック。
  - (使い方 P9)：そのまま「Next」をクリック。
- 合成 (コンパイル)
- **入出力ピンの配置** (「Quartus II の使い方」 P16〜P18)
  - 設計ファイルの各入出力は特定のピンに割り当てる必要がありますので、入出力と割り当てるピン番号の対応は別紙を参照して下さい。また、必ずそれに従った位置に配置して下さい。
- 回路の再コンパイル

【3】**設計した回路の DE0 ボードへの実装** (「Quartus II の使い方」P28-P31)

【4】動作検証

- 動作検証前に、各々のボードの使用する DE0 ボードの **GPIO 0** コネクタ間を平ケーブルで接続されていることを確認して下さい。
  (1) 受信回路、送信回路双方をリセットスイッチである SW0(ボード右下にある 3 つの押しボタンスイッチの右側) を押してリセットする。この際、受信側の 7 セグメント LED には
  ’ＦＦ’と表示され、送信側は全て消える。
  (2) 次に、送信回路側でボード左したの 10 個のスライドスイッチのうち SW0〜SW7 により 4 ビット 2 桁の値を設定する。
  (3) そして、受信側の送信要求スイッチ SW1(3 つの押しボタンスイッチの真ん中) を押して ON。
  (4) 送信側の 7 セグメント LED の値が受信側に表示され送信側の LED が消えたら送信完了 (何種類かの値で確認して下さい。)

3.7. レポートついて

提出レポートには、結果として以下を添付すること.

- 設計した受信回路~~、送信回路~~のソース (input_core.vhd~~, output_core.vhd~~)
- 同受信回路~~、送信回路~~のシミュレーション結果

また、双方の回路についてのシミュレーション結果と設計時に使用した状態遷移図との対応についてを結果として簡単にまとめること.

4. 検討課題

【1】本実験で設計したデータ伝送方式は非同期伝送方式であるが，送信回路・受信回路各々は，内部クロックに同期して処理を行っている．同じ性能の DE0 ボードを用いているので双方とも同じ周波数の内部クロックで動作しているが,

○ もし、受信回路と送信回路とでそれぞれ動作クロック周波数が大きく異なっていた場合のデータ伝送はどうなるのか？

について考察せよ.

【2】回線形態には,「タイム・シェアリング方式」「リアルタイム方式」「リモート・バッチ方式」とあるが，各々について説明せよ.

【3】データ伝送におけるフロー制御について説明せよ.

【4】誤り検出の一つである「巡回符号方式 (CRC 方式)」について説明せよ.

参考文献

[1]“ トランジスタ技術 SPECIAL No.8 ”，ＣＱ出版社

[2]“ ＶＨＬＤによるハードウェア設計入門 ”，長谷川裕恭，ＣＱ出版社

[3]“ A VHDL Primer ＶＨＤＬ言語入門 ”，Jayaram Bhasker，ＣＱ出版社