

デジタル回路の基礎知識

- OPEN-IT FPGA トレーニングコース受講前の準備として -

第 1.4 版

2014 年 9 月 1 日

高エネルギー加速器研究機構

素粒子原子核研究所

エレクトロニクスシステムグループ

内田 智久

はじめに

Open-It FPGA トレーニングコースを5年以上開催していますが、最近トレーニングコースで行う演習時間を長くして欲しいと頼まれる事が多くなりました。トレーニングコースの開催期間を2日間から3日間へ変更する事も検討したのですが、周りの方々に相談したところ3日間以上になると参加できなくなる人も増える事が分かりました。そこで、トレーニングコースでの説明内容からデジタル回路に関する基礎知識の部分を受講者の皆さんに予習していただく事で演習時間を延ばす方法を採用する事にしました。これまでも受講者の方に事前に予習しておく事を教えてほしいと聞かれた事も少なくありません。これは、初めて回路について学ぶので不安を感じているのだと筆者は想像しています。この様な不安もこの文書で払拭されると期待しています。受講者の皆さんには負荷をかけてしまいますがご協力お願いいたします。

この文章には Open-It FPGA トレーニングコースを受講する前に知っていて欲しい事をまとめてあります。この文章のみで完全に理解することは難しいかもしれません。ここで説明する内容はトレーニングコースで簡単に復習しますので完全に理解する必要はありませんが理解できなかった点や疑問点などを書き出してから受講してください。そして、トレーニングコース受講時に講師にその疑問点について質問して解決してください。

これから新しい事を学ぶときに大切なことは学ぶ対象全体に共通する原理原則を概念的、感覚的に理解することだと筆者は考えます。共通する原理原則を理解する事で理解が要素毎から要素間に拡張されるからです。この文章の目的はデジタル回路全体に共通する概念を伝えることです。筆者は概念を伝えるためには定性的な説明の方が適していると考えていますので、この文章での説明は全て定性的です。悪く言えば厳密な説明はありません。この文章の内容に物足りなさを感じる方も居ると思います。その様な方にとってこの文書が詳細な書籍を読むきっかけになれば筆者としては非常に嬉しく思います。

本文章に書かれている各章は独立です。しかし、前に書かれている章の知識を後ろの章ではすでに知っているとして書いていますので、先頭から順番に読み既に知っている節は読み飛ばす方法をお勧めします。

皆さんとトレーニングコースでお会いできる事を楽しみにしています。

2014年7月12日 筆者

はじめに _____	2
1. Field Programmable Gate array _____	4
2. デジタル技術を使用する理由 _____	5
3. デジタル技術での数の表現 _____	10
2進数.....	10
16進表現.....	10
4. デバイス間のデジタル信号規格 _____	11
5. デジタル回路の基本 4 要素 _____	13
NOT ゲート.....	13
AND ゲート.....	13
OR ゲート.....	14
記憶素子.....	15
6. 回路の階層化 _____	16

1. FIELD PROGRAMMABLE GATE ARRAY

多くの皆さんは Field Programmable Gate Array (FPGA)について既に知っていると思いますが簡単に説明します。

FPGA は集積回路実装術の一つであり、私たちユーザーが設計した電気回路を集積する一つの方法です。

FPGA を一言で表現するなら、ユーザーが回路情報を書き換えることができる半導体集積回路です。目的に応じて幾つかの種類があるのですが、多くの FPGA は何度でも書き換えることができます。しかし、変更可能というのはちょっと不思議です。何故なら集積回路はシリコン基板上にトランジスタ、抵抗、コンデンサなどを形成して実現しますので製造後の回路変更ができないのが自然です。ところが、FPGA は何度でも回路変更が可能な仕組みを持っています。書き換えることができる理由は FPGA に実装する事ができる回路がデジタル回路に限定されていることが強く関係しています。書き換えることができる仕組みはトレーニングコースで解説します。

ポイント：

- FPGA はユーザーが回路を変更する事ができる集積回路実装技術である
- FPGA に実装可能な電気回路はデジタル回路のみである

2. デジタル技術を使用する理由

FPGA 上にデジタル回路は実装する事ができますが、アナログ回路を実装する事はできません。FPGA に実装できるデジタル回路、もう少し広い意味でのデジタル技術は私たちの日常生活の中でも広く使用されています。以前アナログ技術で実現されていた装置が年々デジタル化されています。例えば録音された音楽を聴くとき、以前はアナログ技術を使用したレコード盤を使用していました。しかし、今ではデジタル技術を使用した CD やデジタルプレーヤを使います。テレビ放送もアナログ技術を使用していましたがデジタル化されました。身の回りにある電気製品の多くがデジタル化され続けています。

この様に近年多用されているデジタル回路を FPGA 上に実装するのですが、デジタル回路について学ぶ前に何故デジタル回路を使用しなければいけないのかを確認します。何でもデジタル化すれば良いわけではなく、もちろん何でもアナログ化すれば良いわけではありません。どちらも長所と短所を持っています。より質の高い実験データを得るためには両技術を適切な場所を使用して計測装置を開発する必要があります。その為にデジタル回路を使用する理由、すなわち長所を確認します。

デジタル回路の長所を考えるために、音楽を聴くときに使用するレコード盤と CD を比較します。

まずはレコード盤を用いて音楽を聴く装置、レコード・プレーヤー装置について考えます。図 2-1 を見てください。

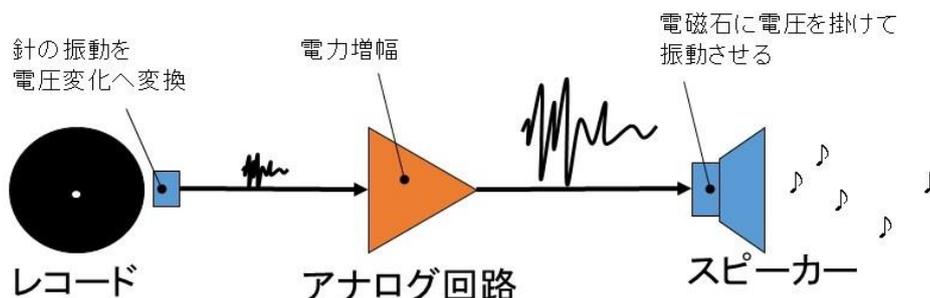


図 2-1 レコード盤再生装置

これはレコード・プレーヤー装置を概念的に書いた図です。音楽はアナログ信号であり、音楽を聴く人間も音であるアナログ信号を受け取ります。音楽情報はレコードの溝として記録されています。その溝を針でなぞる事で針が振動します。針には磁石が付いていて、コイルの中で振動するようになっています。針の振動が磁石を動かし、その磁石がコイルに誘起する電圧を取り出します。小さな振動で発生させた電気信号なので電圧振幅は小さく電力も小さいのでアナログ回路で増幅します。大きな信号へ増幅しスピーカーを駆動することで空気を振動させ音楽を再生します。スピーカーは振動板、コイル、磁石で構成されています。磁石の周りに巻かれたコイルに電流を流すことでコイルが振動します。コイルには振動板が接続されているので音が再生されます。

この装置はとても分かりやすいと思いませんか？

アナログ信号をアナログ記録し、アナログ信号を読み出し、その信号をアナログ回路で増幅して再生する。しかし、今ではデジタル化されています。次はデジタル技術を使用している CD について考えます。

図 2-2 は CD 装置を概念的に表現した図です。

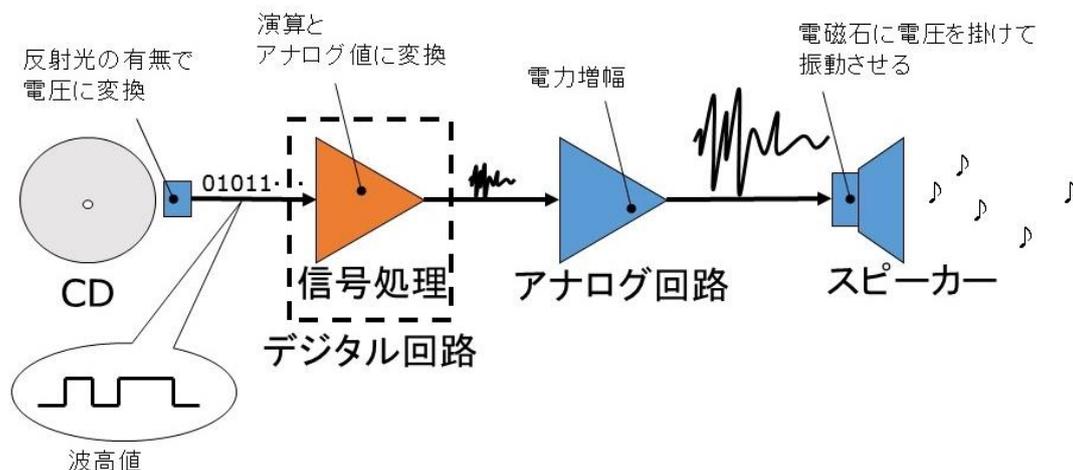


図 2-2 CD 再生装置

先ほどのレコード盤と比較すると図中に信号処理と書いたデジタル回路が増えています。この事に注意してください。CD の場合は記録されている音楽情報が既にデジタル化されています。CD に光を当て反射光の有無で信号を再生します。図中の波高値と書かれている部分の波形が再生波形の例です。横が時間軸で縦が光の反射量です。光量を電圧へ変換して電気信号にします。すると時々刻々と変化する矩形波が出力されます。この信号をデジタル信号処理回路で信号処理を行い、その後、デジタル値をアナログ値へ変換する回路 Digital-to-analog Converter (DAC)を用いてアナログ信号へ変換します。アナログ信号へ変換された後は先ほどの例であるレコード盤と同じです。電力増幅してスピーカーを鳴らします。

ここで、皆さん、不思議に思いませんか？

デジタル化された CD は新たに回路が増えているのです。一見、物事を複雑にしただけのように見えます。しかし、デジタル技術の長所により CD の方が総合的な性能が良くなります。何故でしょうか？その理由がデジタル技術を採用する理由です。

デジタル回路（技術）の長所とは何でしょうか？

次を読む前に少し考えてください。

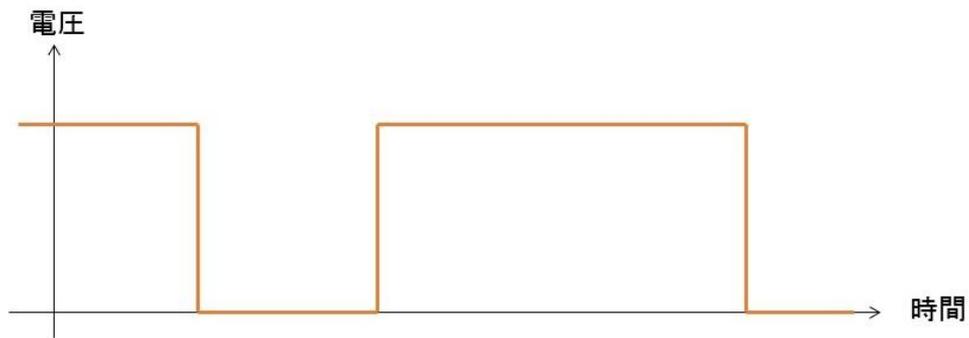


図 2-3 デジタル信号の例

長所を説明する前にデジタル信号について復習しておきます。デジタル信号とは例えば図 2-3 のような信号です。電圧が時間変化しますが電圧が高い状態と低い状態の二つの状態間を行き来する信号です。一對の信号線でたった 2 状態しか伝えることができません。

注意：ここでは電圧を状態値として使用しましたが電流を用いることもあります。

このような信号を採用する長所はなんでしょうか？

それはノイズなどの外乱に強いことです。信号伝搬の実験を経験したことがある方は分かると思いますが信号波形を変化させずに伝送する事はとても難しいです。先ほどのデジタル波形も電線を用いて伝送すると図 2-4 のような波形になるかもしれません。



図 2-4 実際のデジタル信号の例

図 2-4 はかなり大きさに書きましたがこのような波形の乱れを観測する事ができます。この様に波形が乱れてしまってもデジタル信号なら伝え誤ることはありません。それは図 2-5 に示すように電圧の高低を判断する電圧が一点ではなく範囲を持っているからです。

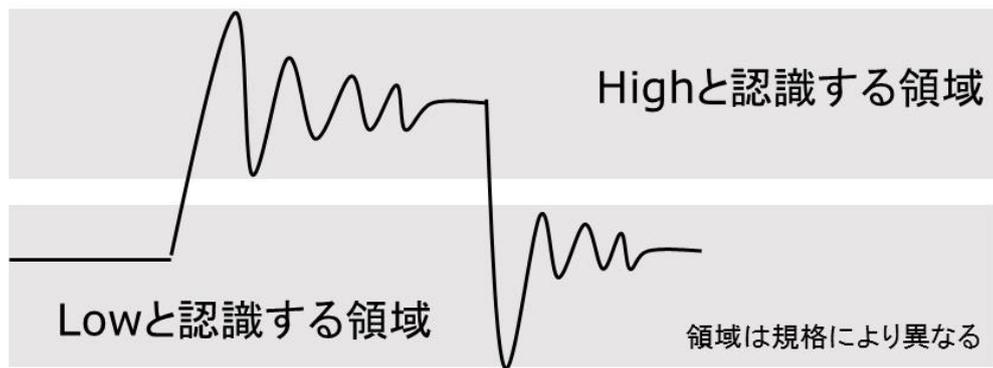


図 2-5 電圧の高低を判定する電圧範囲

仮にこの信号がデジタルではなくアナログ信号で矩形波を伝搬させようとしていたと考えると元の波形と全く異なる値を受け取ることになりますので誤った信号処理を行ってしまいます。ところがデジタル信号では間違えなく伝えることができるのです。

デジタル信号は情報量を二つに下げ、一度に伝送する事ができる情報量を犠牲にすることでノイズ耐性を高めていると考えることができます。アナログ信号は信号の全ての点の電圧が信号情報なので情報量は非常に多い、点の数と考えると無限大！

デジタル回路はアナログ信号と異なり物理量（連続値）を伝送するのではなく数値を伝送している点も注意してください。アナログ信号は電圧や電流などの物理量（連続量）を使用します。その物理量が信号情報となります。全ての点に意味があるため雑音と信号の識別はできません。デジタル信号も電圧や電流などの物理量を信号伝達に使用しますが、その物理量の値を使用するのではなく範囲を使用して0と1の値である離散値として情報を伝達します。アナログ信号処理の場合は電流や電圧を回路でそのまま処理しますが、デジタル回路ではもはや電圧や電流などの物理量（連続量）の概念はなく数値（整数値）として扱い処理します。

デジタル技術の長所をまとめます。最大の長所はノイズに強いことです。ノイズに強いので信号をデジタル化する事で間違って処理することがほとんどなくなります。検出器から出力される信号は微弱なアナログ信号です。そのまま伝送や処理を行うとノイズの影響を受けて本来の波形と異なってしまいます。そこで、計測システムでは検出器信号の劣化を最低限に抑えるために検出器が出力する信号をアナログ回路で必要最低限処理し、できるだけ早くデジタル信号へ変換します。一度デジタル化してしまえばノイズが強くなるので信号の扱いが簡単になります。例えば光ファイバを用いてデータを長距離伝送することも簡単に行うことができます。デジタル化の長所は様々な事がありますが、様々な機能を実現可能にしている理由の根源はノイズに強いことです。

デジタル信号は数値ですからデジタル回路は2進演算を行う回路と考えることができます。数値演算を間違えることなく実行できるので数学的なテクニックを用いてアナログ技術では実現が難しい回路を作ることができます。例えば、先ほどの例で登場したCDで用いられているエラー訂正符号技術があります。CDの表面を見ると表面が汚れていたり、傷がついていることが少なくありません。そのような汚れや傷でデータ読み出しが間違えることがないのでしょうか？実は間違えているのです。デジタルでも全てのノイズや外乱の影響を完全になくすことはできません。CDの読み出しが間違っても正しく読み書きできる、これはとても不思議です。コンピュータはデータが少しでも間違えれば文字化けや暴走してしまうはずですが、何故なら全く異なる値に

なってしまうのですから。実はエラーを訂正する仕掛けが組み込まれています。CD では間違いが起こることを想定して本来のデータ以外にエラー訂正符号と呼ばれる特別なデータが書き込まれています。この特別なデータを用いてデータを計算するとエラーが起きている個所を特定する事ができます。デジタルが取る状態は 2 つですので間違っている場所が分かれば値を他方へ変換すれば正しい情報となります。例えば間違っている位置の値が 0 であれば 1 へ変換することで正しいデータとなります。この様な処理をアナログ回路で行うことは現実的実現する事は容易ではありません。

この様な長所があるデジタル技術ですが短所もあります。デジタル処理の概念はとても古くトランジスタが発明される以前からありました。しかし、デジタル処理の短所により採用が見送られてきたのです。

その短所とは回路規模が大きくなることです。

デジタル信号は 1 対の信号で送ることができる情報が 2 状態のみです。例えば波形を送りたいと思ったら多くの信号線（例えば 8 対）が必要です。信号の両端には必ずトランジスタを使用した回路が必要です。アナログ信号であれば 1 対で良い回路が複数対の回路が必要になるのです。これはアナログ回路と比較すると回路規模（例えば使用トランジスタの数）がとても大きくなります。この様な理由により集積化が難しかった以前はデジタル回路を採用する事ができませんでした。しかし、近年の微細化加工技術の発展により大規模回路を数 mm 四方の大きさに作る込むことができるようになったので問題ではなくなりました。短所が解決された事でデジタル回路が多数採用され現在のようにデジタル化が進みました。

ポイント：デジタル技術はノイズ耐性が高い

3. デジタル技術での数の表現

2 進数

デジタル技術では 2 状態の信号をやり取りするので処理は 2 進数の演算が中心になります。頻繁に 2 進数表現を使用するので復習しておきましょう。

物や要素の個数などの数自体は普遍な概念ですが、その表現方法は多数あります。日常では個数などを数えるために 10 進数表現を使用します。

数 Z の N 進数表現を式で表現すると、数 Z を下式のように表した時の a_i を i が小さい方から順番に右から左へ並べる表現方法のことです。

$$Z = \sum_{i=0}^m a_i \times N^i, a_i \in \{0, \dots, N-1\}$$

2 進数表現とは a_i の母集合が 0 と 1 で構成される場合です。例えば 10 進数の 6 を 2 進数で表現すると 110 となります。

2 進数表現の 1 桁の事をデジタル技術ではビット (bit) と呼びます。また式の m の事をビット数と呼びます。 m 桁の 2 進数は 0 から 2^m-1 まで表現できることを確認してください。

2 進数で表現した時、 $i=m$ の桁を最上位ビット (Most Significant Bit, MSB)、 $i=0$ の桁を最下位ビット (Least Significant Bit, LSB) と呼びます。2 進、16 進表現した場合、通常は右端が LSB、左端が MSB となります。16 進表現については次の節で説明します。

16 進表現

2 進数のビット数が増えるととても見難くなります。例えば、最近のコンピュータは 64 ビット単位の数値を使って動作しますが、64 桁の 2 進数は数字が分からないばかりか、間違えなくメモを取ることも慎重になります。

そこで少しでも見やすくするために 16 進表現が導入されました。2 進数を 4 ビット毎にまとめて分け、4 ビットを一つの桁として表現する方法です。この方法を採用するために 16 個の要素を新しく導入します。通常私たちが使用する数の要素は 10 個です。一桁で表現できる数字は 0 から 9 までの 10 個です。これでは足りません。そこで、アルファベットを使って要素を 16 個へ増やします。0 から 9 までの数字に加えて A, B, C, D, E, F の 6 個のアルファベットを加えます。アルファベットは A からそれぞれ十進数の 10, 11, 12, 13, 14, 15 と対応させます。例えば、2 進数で表現した場合の 10110010 は 16 進表現では 0xB2 となります、B2 の前の 0x は数が 16 進表現であることを示すために先頭に付ける識別記号です。

この方法で幾分か読みやすくなりましたが、見て直ぐに数字へ変換できる人は少ないです。しかし、この方法は頻繁に使用されます。その理由はデジタル回路で値を数値として使用することより符号として使用する機会の方が多いからです。符号として使用する場合、2 進数のビット列パターンが重要となります。現時点では納得できないかもしれませんが、ソフトウェアや回路設計を体験すると実感できると思います。

4. デバイス間のデジタル信号規格

皆さんが計測システムや計測装置を開発する場合、一つのチップで全ての処理を行うことはありません。目的の機能を実現するために複数のチップや装置を使用します。例えば、検出器やセンサー信号をデジタル化したデータを FPGA で処理するために、センサー信号をデジタル化するチップ(IC)と FPGA を接続してデジタル化したデータを FPGA へ転送しなければいけません。

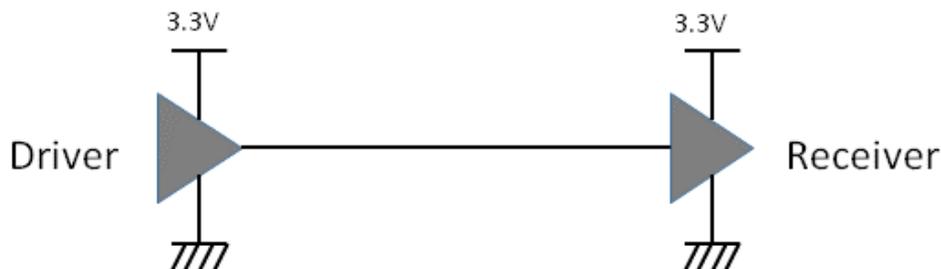


図 4 - 1 チップ間通信

例として図 4 - 1 のような状況を考えます。この図は異なるチップに搭載された信号を駆動する Driver と信号を受信する Receiver 回路が配線で接続されている状況を表しています。この状況のようにデジタル信号を複数のチップ間でやり取りする場合、その信号について規則を設ける必要があります。デジタル信号は 2 つの状態を伝送する事を説明しましたが、この 2 つの状態を区別するために同じ条件を使用する必要があります。この条件を使用する事で次のような問題を解決する事ができます。

- High level (H レベル) と認識するための条件は？
- Low level (L レベル) と認識するための条件は？

電圧変化により状態を伝える場合、何ボルト以上から H レベルであり、何ボルト以下から L レベルなのかを予め決める必要があります。

識別条件を独自に決めると他者が開発した異なる基準を採用したチップと接続できなくなります。そこで、世界的に統一規格が定められています。この規格を使用する事でお互いを接続して通信する事が出来ます。しかし、この規格は用途毎に細分化されて定められているので種類がとても多いです。FPGA は様々な規格に対応する事が出来ますので FPGA に接続しているチップの規格に FPGA 側を合わせて使用してください。

最後に規格例として私たちが頻繁に使用する LVCMOS33 と呼ばれる規格について簡単に説明します。この規格では図 4 - 1 の様に一つの信号線の両端に送受信回路が接続されます。各チップは基準電位となる共通のグランド線と電圧 3.3V の電源が接続されています。受信側で 2 つの状態を区別するための条件は図 4 - 2 の様に定められています。

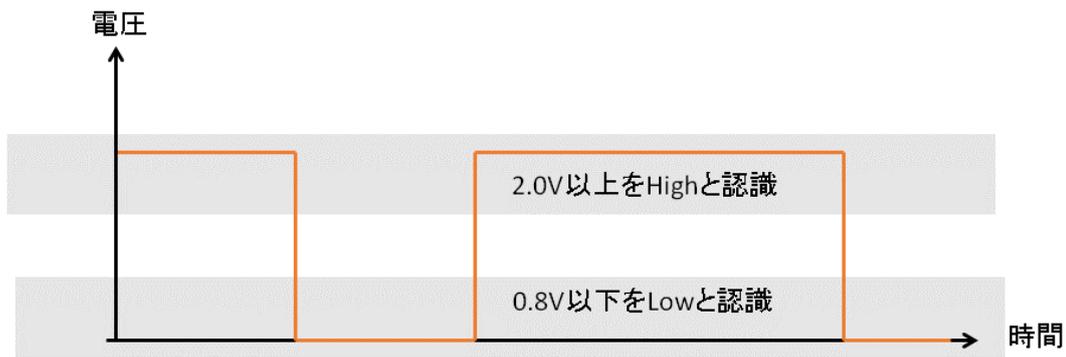


図4-2 LVCMOS33 規格

図中のオレンジ色の線が信号変化を表しています。この規格では 2V 以上を電圧が高い状態と認識し 0.8V 以下を電圧が低い状態と認識します。注目してもらいたいのは 2つの状態間にどちらの状態でもない隙間がある事です。この隙間は禁止領域と呼ばれ、この隙間に信号がとどまる事は許されていません。もちろん変化時にこの領域を通過する事は許されています。モノづくりの観点ではこの隙間は重要です。この様な隙間を設けることができる事がノイズに強いデジタルらしい工夫です。

この隙間にどのような意味があるのでしょうか？

この隙間は送受信回路の特性ばらつきを吸収するために設けられています。デジタル回路はトランジスタにより構成されています。この回路はある電圧より高い場合は High レベルと認識し低い場合は Low レベルと認識します。この判断電圧は判定点なので High レベルでもなく Low レベルでもない状態になります。その様な信号を回路が受信すると中間電位を出力してしまいデジタル値を正しく伝達できなくなってしまいますので禁止しなければいけません。さらに、この判定電圧はトランジスタの製造ばらつきや電源電圧値によりある範囲でばらつきます。回路を工夫してばらつき範囲を小さくする事は出来ませんがゼロとすることはできません。

そこで、ある範囲を禁止領域と決める事で特性ばらつきを吸収しお互いの通信を保証しているのです。

この禁止領域を決める理由がもう一つありますが、トランジスタのアナログ動作に起因する理由なので理解するためにはトランジスタの動作を学ぶ必要があります。興味ある方は集積回路の教科書を参照してください。

ポイント:お互いに通信するために信号規格に従い状態 0 と状態 1 を識別する共通基準を使う必要がある。

5. デジタル回路の基本 4 要素

コンピュータに代表されるデジタル回路は次の 4 つの基本要素から構成されています。

- NOT ゲート (インバータ)
- AND ゲート
- OR ゲート
- 記憶素子

どんなに複雑な動きをする回路でもこの 4 つの要素へ分解する事が出来ます。各要素の動作を確認しましょう。

NOT ゲート

NOT ゲートはデジタル回路の基本形です。別名インバータとも呼ばれます。図 5 - 1 に回路図記号と真理値表を示します。

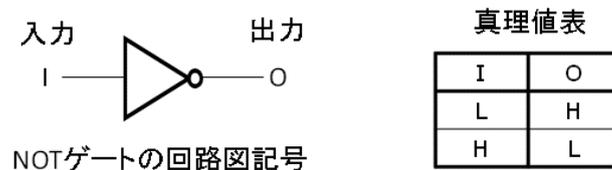


図 5 - 1 NOT ゲートの回路図記号と真理値表

真理値表とは入出力の論理関係を表した表です。NOT ゲートの場合、入力の状態と異なる状態が出力に現れる事を意味しています。H レベルを入力すれば L レベルが出力され、L レベルを入力すれば H レベルが出力されます。このような動作を (論理) 反転動作と呼びます。会話時は「信号を反転する」などと言います。

回路図記号は三角形を使用します。入力または出力線の隅に丸印をつけます。この丸印が信号の論理反転を表します。

NOT ゲートは論理を反転させる事が出来るので注目する論理を変更したいときに使用します。例えば、ある機能が動作しているときに H レベルになる信号 A がある時にある回路で信号 A を使用する事を想定します。動作が止まっている時、すなわち信号 A が L の時に何らかの動作を行う回路を設計する場合、信号 A に NOT ゲートを挿入する事で動作が止まっている時に H レベルになる信号を作ることができます。この様に注目する状態を反転する機能と後述する AND ゲートや OR ゲートと組み合わせて使用する事で様々な条件を検出する回路を設計する事ができます。

ポイント：NOT ゲートは注目する状態を入れ替える事ができる

AND ゲート

図 5 - 2 に AND ゲートの回路図記号と真理値表を示します。

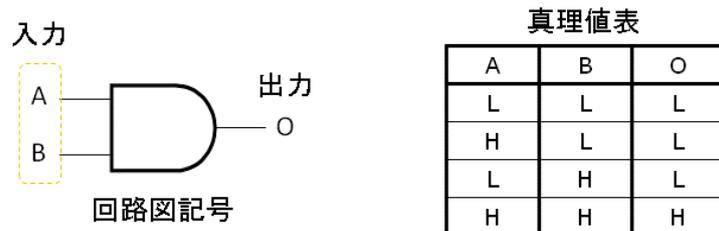


図 5 - 2 AND ゲートの回路図記号と真理値表

AND ゲートは全ての入力が H レベルの時のみ H レベルを出力します。他の場合は L レベルを出力します。これは入力の掛け算と考える事もできます。ここでは入力が 2 つですが入力信号数が 3 以上の AND ゲートもあります。入力が増えても動作は同じです。全ての入力が H レベルの時のみ H レベルを出力します。

補足：3 入力以上の AND ゲートは 2 入力 AND を複数接続する事でも実現できます。

実際の回路では AND ゲートの入力に NOT ゲートを適当に挿入して特定入力パターンを検出するために使用します。例えば、2 進数の 0 を L レベル、1 を H レベルと考えて、2 桁の 2 進数信号を 2 入力 AND ゲートに入力すると入力が 3（10 進数表現）の時のみ出力が H レベルになります。この回路は値 3 を検出する回路と考える事が出来ます。入力に NOT ゲートを適当に挿入する事で他の値を検出する回路を設計する事が出来ます。

ポイント：AND ゲートは条件検出時に使用する

OR ゲート

図 5 - 3 に OR ゲートの回路図記号と真理値表を示します。

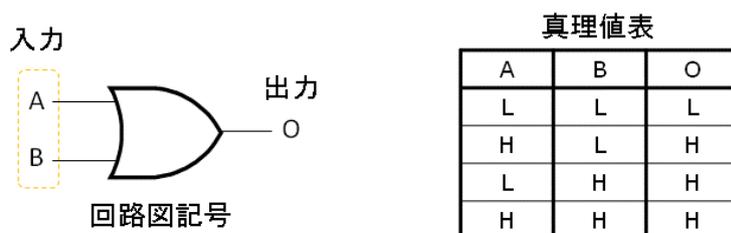


図 5 - 3 OR ゲートの回路図記号と真理値表

OR ゲートはいずれか一つの入力が H レベルの時に H レベルを出力します。OR ゲートも入力信号数が 3 以上の OR ゲートを考える事が出来ます。いずれか一つの入力が H レベルの時に H レベルを出力します。

補足：AND ゲート同様に複数の 2 入力 OR ゲートを複数接続しても作ることができます。

実際の回路では前述の AND ゲートと NOT ゲートを用いて作られる条件検出回路と組み合わせで使用します。例えば、2 進数 2 桁の信号の値 3 または値 1 を検出する回路などを作る事が出来ます。AND ゲート節で説明した値 3 を検出する回路と新たに値 1 を検出する回路を AND ゲー

トと NOT ゲートを用いて設計し OR ゲートの入力に接続すると値 3 または値 1 を検出する回路となります。

ポイント：OR ゲートは条件を足し算する時に使用する

記憶素子

デジタル回路内で使用される記憶素子の種類は複数あります。ここでは回路設計で重要な役割を担う D タイプ・フリップフロップ (DFF) について説明します。

図 5-4 に DFF の回路図記号と動作説明のためのタイムチャートを示します。

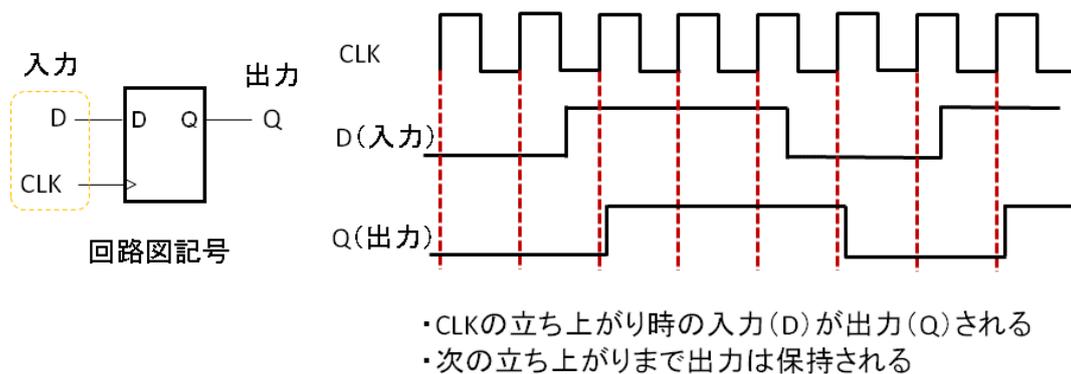


図 5-4 DFF の回路図記号とタイムチャート

前述した他ゲートと比較すると動作が複雑です。DFF の入力は 2 つあります。D 端子と CLK (クロック) 入力です。出力は Q 端子です。タイムチャートは時間変化による信号変化を表した図で横軸が時間で左から右へ流れています。縦軸は各信号の電圧変化を表しています。

DFF の動作を説明します。CLK 端子に入力している信号の L レベルから H レベルへの立ち上がり遷移時に D 端子入力の状態を取り込み (記憶して) Q 端子へ出力します。CLK 端子に入力している信号の立ち上がり時のみ動作するので、CLK 信号の立ち上がり時以外に D 端子の信号が変化しても出力 Q 端子の状態を記憶しているので Q 端子の状態は変化しません。

この DFF と前節まで説明したゲートを使用する事で全てのデジタル回路を作る事が出来ます。

現時点では 4 つの要素と CPU などの複雑な回路の関係を想像する事は難しいかもしれませんが、学習を進めるに従って様々な事がつながると思います。

ポイント:CLK 端子に入力している信号の立ち上がり遷移時に D 端子入力の状態を取り込み (記憶し) Q 端子へ出力する

6. 回路の階層化

デジタル回路は数多くの基本 4 要素から構成されています。それぞれの要素は他の要素と配線により接続されています。それら相互の接続関係を表すために回路図やハードウェア技術言語 (Hardware Description Language, HDL) を使用します。

実際の回路規模はとて大きくなるので一枚の回路図や一つの HDL ファイルに記述すると読み難いものになってしまいます。そこで、あるまとまった機能毎に回路を分け、分けた回路毎に一つの箱を割り当てて回路図や HDL を整理すると読みやすくなります、動作も理解しやすくなります。例えば、図 6-1 のような構成の読み出し回路設計を想定します。その回路は機能毎に大きく 3 つの箱、ASIC I/F、データバッファ、PC I/F へ分けることができます。

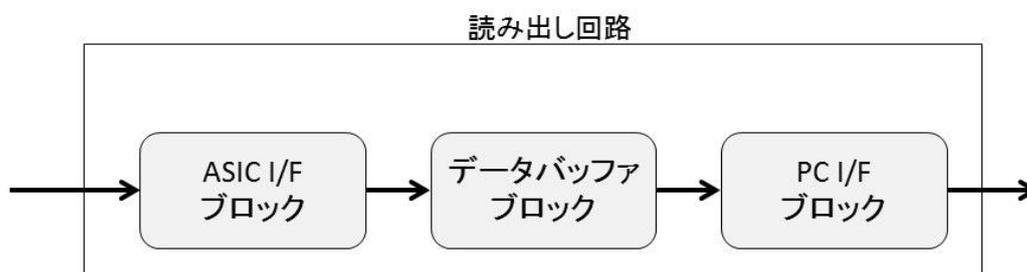


図 6-1 ブロック図の例

この様にある機能を一つの箱として表現した図をブロック図、箱をブロックと呼びます。各箱の中身は数多くの基本要素で構成されていますが、ブロック図で表現すると煩雑な部分を隠蔽する事ができるので理解しやすくなります。言い換えると抽象化により動作を理解しやすくなります。各箱の中をさらに複数の箱へ分解することもできます。この様に階層的に表すことを階層化と言います。

回路図や HDL コード内でも同じように階層化する事で理解しやすくすることができます。回路図や HDL を書くときは機能毎に一つのブロックにまとめて階層化してください。私の経験では読みやすいブロック図を書けることと回路動作の理解度は関係あるように感じます。正しく理解している人はとても読みやすいブロック図を書くことが多いと思うのです。

補足：Verilog-HDL ではブロックの事をモジュールと呼びます。