ロジックICで創るCPUキット

TTM8 解説本

著　電子ライダー
表紙 成瀬歩実

これからCPUを学ぶあなたのために

みやこ電子工房
MIYAKO DENSHI KOBO

ロジック IC で創る CPU キット TTM8 解説本

— これから CPU を学ぶあなたのために —

［著］電子ライダー

技術書典 12 新刊
2022 年 1 月 22 日　 ver 1.0

■免責

本書は情報の提供のみを目的としています。
本書の内容を実行・適用・運用したことで何が起きようとも、それは実行・適用・運用した人自身の責任であり、著者や関係者はいかなる責任も負いません。

■商標

本書に登場するシステム名や製品名は、関係各社の商標または登録商標です。
また本書では、™、Ⓡ、Ⓒ などのマークは省略しています。

まえがき

初めまして、電子ライダーと申します。あなたがこの文章をよんでいるということは自作 CPU に興味があるということですね。うれしい限りです。あなたは自作 CPU というニッチなテーマに何故興味を抱いたのでしょうか。CPU というのは電子工作を生業としない方でも何たるかを知っていると思います。自作 PC をされる方などはこの CPU はどうだとかあの CPU はどうだとか深い知識をお持ちなのではないでしょうか。ところで自作 PC でも電子工作でも CPU(あるいはマイコン) というものは誰かが用意してくれているもを買ってきて使うものです。何か良くはわからないけれど計算をしてくれる石ということで便利に使うことができますね。この CPU は使うものだという概念を取っ払った人のみが自作 CPU という境地にたどり着けるのだと思います。あなたはこういう風に考えたのではないでしょうか「CPU はどういう仕組みで動いているのだろうか。」と。少なくとも私はそうでした。例によって私もマイコンを使った電子工作を長年 (とはいっても電子工作を始めてから現在までに 10 年ぐらいですが・・) やってきました。学生時代のロボット製作なんかは私の人生を振り返っても特に素晴らしい思い出の一つです。ですがそんな素晴らしい活動のなかでももやもやがあったのです。作製した回路ではマイコン内がブラックボックスであったという事実です。便利なツールというもは各段にハードルを下げることにつながります。最近では特にその方針が強い製品がでていると感じます。それを否定するつもりは毛頭ありませんしむしろ良い傾向だと思っています。間口を広げることで電子工作沼に引き込みやすくなりますしね。しかしながら便利ツールというものは同時にブラックボックスを多くしてしまいがちです。マイコンへのプログラムがレジスタの書き換えを行っているということを知っている方が少なくなっているのではないでしょうか。そんなブラックボックスへと興味を持ってくれた方へこの本が役立てば良いと思っています。

この本の目的について

この本は私、電子ライダーが開発したオリジナル CPU、TTM8 について解説する解説本です。TTM8 はロジック IC を組み合わせて作製された CPU で FPGA 上で開発する時とはまた異なる工夫ポイントがあります。そういったポイントをこの本でお伝えできれば良いですね。最も読んでいただきたい方は TTM8 を買っていただいた方です。基本的に組立キットとしての販売している TTM8 ですが、組み立てたあとにも価値を提供したいと考えています。この本はその価値の一つです。内部構造を学んでいただくことで自作 CPU というテーマに目を向けて頂けると嬉しいですね。実は TTM8 は I/O を備えていません。I/O を得るためには拡張基板を接続する必要があるのですね。この拡張基板ですがユーザー自身に作っていただきたいと思っています。いきなり自作 CPU は難しくても拡張基板であれば作ってい頂けるのではないか？ という考えの元です。もちろん TTM8 をまだ手にしていない方にも読んでいただける内容となっています。聞けば情報科の大学では FPGA 上で CPU 作製をする授業があるみたいです。(筆者は電気科です。) そんなこれか

目次

1.1　電子工作

電子工作楽しいですよね。この本を読んでいるあなたはきっと電子工作を日頃から嗜むお方であると存じ上げます。ところで、あなたの電子工作のテーマは何ですか。一口に電子工作と言っても数多の種類がありますよね。アナログ回路、デジタル回路、あるいは重きをメカ部分やソフト部分においた作品なのかもしれませんね。私の場合それが自作 CPU だったというわけです。ところで自作 CPU は電子工作なのでしょうか。一般的には FPGA 上で構成される自作 CPU は電気ではなく情報分野ではないのかと思われるかもしれません。話は変わりますが、近年では機械を動かすのに必ずと言っていいほどマイクロプロセッサ (MPU) が使われます。MPU というものはソフト次第でなんでもできるのがよいところで、機械に合わせて専用回路を構成するまでもなくソフトを変えるだけで電気回路自体は汎用的に使用できるからです。MPU が発明される前は機械に合わせた専用回路が開発されていました。昔の機械の中にある回路はワンチップでプロセッサが載っているのではなく回路全体でコンピュータでした。さて、この本で最初の節であるこの場で何が言いたいかといいますと、みなさんに昔と同じような専用回路を創る電子工作を思い出していただきたかったのです。これは紛れもなく電気屋さんの領域ですよ。

1.2　ブラックボックスへの追及

便利なものは大抵の場合中身が見えません。今や大抵の人が持っているスマホがどうやって動いているか。車がどうやって走っているのか。知っていますか？ 私はよく知りません。これはハードウェアエンジニアにとっても同じで便利に開発を進めようとするとブラックボックスを使わざるを得ない事があります。マイコンを使う事はできるけれどマイコンを創れる人は少ないでしょう。これはつまりマイコン内部はブラックボックスだということです。自作 CPU というテーマはこのブラックボックスへの追及です。どうやって動いているか不明な存在を明らかにすることで心のもやもやを晴らしてあげようとするわけです。自作 CPU 最大の益はこれです。

1.3　ロジック IC

ロジック IC というものをご存じですか。需要も少なくなって生産終了品が多くなっています。74 シリーズと 4000 シリーズが有名どころです。AND とか OR など論理演算はプログラムを書く人ならご存じでしょう。たとえば 7408 では 2 入力の AND 素子が 4 つ封入されています。論理演算以外にもエンコーダ、デコーダ、トライステート、レジスタ、カウンタなど様々あります。昔の機械に合わせた専用回路はこれらを組み合わせて構成されていました。昔の基板を眺めてみると DIP の IC がたくさん並んでいる様子が見て取れるはずです。IC に書いてある型を読んでみましょう。74LSxx と書いている場合がほとんどだと思われます。

▲図 1.1: ロジック IC

1.4　自作 CPU 流派の違い

自作 CPU と一口に言っても流派があるのはご存じでしょうか。大きく分けて二つに分かれます。ソフトウェア派とハードウェア派です。そもそも CPU を構成するための論理回路というものは「真」と「偽」の二値で示す概念的なものであるので回路を電気で組む必要性はないです。伝達速度含め様々な点で大変便利だから電気が使われるのです。ソフトウェア派では CPU をパソコン上で再現します。さらに FPGA [*1] に書き込んで現実世界に召喚します。いずれにしてもソフトウェアで CPU を構成する方法です。対してハードウェア派では論理回路の構成を実際の電子部品で行う流派です。TTM8 構成で採用しているロジック IC を用いた構成が最も一般的ですが中には、トランジスタやリレーで組んでしまう猛者もいます。実際に配線を組まなくてはならないので、自作 CPU という題材も相まって非常に密な回路となる事が多いです。

ソフトウェア派とハードウェア派が存在しますが、多くの場合ソフトウェア派を選択する人が多いです。開発が安易な点、配線を考慮しなくてよい点、再現性、FPGA を用いれば現実世界にも召喚できる点。ほとんどの要素で優れているからです。わざわざハードウェア派を選択して開発を進める稀有な人はそれを選択した明確な意思を持っていると考えます。これは私、個人の考えなのですが、ソフトウェアというものはどうしても現実世界に手に取れる形では存在できないものです。だから完成した達成感と実感はハードウェアと比べると薄いと私は思っています。ハードウェアという簡単に変えられなくリアルに重さを感じられる"物"としての存在は訴えかけるパワーは別格であるとこれまでの工作経験からの確信です。こうした点から私はハードウェアを選択するのです。

1.5　ロジック IC 派の利点

ハードウェア派の中では一般的な手法であるロジック IC を用いた構成方法では様々な点で利点があります。箇条書きでこれらを記すと以下のようになります。

[*1] 論理回路を書き換え可能にする IC

2.1　コンピュータを構成する要素

いまやほぼすべての人が手に持っているコンピュータ。コンピュータを構成する要素を解説します。スマホやパソコン。そのほかには車や冷蔵庫の中で動かしているのもコンピュータです。コンピュータを構成する要素は 3 つです。**CPU**、**メモリ**、**I/O** です。本誌のテーマである CPU は計算を担う部分です。メモリはプログラムが保存されたり計算用のデータが保存される場所です。そして I/O はこれら CPU とメモリにデータを入力したり、計算されたデータを出力すう部分です。パソコンでいうとディスプレイやキーボード、HDD がこれに当たります。まずはこれら三つの要素でコンピュータが成り立っているということを覚えておきましょう。

2.2　組み合わせ回路と順序回路

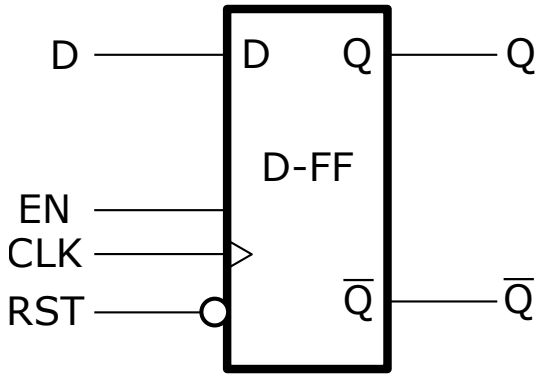
論理回路は組み合わせ回路と順序回路で構成されます。AND ゲートや OR ゲートのように入力に対して即座に出力が反映されるような回路です。出力が現在の値によってのみ決まる回路とも言えます。対して順序回路とは過去の状態を含めて出力が決まります。カウンタが例に挙げられます。カウンタはカウントアップやカウントダウンする特性上過去のデータが必要で保持しておく必要があります。そこで使われるのがレジスタです。

2.3　レジスタ

レジスタはデータを保持しておくための入れ物です。4bit 幅のレジスタであれば 4bit のデータが。8bit 幅のレジスタでは 8bit のデータが格納されます。格納とはいうけれど一体どうやってデータを保持するのか。使われるのはフリップフロップ (以下 FF) という回路です。FF には入力 D とクロック入力とリセット入力があります。出力は単純に今保持しているデータを出力する Q のみです。大事なのは一つで**クロック入力の立ち上がり時の入力 D の値を保持する**だけまず覚えてください。クロックの立ち上がりというのが特に大切で CPU が持っているすべてのレジスタがこのタイミングで値を更新します。

LOAD 入力

IC としてパッケージに入っている DFF にはレジスタとして使用しやすいように LOAD 入力が備わっているものがあります。LOAD 入力が有効であると入力 D がラッチされ格納されます。LOAD 入力が無効であると格納中の値が保持され続けます。よって、このレジスタにデータを格納するか否かを選択できるようになります。



▲図 2.1: D フリップフロップ

フリップフロップの種類

フリップフロップは日本語でいうところのギッコンバッタンという擬音とのことです。0 と 1 がシーソーのように入れ替わる様が相応しい名前ですね。FF にもいくつか種類があります。今回の説明では D-FF というものを説明しています。D は"Delay"の D で入力の状態が遅れて出力されている事を示しています。レジスタにはおおよそ D-FF が使われています。他には RS-FF、JK-FF、T-FF があります。大抵の場合フリップフロップの IC には Q 出力と $\bar{Q}$ 出力があります。わざわざ $\bar{Q}$ 出力を設けているわけではなくフリップフロップを構成すると必然的に $\bar{Q}$ 出力が現れるのです。

2.4　クロックとリセット

CPU を含める同期回路を考えるうえで大切なのがクロックです。同期とはつまりクロック同期を意味します。すべてのレジスタがクロック入力を持っており、クロックの立ち上がりに同期して動作します。なのでクロックの速度を早めれば CPU の動作は早くなりますし逆に遅くすれば遅くなります。パソコンのオーバークロックとはこのことですね。

リセットは簡単です。これもクロック同様にすべてのレジスタに入力されています。この信号が 0 になるとすべてのレジスタが初期値にリセットされプログラムが初めからスタートするのです。

2.5　アドレスとメモリ

さて少し難しいかもしれませんがアドレスという概念を理解しなくてはいけません。アドレスとはその名の通り住所です。レジスタとはデータを保持しておく入れ物だと説明しました。この入れ物の住所をしめすものがアドレスです。TTM8 での例を示しますと、C レジスタはアドレス C2

3.1　TTM8 の特徴

それでは、いよいよ本題に入ることになります。本章からオリジナル CPU、TTM8 を解説していきます。TTM8 最大の特徴はやっぱり自作 CPU としてロジック IC で組みあがっていることです。が、一旦ロジック IC で組みあがっている事は忘れてもらって完成した CPU としての TTM8 についてお話します。どんな CPU でも基本的な動きの流れは似ているものですがそれでも CPU によって個性というものがあります。TTM8 も特徴的な論理を持っているのでポイントに分けて解説していきます。なおこれらの説明文は TTM8 に関しての説明であるので一般的な CPU においての説明とは異なっている可能性があります。ご留意ください。使い方自体は取扱説明書をご確認ください。

▼表 3.1: TTM8 スペック表

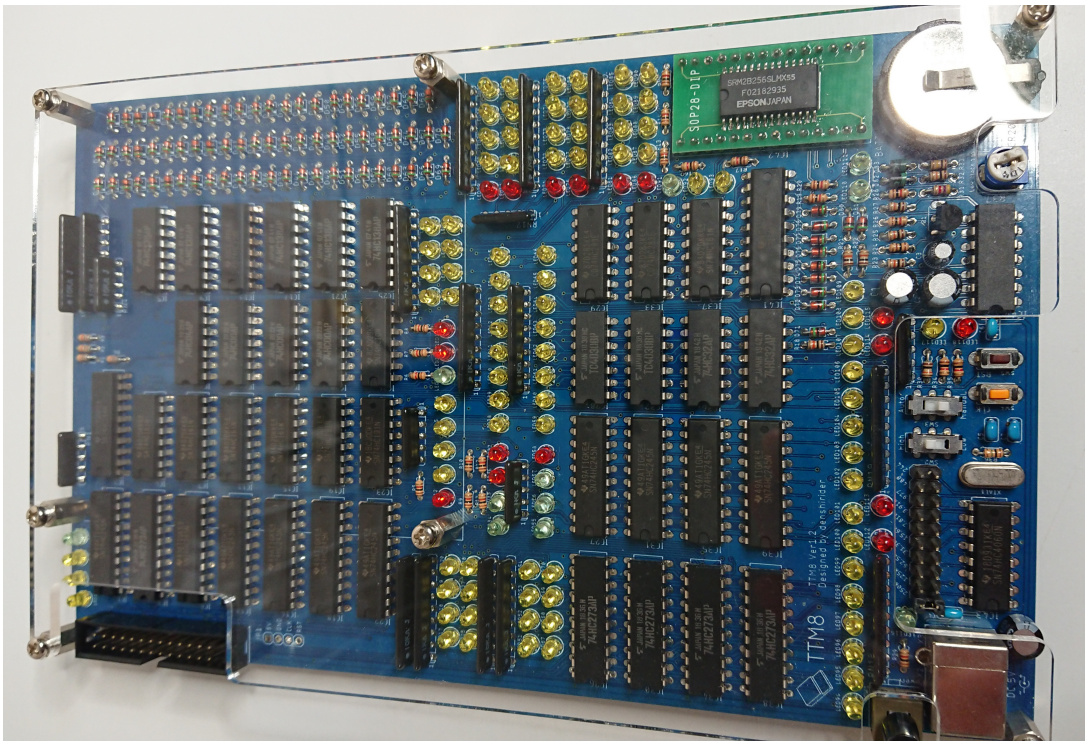
電源電圧	5V
消費電流	1A 以下
bit 数	8bit
命令種類	16 種類
総レジスタ数	10 個
汎用レジスタ数	2 個
1 プログラムにおける最大命令数	64 命令
スタックおよびユーザ操作用 RAM 領域	32Byte
クロック	ジャンパーピンとスイッチにて選択 段階的に 250kHz から 244Hz、低周波クロック、手動クロック
入出力	なし (拡張して任意の I/O を付与できる)

3.2　ブロック図

CPU 含め論理回路を理解する上で大いに役立つのがブロック図です。CPU を構成する各ブロックがどういったつながりをしているのか一目で分かるのがポイントです。各ブロックはデータバスあるいはアドレスバスで接続されておりデータのやり取りはデータバス、アドレスのやり取りはアドレスバスで行われます。各ブロックについてまずは概要をご説明いたします。図 3.2 にブロック図を示しました。

3.3　TTM8 のレジスタの説明

TTM8 には A レジスタから D レジスタさらにプログラムカウンタ、スタックポインタなど含めて 10 種類のレジスタを持っています。ここではそれぞれの概要について説明します。



▲図 3.1: TTM8 写真

計算用レジスタ

ブロック図の左側に目を向けてみると ALU(Arithmetic Logic Unit) というブロックがあります。これが CPU の肝の一つである計算を実行するブロックです。ALU がつながっているのは A レジスタ、B レジスタ、フラグレジスタ、データバスです。フラグレジスタは一旦置いておくと、ALU は A レジスタに格納されている値と B レジスタに格納されている値を計算してデータバスに流すという役割です。表題の計算用レジスタというのはこの A レジスタ、B レジスタの事です。これらのレジスタは ALU へ値を渡す専用のレジスタで自由に使えるレジスタではありません。A レジスタ、B レジスタは計算元の値が入るのだと覚えておきましょう。

自由に使えない A レジスタ、B レジスタ

ALU 専属の計算用レジスタ、A レジスタ、B レジスタ。自由に操作できないレジスタであるがこのように役割が決められると CPU 内部処理が簡潔になる。あれにもこれにも対応するとそれだけ回路が必要になるのだ。特に今回の場合ロジック IC による自作 CPU である。回路の単純化は使用する IC の数へ直結する。

汎用レジスタ

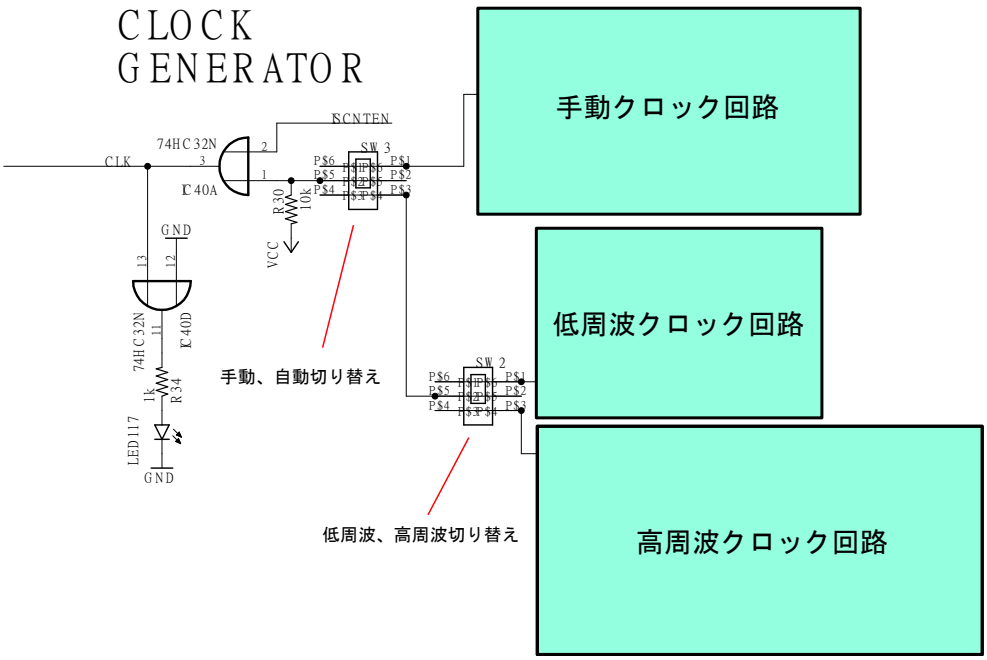
A レジスタ、B レジスタの下に目を向けてみると C レジスタ、D レジスタがあります。これら

4.1　クロックとリセットの構成

クロックとリセットは CPU に外部から与えられる信号です。この二つの要素がなければ動作することはできません。

クロック

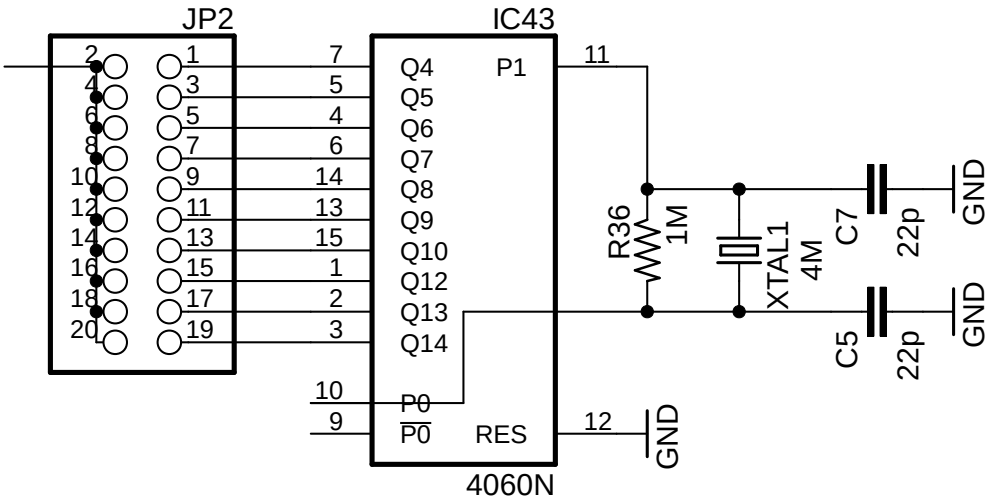
クロックは CPU の動作の要となる重要な信号です。外部より生成する必要があります。クロック生成回路をクロックジェネレータと言います。こいつの精度が CPU の安定性に直結するので適当な設計は許されません。TTM8 では大きく 3 つのクロックを選択することができます。高周波クロック、低周波クロック、手動クロックです。それぞれのクロックはスイッチにて切り替える方式です。それぞれ実際の回路を見ていきます。



▲図 4.1: クロックジェネレータ

高周波クロック

高周波クロックはクリスタルから生成された精度の高いクロックです。(この場合の精度とは周波数にブレがないこと。)4MHz のクリスタルを発振回路を内臓したカウンタ 4060 という IC に入力して分周しています。4MHz の周波数はロジック IC で組んだ CPU には高速であるのでカウンタである 4060 を通して分周したクロックを TTM8 に流しています。最大で 250kHz。最小で 244Hz です。ピンヘッダを使ってショートピンで切り替えます。



▲図 4.2: 高周波クロック

動作可能な最大周波数は?

組み合わせ回路は現在の入力で出力が決まりますが物理的な回路である以上どうしても数ナノ秒から数マイクロ秒程度のラグがあります。最大の動作周波数を知るには一番遠回りで複雑な組み合わせ回路の遅延時間を知る必要があります。この遅延時間がクロックの周期より短ければ問題なく動作するはずです。TTM8 では回路をロジック IC で組んでいますから各 IC データシートを眺めて遅延時間を調べます。そして一番遅延時間が長い経路の遅延時間が分かれば最大動作周波数が分かるのです。

低周波クロック

低周波クロックは抵抗とコンデンサで組んだ RC 発信回路です。抵抗とコンデンサの値により周波数が決定されますが抵抗を可変抵抗としていることである程度の範囲を自由に設定することができます。ただし精度は悪く任意の周波数にピタっと合わせるのは難しです。大体設定した付近の周波数で発振します。回路はシュミットトリガインバータ (7414) を使用した回路です。抵抗値は 100Ω と可変抵抗 $100k\Omega$。コンデンサは $33uF$ となります。シュミットトリガインバータとは、論理的には NOT 素子と同様ですが閾値にヒステリシスを持った素子です。すなわち 0 から 1 に切り替わる閾値と 1 から 0 に切り替わる閾値が異なります。このように閾値が二つあることでアナログ的なカーブを描いた電圧を入力にしても動作が不安的になりません。

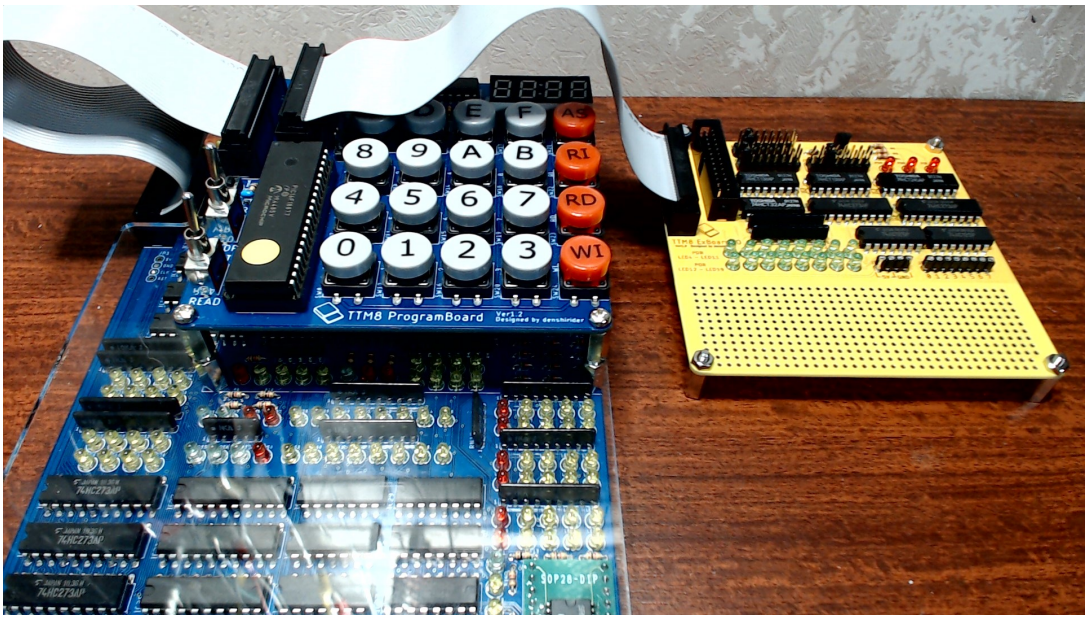
計算式

1. 電源投入始めコンデンサは充電されていない。
2. コンデンサが充電されていないため入力が L となる。
3. インバータで反転して OUT が H となる。
4. OUT から電流が抵抗を通りコンデンサに充電される。

5.1　拡張 I/O を自作しよう

TTM8 に I/O を拡張してコンピュータとして完成させましょう。I/O はパソコンでいうところのディスプレイ、キーボードですけれど組み込み開発の界隈では通信やタイマーの機能を言います。マイコンを選定した経験のある方はどんな通信機能を持っているか、タイマーは何チャンネルあるのか、PWM は何本だせるかなどを見て選定したことと思います。I/O をつくるとはつまりこのような周辺機能を創りこむことです。これからは何チャンネルあるかなど気にする必要はありません。好きなだけ増設しましょう。

拡張 I/O を載せた回路基板を TTM8 拡張ボードとします。よく見ると TTM8 プログラムボードには拡張ポートが二口あります。配線を次々と数珠繋ぎしていき拡張ボードをつなげていくことができます。



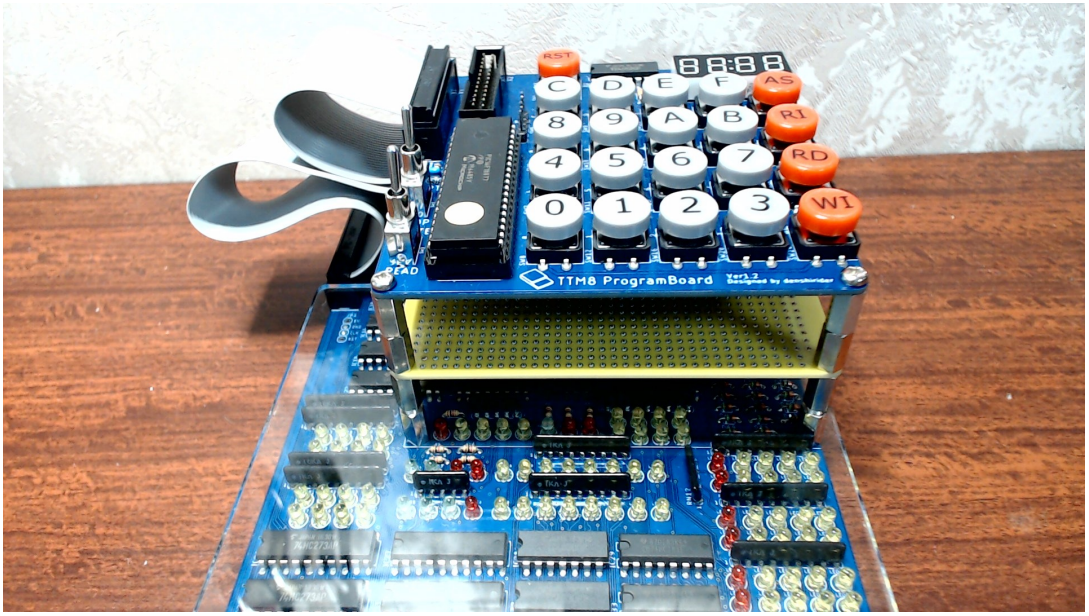
▲図 5.1: 拡張ボードを接続する

I/O のレジスタ

周辺機能を使うときは何かと設定が必要です。タイマーであればどうでしょうか。以下のような項目を設定しなければいけません。

- 入力するクロック
- 入力したクロックの分周比
- リセットするカウント値

タイマーは中にカウンタを持っているはずですがこのカウンタに入力されるクロックの周波数に



▲図 5.2: 拡張ボードを積み上げて接続する

よってタイマーに設定される時間は大きく変わります。100MHz でカウントしていくのと 100kHz でカウントしていくのは明らかです。なので入力するクロックの設定は備えている事が多いでしょう。1 カウントの時間を決めたら次はどれだけカウントするか決めなくてはなりません。同じ時間で数えていっても 5 回数えるのと 100 回数えるのでは違いますよね。なのでリセットするカウント値は設定する必要があるでしょう。

このような設定をどうやって組んでいくかというとレジスタを用います。レジスタはデータを入れておく箱でした。この箱にクロックを選択する設定やカウント値を入れておくのです。タイマーが適切に組まれていれば設定どおりに動いてくれることでしょう。

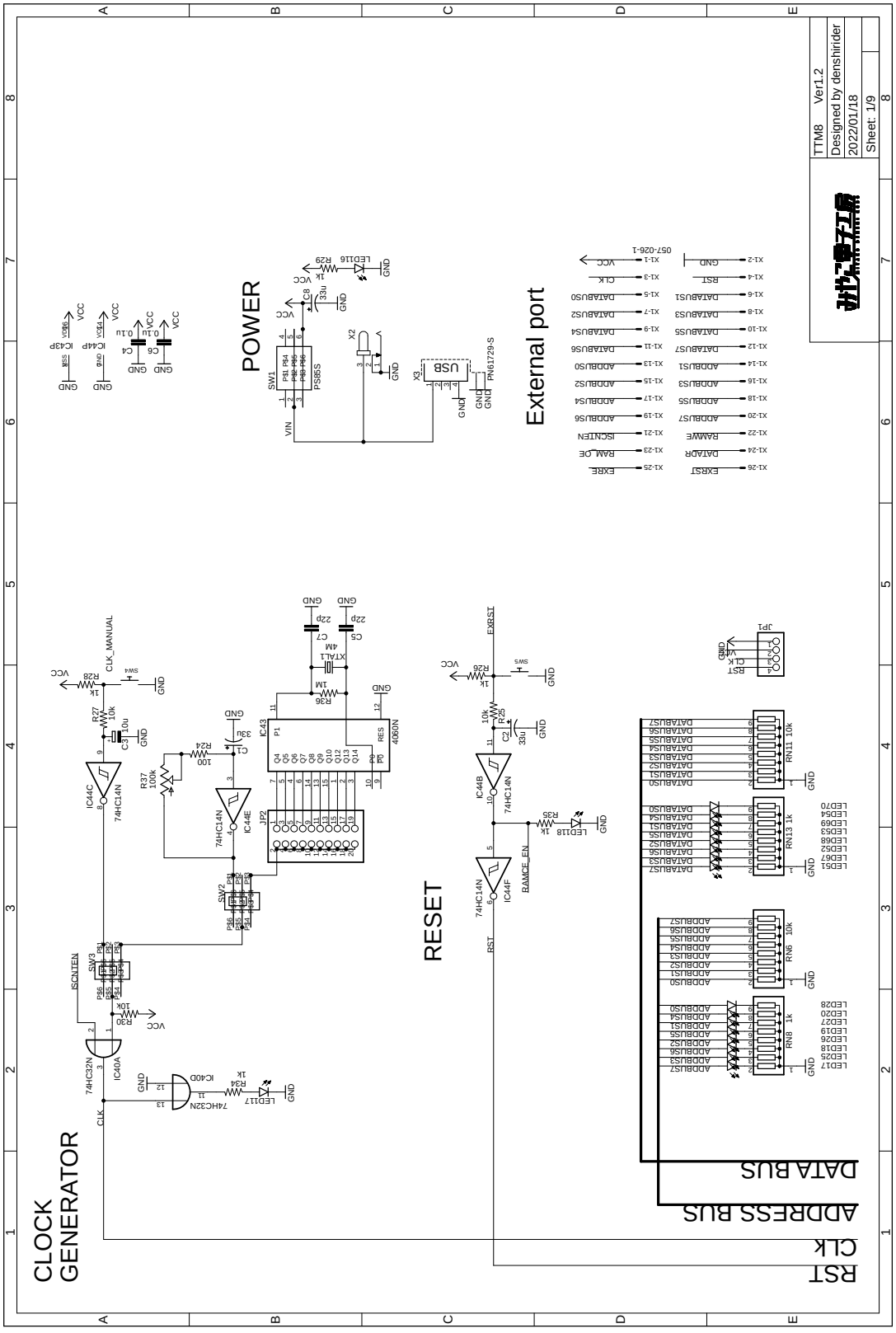
出力ポート、入力ポート

初めからタイマーを構成すると大変なので最も簡単な 0,1 を操作する入力ポートと出力ポートを考えてみます。

出力ポート

「3.12 プログラム例」(p.27) で示したプログラム例では出力ポートの代わりに C レジスタ D レジスタにぶら下げっている LED で表示していました。出力ポートはつまりこれの事です。C、D レジスタのような汎用で使用できるレジスタを外部において LED の代わりにピンヘッダなどに出力してしまえば自由に引き出せる出力ポートの完成です。出力ポートレジスタにデータを書き込めば書き込んだデータが出力ポートから出てくるはずです。

C、D レジスタは汎用のレジスタなので書き込むこともできますが読み込むこともできるようにデータバスに返していました。出力ポートとしてのレジスタであれば必ずしも読み込めるようにす


TTM8 Ver1.2
Designed by denshirider
2022/01/18
Sheet: 1/9
CLOCK GENERATOR
POWER
RESET
External port
RST
CLK
ADDRESS BUS
DATA BUS
CLK_MANUAL
ISCNTEN
RAMCE_EN
EXRST
USB
PN61729-S
057-026-1
4060N
74HC14N
74HC32N
PSMSS

ロジック IC で創る CPU キット TTM8 解説本

これから CPU を学ぶあなたのために

2022 年 1 月 22 日　 ver 1.0 （技術書典 12）

著　者　　電子ライダー
イラスト　成瀬歩実
発行所　　みやこ電子工房
発行者　　電子ライダー
連絡先　　denshirider@gmail.com
　　https://denshirider.github.io/denshirider-site/index.html
　　@denshirider (https://twitter.com/denshirider)

(powered by Re:VIEW Starter)