

TeC7 ハードウェアマニュアル Ver. 1.0.1

徳山工業高等専門学校
情報電子工学科

Copyright © 2019 - 2020 by
Dept. of Computer Science and Electronic Engineering,
Tokuyama College of Technology, JAPAN

本ドキュメントは CC-BY-SA 4.0 ライセンスによって許諾されています。

目次

| | | |
|--------------|------------------------------|----------|
| 第 1 章 | 概要 | 1 |
| 1.1 | TeC7 | 1 |
| 1.2 | TeC7 の外観 | 1 |
| 1.3 | TeC7 の内部構造 | 3 |
| 1.3.1 | クロックとリセット | 4 |
| 1.3.2 | 動作モード | 4 |
| 1.3.3 | TaC による TeC の補助 | 4 |
| 1.4 | TeC の内部構造 | 5 |
| 1.4.1 | CPU, メモリ, 入出力装置 | 5 |
| 1.4.2 | コンソール | 5 |
| 1.4.3 | 割り込みコントローラ | 5 |
| 1.5 | TaC の内部構造 | 6 |
| 1.5.1 | CPU | 6 |
| 1.5.2 | コンソール | 6 |
| 1.5.3 | 割り込みコントローラ | 6 |
| 1.5.4 | メモリ | 7 |
| 1.5.5 | MMU (Memory Management Unit) | 7 |
| 1.5.6 | マイクロ SD ホストコントローラ | 7 |
| 1.5.7 | I/O ポート | 7 |
| 1.5.8 | タイマー | 7 |
| 1.5.9 | シリアル I/O (SIO) | 7 |
| 1.5.10 | TeC アダプタ | 7 |
| 1.5.11 | RN4020 アダプタ | 7 |
| 第 2 章 | TeC7 の操作方法 | 9 |
| 2.1 | ジャンパの設定方法 | 9 |
| 2.2 | コンソールのランプやスイッチ | 10 |
| 2.2.1 | アドレスランプ・データランプ | 10 |
| 2.2.2 | ロータリースイッチ | 10 |

| | | |
|--------------|--------------------------------|-----------|
| 2.2.3 | データスイッチ | 11 |
| 2.2.4 | プログラム実行に使用するランプとスイッチ | 11 |
| 2.2.5 | データ書き換えに使用するスイッチ | 11 |
| 2.3 | 操作手順 | 11 |
| 2.3.1 | リセット | 11 |
| 2.3.2 | CPU レジスタや PSW の表示と書き換え | 11 |
| 2.3.3 | メモリの表示と書き換え | 11 |
| 2.3.4 | プログラムの停止・実行・デバッグ | 12 |
| 第 3 章 | TaC のアーキテクチャ | 15 |
| 3.1 | CPU の概要 | 15 |
| 3.1.1 | データ形式 | 15 |
| 3.1.2 | アドレス空間 | 15 |
| 3.1.3 | 実行モード | 15 |
| 3.1.4 | CPU レジスタと PSW | 16 |
| 3.1.5 | 機械語命令 | 16 |
| 3.1.6 | 割込み (Interrupt)・例外 (Exception) | 17 |
| 3.2 | メモリマップと I/O マップ | 17 |
| 3.2.1 | メモリ空間 | 18 |
| 3.2.2 | I/O 空間 | 18 |
| 3.3 | MMU (Memory Management Unit) | 18 |
| 3.3.1 | 違反アドレス | 18 |
| 3.3.2 | 違反原因 (0000 00BV) | 18 |
| 3.3.3 | ページ番号 | 19 |
| 3.3.4 | IPL 切離し (0000 000I) | 19 |
| 3.3.5 | MMU 有効化 (0000 000E) | 19 |
| 3.3.6 | TLB (TLB[0]～TLB[7]) | 19 |
| 3.4 | IPL プログラム | 20 |
| 3.4.1 | TeC モード | 20 |
| 3.4.2 | TaC モード | 20 |
| 3.4.3 | DEMO モード | 21 |
| 3.4.4 | RESET | 21 |
| 3.5 | 周辺装置 | 21 |
| 3.5.1 | タイマー | 21 |
| 3.5.2 | FT232RL (シリアル I/O) | 22 |
| 3.5.3 | TeC (シリアル I/O) | 22 |
| 3.5.4 | マイクロ SD ホストコントローラ | 22 |
| 3.5.5 | 入出力ポート他 | 23 |

| | | |
|--------|---------------------------|----|
| 3.5.6 | SPI インタフェース | 23 |
| 3.5.7 | 入力ポート割り込み | 24 |
| 3.5.8 | RN4020 アダプタ | 25 |
| 3.5.9 | TeC アダプタ | 25 |
| 3.5.10 | コンソール | 26 |
| 付録 A | TaC に関する資料 | 29 |
| A.1 | データ形式 | 29 |
| A.2 | メモリマップと I/O マップ | 29 |
| A.3 | 機械語命令表 | 29 |
| A.4 | プリント基板回路図 | 29 |

第 1 章

概要

このマニュアルでは TeC7 のハードウェアと IPL プログラムについて解説を行う。

1.1 TeC7

TeC7 は、内部に TeC と TaC の二つのコンピュータを内蔵したマイコンボードである。

TeC 高校生や高専の低学年の学生が、ノイマン型コンピュータの動作原理を学ぶために開発された 8 ビットコンピュータである。コンソールパネルを用いて、二進数で機械語プログラミングを体験することができる。TeC については、「TeC 教科書」^{*1} に詳しい説明がある。

TaC 大学生や高専の高学年の学生がオペレーティングシステムやコンパイラを学習する際に、ターゲットとなるコンピュータのサンプルとして開発した 16 ビットコンピュータである。本マニュアルは、主に TaC として使用する際の TeC7 について解説する。

1.2 TeC7 の外観

図 1.1 に TeC7 の写真を示す。TeC7 は一枚のプリント基板に実装されている。以下に基板の主要な部品などを紹介する。

コンソールパネル ユーザはコンソールパネルを用いて、TeC または TaC の CPU レジスタやメモリの内容を読み書きしたり、プログラムを機械語命令単位でステップ実行したりすることができる。つまり、コンソールはハードウェア仕掛けのデバッグである。TeC では機械語プログラムのデバッグに、TaC ではオペレーティングシステムのデバッグに使用する。オペレーティングシステムのカーネル内部まで、ステップ実行しながらデバッグすることが可能である。

JTAG コネクタ FPGA を設定（コンフィグ）する設計データを書き込むために使用する。プリント基板上で FPGA とフラッシュメモリからなる JTAG チェインを構成しており、JTAG コネクタから FPGA とフラッシュメモリにアクセスすることができる。

スピーカ コンソールを操作した際に操作音を発生する。TeC は電子オルゴールプログラム等で使用することもできる。

^{*1} <https://github.com/tctsigemura/TecTextBook/raw/master/tec.pdf>

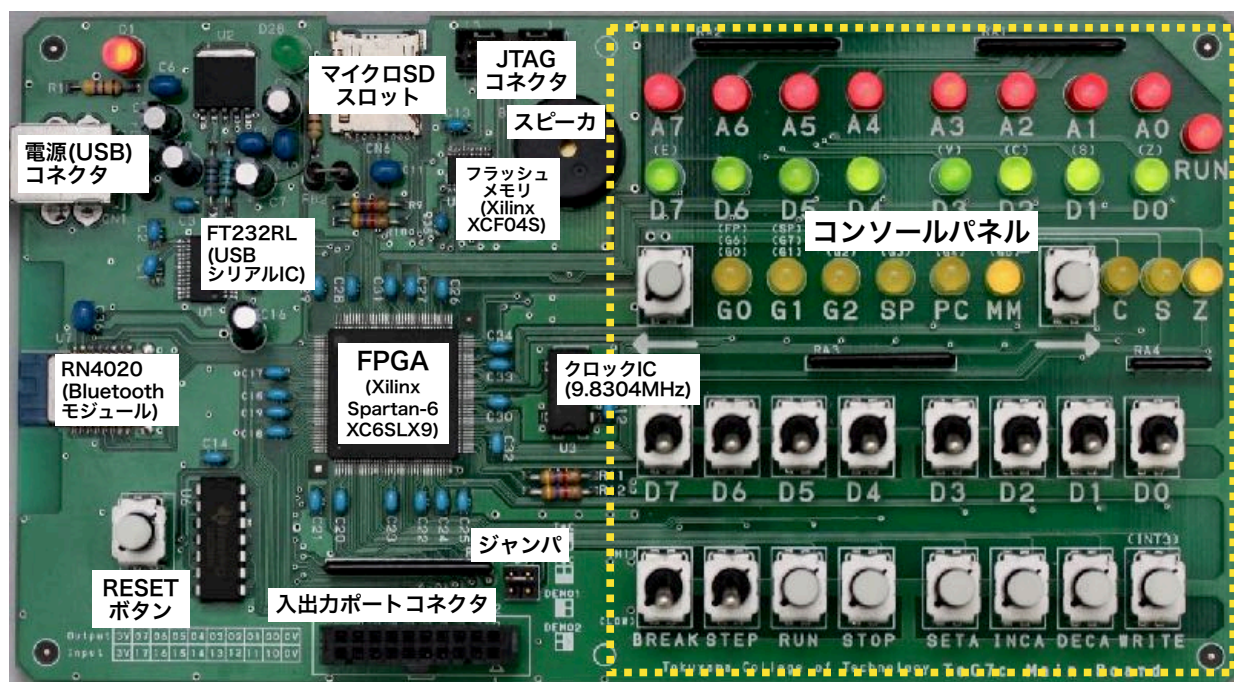


図 1.1 TeC7 の写真

フラッシュメモリ 電源が遮断されても内容が消えないメモリである。TeC7 に電源が投入された時、フラッシュメモリから FPGA コンフィグ用のデータが読み出される。内容は JTAG コネクタから書き換えることができる。使用している部品は、Xilinx XCF04S である。

マイクロ SD スロット TaC の二次記憶装置としてマイクロ SD を使用することができる。

電源 (USB) コネクタ 電源を供給するために使用する。また、FT232RL と接続してあるので PC とシリアル通信をすることも可能である。

FT232RL PC と USB で接続してシリアル通信をするための IC である。

クロック IC 9.8304MHz のクロック信号を出力する水晶発振器である。FPGA にクロック信号を供給する。

FPGA TeC, TaC の CPU, メモリ等、全ての主要な論理回路を内蔵する。使用している FPGA は、Xilinx Spartan-6 XC6SLX9 である。

RN4020 BLE (Bluetooth Low Energy) 規格の通信モジュールである。FT232RL と同様な通信を Bluetooth 経由で行うことができる^{*2}。b バージョン以降の TeC7 に実装されている。

ジャンパ TeC7 を TeC として使用するか、TaC として使用するかを定める。その他に、Bluetooth モジュールのリセットや、デモンストレーション機能の呼び出しにも使用できる。

^{*2} 通信相手には BlueTerminal (<https://github.com/tctsigemura/BlueTerminal>) をインストールする必要がある。

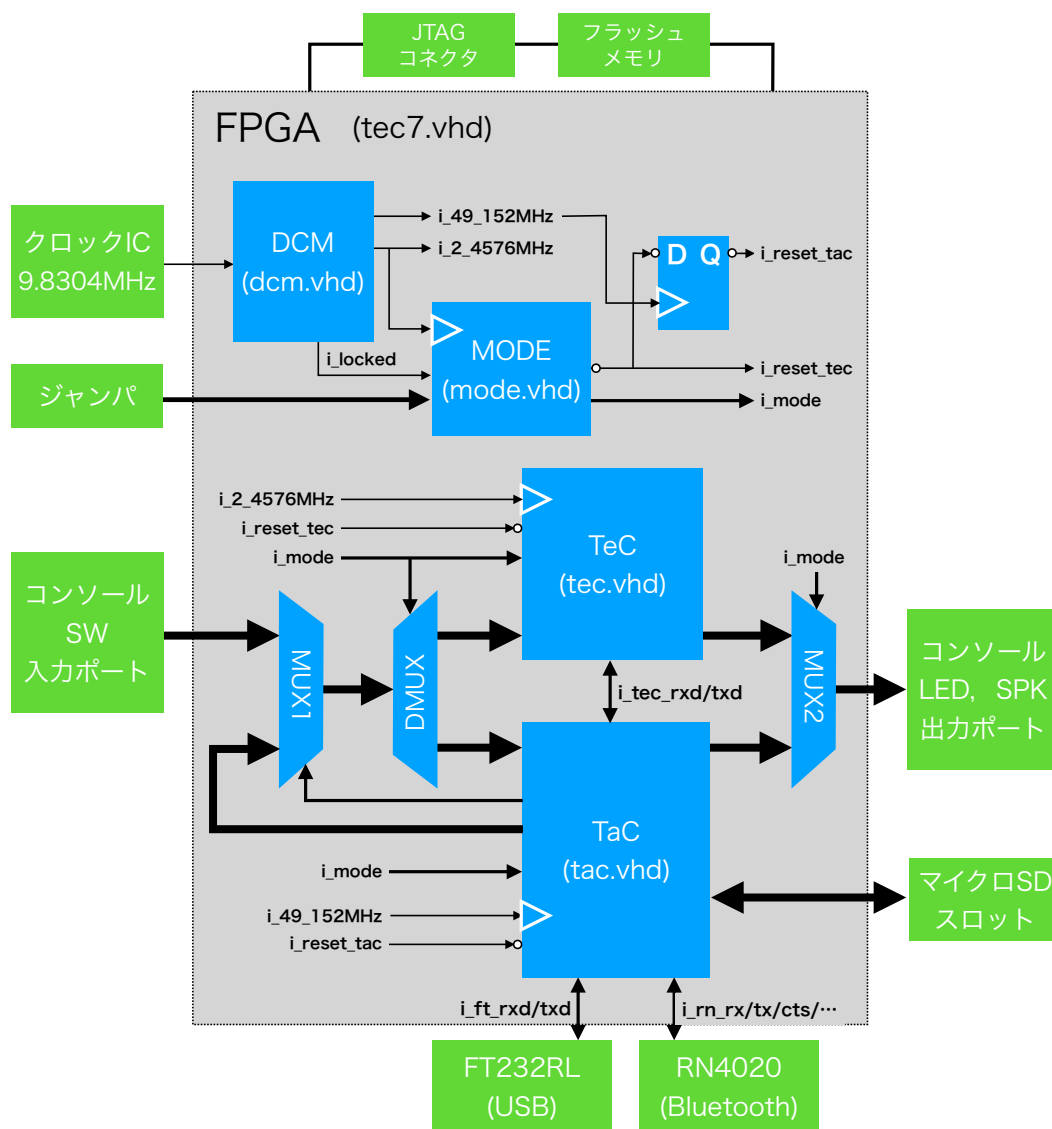


図 1.2 TeC7 のブロック図

1.3 TeC7 の内部構造

図 1.2 に TeC7 のブロック図を示す。図中央の灰色の大きな長方形は FPGA を表しており、主要な論理回路は全て FPGA に内蔵されていることが分かる。FPGA 内部の回路は VHDL で記述されている。TeC7 の回路を記述した VHDL のソースコードは GitHub^{*3} に公開してある。緑色の長方形はプリント基板上に実装された FPGA 以外の部品である。基板の回路図を付録 A、図 A.5 に示す。以下では、図 1.2 を参照しながら TeC7 の回路構成の概要を説明する。

*3 <https://github.com/tctsigemura/TeC7/tree/master/VHDL>

表 1.1 i_mode の値と意味

| i_mode | 意 味 |
|--------|---------------------------------------|
| 000 | TeC モード (TeC7 が TeC として動作) |
| 001 | TaC モード (TeC7 が TaC として動作) |
| 010 | DEMO1 モード (電子オルゴールプログラム入力済みの TeC モード) |
| 011 | DEMO2 モード (演奏データ入力済みの DEMO1 モード) |
| 111 | リセット (RN4020 を工場出荷時の状態に戻す) |

1.3.1 クロックとリセット

DCM は、外部から供給される 9.8304MHz のクロック信号から、Spartan-6 の DCM (Digital Clock Manager) を用いて、TeC 用の 2.4576MHz、TaC 用の 49.152MHz クロック信号を生成する。DCM は電源投入後、クロック出力が安定すると i_locked を '1' にする。

i_locked が '1' になると MODE はジャンパの設定を読み取り結果を i_mode に出力する。ジャンパの読み取りが完了すると、i_reset_tec と i_reset_tac が '1' になり、TeC と TaC が動作を開始する。

1.3.2 動作モード

i_mode の値により TeC7 の動作モードが決まる。i_mode の値と動作モードの対応を表 1.1 に示す。

- 「TeC モード」、「DEMO1 モード」、「DEMO2 モード」では、TeC がコンソールと接続される。その様子を図 1.2 で確認する。図中の「TeC」が TeC コンピュータである。この内部に、TeC の CPU や主記憶、入出力装置などの回路が組み込まれている。i_mode の値が 001 (TaC モード)、111 (リセットモード) 以外の場合、図中のデマルチプレクサ (DMUX) とマルチプレクサ (MUX2) が切り替わり TeC とコンソールが接続される。i_mode の値が「DEMO モード」の場合は、TeC のメモリに予め電子オルゴールプログラムが入力された状態になる。
- 「TaC モード」では、TaC がコンソールと接続される。図中の「TaC」が TaC コンピュータである。i_mode の値が 001 (TaC モード) の場合は、TaC にコンソールが接続される。
- 「リセットモード」では、TeC も TaC もコンソールに接続されない。

TeC は「TaC モード」では停止したままになる。一方で TaC は i_mode の値に関係なく IPL プログラムの実行を開始する。IPL がモードに対応した動作を行う。

1.3.3 TaC による TeC の補助

TeC のシリアル入出力 (i_tec_rxd/txd) は TaC に接続されており、「TeC モード」では TaC がシリアル入出力の中継装置の役割を担う。通常、TaC はシリアル入出力を FT232RL に中継する。しかし、RN4020 が Bluetooth 接続を確立した場合は、FT232RL と RN4020 の両方に中継ようになる。TeC は TeC7 が USB と Bluetooth のどちらで PC に接続されているのか知る必要がない。

通常、図中の「MUX1」はコンソールを「DMUX」に接続している。「TeC モード」時に、シリアル通

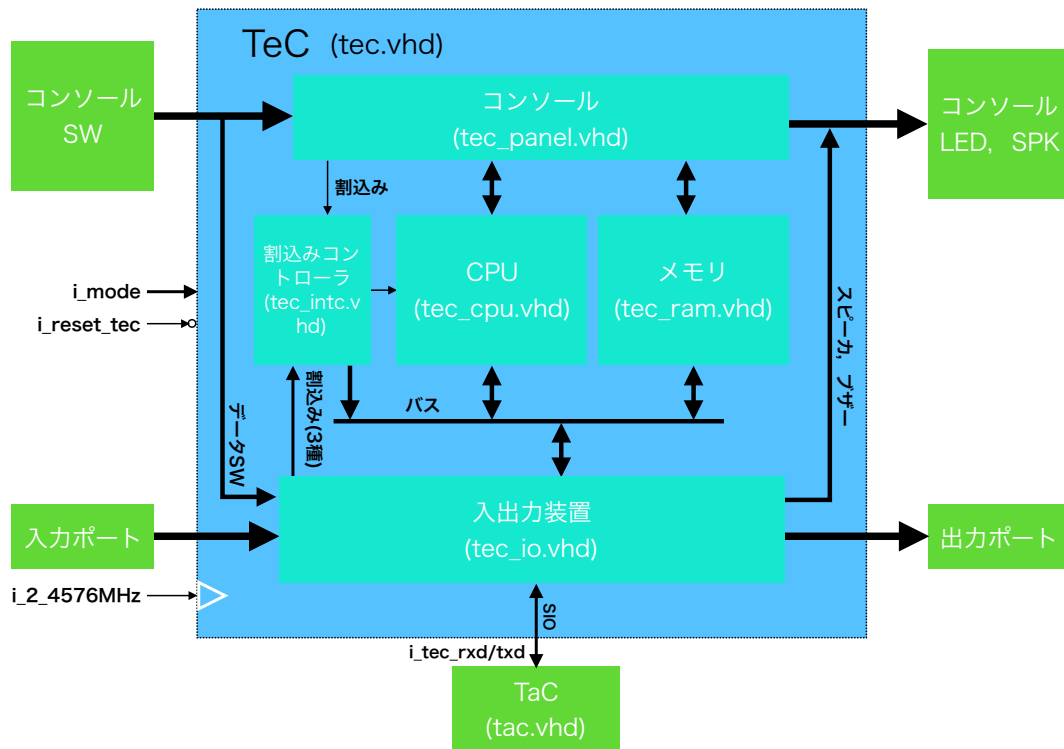


図 1.3 TeC のブロック図

信で受信した内容が TWRITE プログラム^{*4}のものなら、通信を中継している TaC が“MUX1”を切り換え TeC のコンソールを操作し、受信したプログラムを TeC のメモリに書き込む。この機能は TaC の IPL プログラムに組み込まれている。

1.4 TeC の内部構造

図 1.3 に TeC のブロック図を示す。この図は、図 1.2 の“TeC”ブロックの内部を表している。

1.4.1 CPU、メモリ、入出力装置

CPU とメモリや入出力装置はバスを介して接続されている。入出力装置には、シリアル入出力 (SIO)、入出力ポート (PIO)、タイマー、A/D コンバータの機能が含まれる。SIO は TaC に接続されており、通信データは TaC が FT232RL や RN4020 に中継する。コンソールのデータ SW やスピーカは入出力装置としても使用することができる。

1.4.2 コンソール

コンソールは、CPU とメモリに専用の回路で接続されている。完全にハードウェア制御で動作するので、プログラム実行中でも操作が可能である。

1.4.3 割り込みコントローラ

割り込みコントローラは、コンソール、入出力装置から発生する四種類の割り込みを CPU に伝える。CPU が割り込み認識サイクルに入ったらバスに割り込み番号を出力する。

^{*4} Util-- (<https://github.com/tctsigemura/Util-->) に含まれるプログラム書き込みツールのこと。

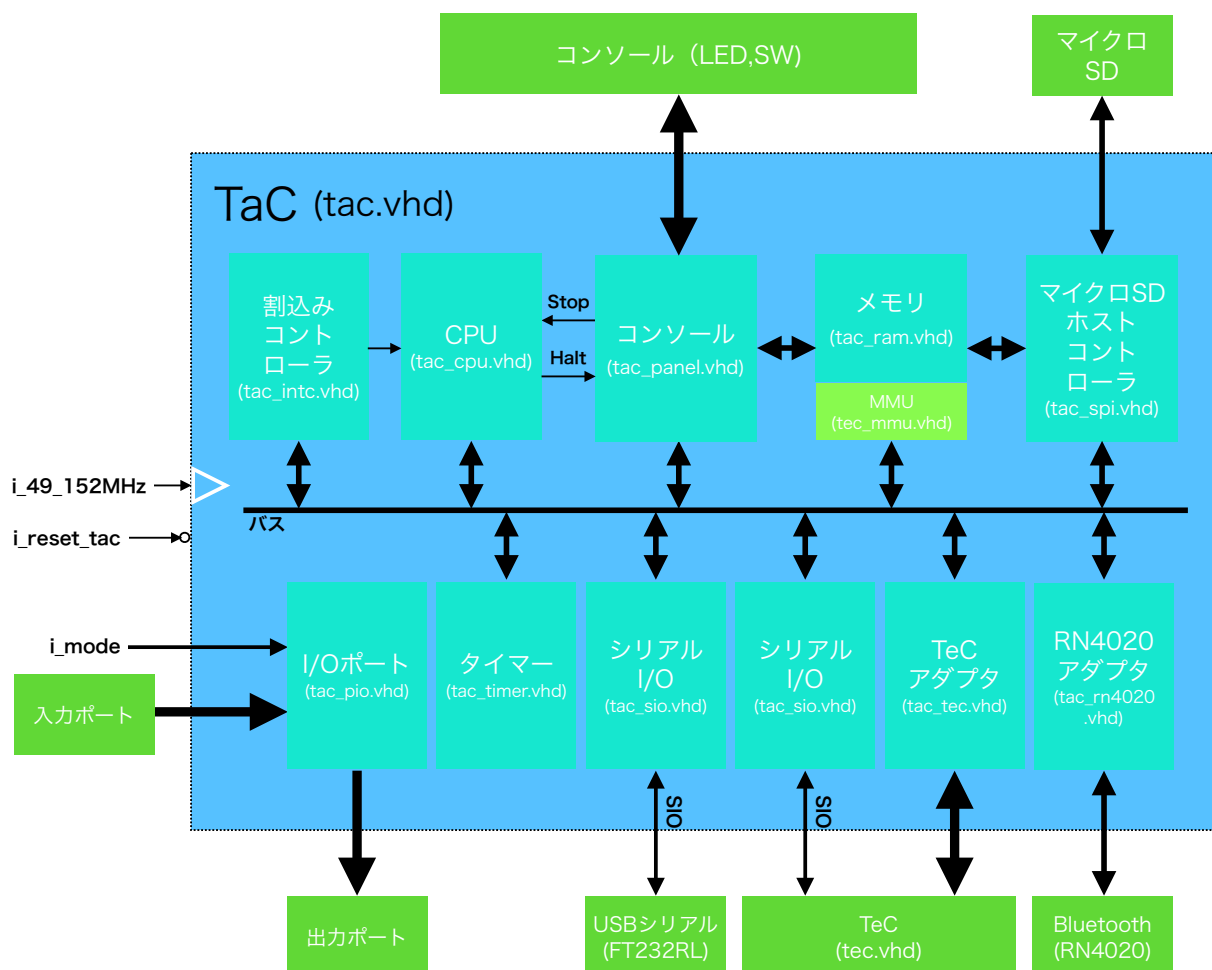


図 1.4 TaC のブロック図

1.5 TaC の内部構造

図 1.4 に TaC のブロック図を示す。この図は、図 1.2 の “TaC” ブロックの内部を表している。

1.5.1 CPU

CPU は、バスを通してメモリは入出力と接続される。コンソールからの Stop 信号が入力されている間は CPU は命令実行を停止する。CPU が Halt 機械語命令を実行した場合は、コンソールに Halt 信号を出力する。

1.5.2 コンソール

コンソールの SW や LED, SPK を接続するブロックである。

コンソールは、CPU が命令実行を停止している間だけ機能する。CPU が命令実行を開始するとコンソールの表示は変化しなくなる。その間は、プログラムから入出力装置として使用することができる。

1.5.3 割り込みコントローラ

マイクロ SD ホストコントローラ、I/O ポート、タイマー、シリアル I/O、RN4020 アダプタから発生する合計 10 種類の割り込み信号と、CPU、MMU から発生する 6 種類の例外信号を入力し、CPU

に割り込み（例外）の発生を知らせる。CPU が割り込み（例外）認識サイクルに入ったらバスを通して割り込み（例外）番号を CPU に伝える。

割り込み（例外）は入力信号が '0' から '1' に変化する際に発生する。同じ種類の割り込み（例外）が複数回発生する場合は、入力信号を一旦 '0' に戻す必要がある。

1.5.4 メモリ

MMU を通してバスに接続される。容量は 64KiB (32KiW) である。16 ビット単位、または、8 ビット単位のデータを読み書きできる。16 ビット単位でアクセスする場合は偶数アドレスを指定する必要がある。マイクロ SD ホストコントローラやコンソールは、バスとは別の配線でメモリに接続されおり、DMA (Direct Memory Access) 方式でメモリをアクセスすることができる。

1.5.5 MMU (Memory Management Unit)

ページング方式の MMU である。8 エントリの TLB (Translation Look-aside Buffer) を内蔵し、8 ビットのページ番号を 8 ビットのフレーム番号に変換する。TLB の管理は OS が行う前提で設計されており、ページテーブルの検索機能は持っていない。OS は I/O 命令で MMU の設定を変更できる。

1.5.6 マイクロ SD ホストコントローラ

マイクロ SD を SPI モードに切り換え、512 バイトのセクタ単位で読み書きを行う制御をハードウェアで行う。CPU は、LBA (Logical Block Addressing) 方式で表現する 32 ビットのセクタアドレス、メモリ上の 512 バイトのバッファのアドレスをレジスタに書き込み、開始を指示するだけでセクタの読み書きができる。

1.5.7 I/O ポート

プリント基板上的の入出力ポートと接続される。8 ビット入力、8 ビット出力が基本であるが、4 ビット入力、12 ビット出力に切り換えることもできる。また、ハードウェアによるシリアル・パラレル変換機能を持つ SPI ポートとして使用することもできる。更に、入力ポートの状態に変化があった時、割り込みを発生する機能も持つ。

1.5.8 タイマー

1 ミリ秒単位で周期を設定可能な独立した 2 本のインターバルタイマーである。割り込みを発生することができる。

1.5.9 シリアル I/O (SIO)

調歩同期方式の 9,600 ボーのシリアル通信回路である。プリント基板上的の FT232RL と接続するもの、TeC の SIO と接続するものの二つある。

1.5.10 TeC アダプタ

MUX1, DMUX を通して TeC のコンソール入力に接続されている。このアダプタを通して TeC のコンソールを操作することができる。

1.5.11 RN4020 アダプタ

Bluetooth モジュール (RN4020) を接続する回路である。調歩同期方式 115,200 ボーのシリアル通信回路と、RN4020 の一部の外部ピンを操作・監視する回路を内蔵している。

第 2 章

TeC7 の操作方法

TeC モード, DEMO1 モード, DEMO2 モードでの操作方法是「TeC 教科書」^{*1} の第 4 章に詳しく説明されているので, ここではモードを切り換えるジャンパーの設定方法と, TaC モードでの操作方法だけを説明する.

2.1 ジャンパの設定方法

ジャンパはプリント基板中央下 (図 1.1 参照) に配置された小さな部品である. 基板側に四本のジャンパピンが正方形に配置されている. 隣り合った二本のジャンパピンをジャンパプラグで導通させることにより, TeC7 の動作モードを設定する. ジャンパの回路図とジャンパプラグの設定位置を図 2.1 に示す. ジャンパピンの 1, 2 番が FPGA 内部の MODE ブロック (図 1.2 参照) に接続されている. なお, TeC7 がジャンパの設定を読み取るのは電源投入時だけである.

TeC モード ジャンパピンの 2 番を GND に接続する.

TaC モード ジャンパピンの 1 番を GND に接続する.

DEMO1 モード ジャンパピンの 1, 2 番の両方を解放する.

DEMO2 モード ジャンパピンの 1 番と 2 番を接続する.

RESET (RN4020 のリセット) ジャンパピンの 1, 2 番の両方を GND に接続する.

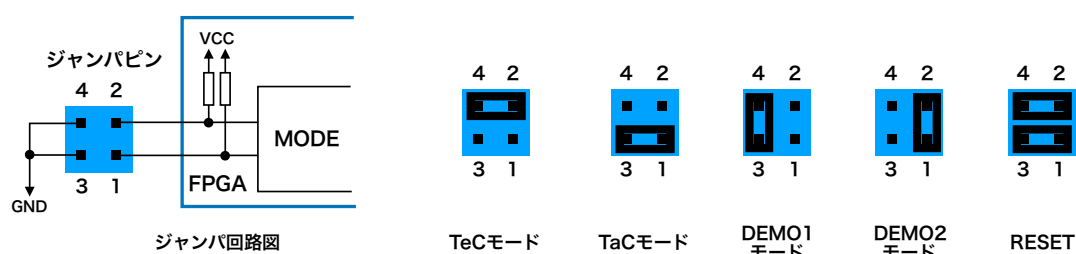


図 2.1 ジャンパ

^{*1} <https://github.com/tctsigemura/TecTextBook/raw/master/tec.pdf>

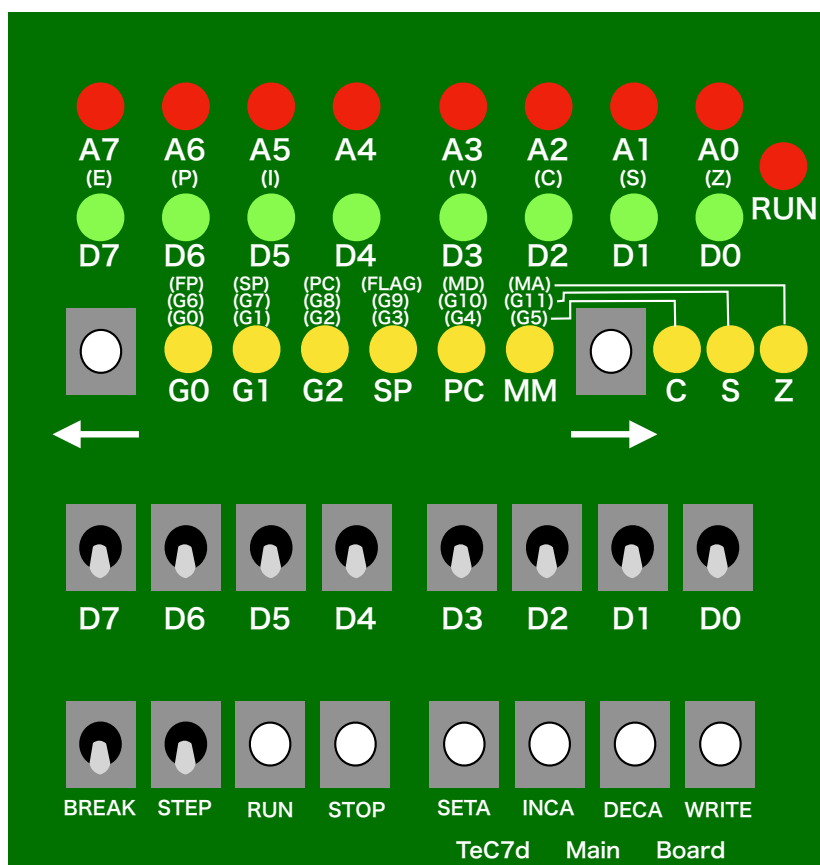


図 2.2 TeC7d のコンソールパネル

2.2 コンソールのランプやスイッチ

TaC はリセットされると自動的に IPL プログラムの実行を開始する。プログラム実行中は TaC のコンソールが操作不能になる。プログラムを停止しないとコンソールは使用できない。

図 2.2 にコンソールパネルの略図を示す。TeC モードと TaC モードで役割が変化するランプには、カッコ書きの小さな文字で TaC モードでの役割が表示してある。例えば D7 ランプは、TaC モードではフラグの割り込み許可ビット E を表示することがあるので、“(E)” の表示がしてある。

2.2.1 アドレスランプ・データランプ

TaC のアドレスやデータは 16 ビットなので、アドレスランプとデータランプを合わせた 16 個の LED で表示する。アドレスとデータを同時に表示することはできない。

2.2.2 ロータリースイッチ

左右矢印の押しボタンでアドレス・データランプに表示するものを切り換える。TaC は CPU レジスタを 14 本持っているので、G0, G1, G2, SP, PC, MM の六つのランプで選択中のものを表現できない。そこで、C, S, Z ランプと組合せて選択中のものを表す。C ランプが点灯中は、六つのランプの上側に三行のカッコ書きで表示してあるものから、一番下の行を読むと何を選択しているか分かる。同様に S ランプが点灯中は中央の行を読めば良い。同様に Z ランプが点灯中は一番上の行を読めば良い。

CPU レジスタの他に、PC, FLAG, MD (Memory Data), MA (Memory Address register) が選択できる。MA は、表示や操作の対象となるメモリのアドレスを記憶しているコンソールのレジスタである。TeC と異なりフラグの状態もデータランプに表示される。プリント基板上でデータランプ上側に印刷されたカッコ書きの表示は、フラグの名前を表している。

2.2.3 データスイッチ

D7 から D0 のトグルスイッチは、8 ビットのデータやアドレスを入力するために使用する。TaC のデータやアドレスは 16 ビットなので二回に分けて入力する。

2.2.4 プログラム実行に使用するランプとスイッチ

RUN ランプは CPU がプログラム実行中に点灯する。BREAK, STEP スイッチと RUN, STOP ボタンはプログラムの実行開始・停止などを指示するために使用する。

2.2.5 データ書き換えに使用するスイッチ

WRITE ボタンを押すと、ロータリースイッチで選択しているものにデータが書き込まれる。SETA, INCA, DECA は MA の値を操作するために使用する。

2.3 操作手順

以下の解説は、特に明示しない限り TaC モードでの操作方法の説明である。

2.3.1 リセット

電源投入時に TeC と TaC の両方がリセットされる。その後は、図 1.1 左下の RESET ボタンを押してリセットすることができるが、TeC7 のモードによってリセットの条件が異なる。

TaC モード RESET ボタンを押すと TaC だけがリセットされる。

他のモード RESET ボタンを押すと通常は TeC だけがリセットされる。しかし、SETA ボタンを押した状態で同時に RESET ボタンを押すと TeC と TaC の両方がリセットされる。

2.3.2 CPU レジスタや PSW の表示と書き換え

次の手順で CPU レジスタや PSW の表示と書き換えを行う。

1. ロータリースイッチを操作して目的のレジスタ等を選択する。この時点で、レジスタ等の内容がアドレス・データランプに表示される。
2. データスイッチにデータの上位 8 ビットをセットする。
3. WRITE ボタンを押す。
4. データスイッチにデータの下位 8 ビットをセットする。
5. WRITE ボタンを押す。

2.3.3 メモリの表示と書き換え

メモリは 8 ビット (1 バイト) 毎にアドレス付けがされているが、コンソールからは 16 ビット (2 バイト=1 ワード) 単位で操作する。コンソールでは、常に、偶数アドレスを用いる。

アドレスの設定

メモリのデータを読み書きするためにはアドレスを指定する必要がある。まず、MA にアドレスを設定する。メモリは 2 バイト単位で操作するので必ず偶数アドレスを用いる。ユーザが奇数アドレスを

入力できないようにしてある。そのため、目的アドレスの上位 8 ビットの LSB が 1 の場合、一度目の SETA ボタンの操作時点では LSB が 0 と表示されるが正常である。

MA にアドレスを設定する手順は次の通りである。

1. ロータリースイッチを MA の位置に合わせる。
2. データスイッチにアドレスの上位 8 ビットをセットする。
3. SETA ボタンを押す。(WRITE ボタンではない)
4. データスイッチにアドレスの下位 8 ビットをセットする。
5. SETA ボタンを押す。

アドレスの変更

ロータリースイッチが MD または MA の位置にある時、INCA、DECA ボタンを押すと MA のアドレスを増やしたり減らしたりできる。アドレスが偶数になるよう増減は 2 刻みになる。

データの書き込み

MA に目的のアドレスを設定した後、以下の操作をすることでメモリにデータを書き込む。なお、データを書き込んでも MA は自動的に増加しないので注意が必要である。

1. ロータリースイッチを MA または MD の位置に合わせる。
2. データスイッチにデータの上位 8 ビットをセットする。
3. WRITE ボタンを押す。
4. データスイッチにデータの下位 8 ビットをセットする。
5. WRITE ボタンを押す。

2.3.4 プログラムの停止・実行・デバッグ

OS カーネルの内部まで、ステップ実行などを用いてデバッグすることができる。

プログラムの停止

STOP ボタンを押すとプログラムが停止する。OS が動作中でも CPU とタイマー (3.5.1 参照) が停止し、コンソールから CPU レジスタや PSW、メモリの値を参照したり変更したりすることができる。

特定番地からの実行

PC の値を変更すれば任意アドレスのプログラムを実行できる。

1. プログラムの実行開始番地を PC にセットする。
2. BREAK、STEP スイッチが下に倒れていることを確認する。
3. RUN ボタンを押す。プログラム実行中は RUN ランプが点灯している。

ステップ実行・ブレークポイントを用いたデバッグ

STEP スイッチを上を倒すと、機械語命令を一つ実行し終わる毎に CPU が停止するステップ実行モードになる。ステップ実行モードでは、RUN ボタンを押す度にプログラムを一命令ずつ実行する。

BREAK スイッチを上、STEP スイッチは下に倒すと、ブレークポイント実行モードになる。ブレークポイント実行モードでは、CPU がメモリの MA 番地をアクセスすると CPU が停止する。MA 番地は MMU が出力する物理アドレスではなく、CPU が出力する論理アドレスによるものである。命

令フェッチとデータの読み書きのどちらでも CPU が停止する点が TeC と異なる。

注意

BREAK, STEP スイッチが上に倒れていると、リセットしてもすぐに CPU が停止してしまうので OS が起動しない。通常は、BREAK, STEP スイッチを必ず下に倒しておくこと。

OS デバッグの例

以下では、open システムコールにバグが疑われる時に、TacOS の内部で open() 関数が呼び出された時に CPU を停止する例を示す。

1. TacOS の配布物を展開し `kernel.bin` を作成する。
2. 同時に作成された `kernel.map` ファイルから `_open` のアドレスを確認する。
3. TacOS を起動する。
4. STOP ボタンを押して CPU を一旦停止する。
5. コンソールを操作し MA に `_open` のアドレスを設定する。
6. BREAK スイッチを上倒してから RUN ボタンを押す。
7. `open()` が呼ばれた時点で CPU が停止する。
8. コンソールを操作しバグの原因を調査する。

第 3 章

TaC のアーキテクチャ

TaC のアーキテクチャは「TeC 教科書」^{*1} で詳しく説明されているので、ここでは TaC のアーキテクチャについて簡単に説明する。

3.1 CPU の概要

TaC で使用できるデータの形式、アドレス空間、実行モード、CPU 内部のレジスタ構成、機械語命令、割込みと例外について説明する。

3.1.1 データ形式

図 A.1 の「データ形式」に TaC が扱うことができるデータを示す。16 ビットの整数データと、16 ビットのアドレスデータの他に、8 ビットの整数データを扱うことができる。16 ビットのデータは CPU の内部でもメモリや I/O でも使用できる。8 ビットデータはメモリ上の配列に格納することだけを想定している。

3.1.2 アドレス空間

図 A.1 の「メモリ空間」、「I/O 空間」に TaC のメモリと I/O のアドレス空間を示す。メモリ空間も I/O 空間もバイト単位でアドレスが割り付けてある。しかし、I/O 空間をバイト単位でアクセスする機械語命令を現在の TaC は備えていない^{*2}。メモリや I/O の 16 ビットデータにアクセスする場合は偶数番地を用いる。8 ビットデータはメモリの読み書きだけに使用できる。メモリの 8 ビットデータにアクセスする場合は、CPU レジスタの下位 8 ビットだけが使用される。

3.1.3 実行モード

TaC は「特権モード」、「ユーザモード」、「I/O 特権モード」の三つの実行モードを持っている。

特権モード 全ての機械語命令が実行できるモードである。OS カーネルは特権モードで実行される。

ユーザモード 実行モードを変更したり、ハードウェアの状態を変更したりする**特権命令**を実行することができない。通常、ユーザプログラムはユーザモードで実行される。

I/O 特権モード IN, OUT 機械語命令が実行できるユーザモードである。入出力ポートに接続したオ

^{*1} <https://github.com/tctsigemura/TecTextBook/raw/master/tec.pdf>

^{*2} 以前の TaC は I/O 空間もバイト単位でアクセスできたので、バイト単位でアドレスが割り付けられている。

プシヨンのハードウェア^{*3}を使用するアプリケーションを実行するために用意されている。

3.1.4 CPU レジスタと PSW

図 A.1 の「レジスタ構成」に CPU 内部のレジスタなどを示す。レジスタはどれも 16 ビット幅である。
CPU レジスタ

CPU レジスタは、汎用の G0 (General register 0) から G11, フレームポインタとして使用する FP (Frame Pointer), 特権モード用のスタックポインタ SSP (System Stack Pointer), ユーザモード (I/O 特権モードも含む) 用のスタックポインタ USP (User Stack Pointer) からなる。これらは全て計算用にもアドレス用にも使用できる。FP, SSP, USP は、以下に説明する特別な意味も持っている。

フレームポインタ (Frame Pointer)

フレームポインタ (FP) は CPU レジスタの一つである。フレームポインタ相対アドレッシングモードで使用できる。このアドレッシングモードを用いると、スタックフレーム内のローカル変数や関数引数へ、1 ワード (2 バイト) の機械語命令でアクセスできる。

スタックポインタ (Stack Pointer)

スタックポインタ (SP) も CPU レジスタの一つである。TaC は特権モード用 (SSP), ユーザモード (I/O 特権モード含む) 用 (USP) の二本のスタックポインタを持っている。SSP は特権モードで SP の位置にマップされ、OS カーネル用のスタックポインタとして使用される。USP はユーザモード (I/O 特権モード含む) で SP の位置にマップされ、ユーザプログラムのスタックポインタとして使用される。USP は G14 として常時マップされており、特権モードでも USP をアクセスすることができる。

PSW (Program Status Word)

PSW は PC (Program Counter) と FLAG からなる。FLAG には、計算結果で変化する V (oVerflow), C (Carry), S (Sign), Z (Zero) と、割込み許可 E (Enable interrupt), 特権モード P (Privilege), I/O 特権モード I (I/O Privilege), ユーザ定義 U (User defined) の各ビットがある。

FLAG は G15 として普通の機械語命令で扱うことも可能であるが、ユーザモードでは E, P, I の各ビットは変化しない。

3.1.5 機械語命令

図 A.4 に TaC の機械語命令の一覧表を示す。HALT は特権モードでしか使用できない**特権命令**である。IN, OUT は特権モードと I/O 特権モードで使用できる命令である。これらの命令を非特権モードで実行すると特権違反例外が発生する。SVC 命令はシステムコールを発行するために SVC 例外を発生する。

ほとんどの転送命令と計算命令で 8 種類のアドレッシング・モードが使用できる。Direct, Indexed, Immediate の三つのアドレッシング・モードを使用する場合は 2 ワードの機械語命令になる。他のアドレッシング・モードの場合は 1 ワード命令である。

Byte Register Indirect アドレッシング・モードだけが、メモリの 8 ビットデータをアクセスする。Byte Register Indirect アドレッシング・モードの ST 命令は、CPU レジスタの下位 8 ビットをメモリに書き込む。これら以外の命令は、メモリから読み出した 8 ビットデータの上に 00h を付加した 16

^{*3} このようなハードウェアは OS によってサポート・管理されない。

表 3.1 割込みの種類と意味

| 割込み・例外 | 意 味 |
|--------------|--|
| 0 Timer0 | ハードウェアタイマー 0 に設定された時刻になった。 |
| 1 Timer1 | ハードウェアタイマー 1 に設定された時刻になった。 |
| 2 RN4020 受信 | Bluetooth モジュールから 1 バイトのデータを受信した。 |
| 3 RN4020 送信 | Bluetooth モジュールへ 1 バイトのデータを送信し終えた。 |
| 4 FT232RL 受信 | USB シリアル変換 IC から 1 バイトのデータを受信した。 |
| 5 FT232RL 送信 | USB シリアル変換 IC へ 1 バイトのデータを送信し終えた。 |
| 6 TeC 受信 | TeC から 1 バイトのデータを受信した。 |
| 6 TeC 送信 | TeC へ 1 バイトのデータを送信し終えた。 |
| 8 マイクロ SD | マイクロ SD のホストコントローラがコマンドを実行し終えた。 |
| 9 PIO | 入出力ポートの監視中のビットに変化があった。 |
| 10 TLB ミス | MMU 有効時に TLB に必要なエントリが見つからない。 |
| 11 メモリ保護違反 | 奇数アドレスでワードデータをアクセスした。または、 ページの保護モード (RWX) に一致しないアクセスがされた。 |
| 12 ゼロ除算 | 割り算機械語命令で「÷ 0」が実行された。 |
| 13 特権違反 | 不適切な実行モードで特権命令が実行された。 |
| 14 未定義命令 | TaC の機械語として解釈できない命令を実行した。 |
| 15 SVC | SVC 機械語命令が実行された。 |

ビットデータを使用する。

3.1.6 割込み (Interrupt)・例外 (Exception)

TaC はベクタ方式 (ベクタは FFE0h 番地～) の割込み・例外機構を備えている。割込み・例外の種類は表 3.1 に示す 16 種類である。ゼロ除算や特権違反のようなソフトウェアに起因する割込みを「例外 (Exception)」と呼ぶ。「TLB ミス」から「SVC」までの 6 種類が「例外」それ以外が「割込み」である。「割込み」の許可と禁止は FLAG の E ビットを操作することで行う。「例外」の発生は禁止できない。

割込みが発生すると次の順で割込み処理が行われる。

1. CPU 内部の一時レジスタに *FLAG* のコピーが作られる。
2. FLAG が変更され、割込みが禁止 (E=0) の特権モード (P=1) になる。
3. PC と *FLAG* のコピーが順にカーネルスタックに PUSH される。
4. PC に割込み (例外) ハンドラの開始番地がロードされ、ハンドラの実行が開始される。

3.2 メモリマップと I/O マップ

メモリ空間と I/O 空間は 8 ビット毎にアドレス付けされている。メモリは 8 ビットデータ、16 ビットデータのどちらも読み書きできる。I/O は 16 ビットデータの読み書きしかできない。メモリの場合

は機械語命令のアドレッシング・モードによって、8ビットデータと16ビットデータの区別をする。16ビットデータは偶数アドレスを指定してアクセスしなければならない。

3.2.1 メモリ空間

図 A.2 の「メモリマップ」に TaC のメモリマップを示す。TaC のメモリ空間は 0000h から FFFFh の 64KiB である。16ビットデータは偶数アドレスからの2バイトに配置され、偶数アドレスを指定してアクセスする。8ビットデータにアクセスするには、Byte Register Indirect モードを用いる。その他のアドレッシング・モードは、16ビットデータをアクセスするために用いる。

リセット時に、E000h から FFFFh に IPL (ROM) が配置される。TaC モードでは、IPL はマイクロ SD から OS を読み出して起動する。その他のモードでは、IPL が TeC の通信を中継する等の機能を果たす。IPL は OS を読みだしたら IPL (ROM) を切り離しメモリ空間全体を RAM にした後、OS に制御を渡す。IPL (ROM) が切り離された後、FFE0h から FFFFh は割込みベクタ領域になる。16種類の割込み・例外ハンドラの入口番地を OS がセットする。

3.2.2 I/O 空間

図 A.2 の「I/O マップ」に TaC の I/O マップを示す。TaC の I/O 空間は 00h から FFh の 128 ワードである。I/O 空間のアドレス幅は8ビットだが、IN, OUT 機械語命令では I/O アドレスが16ビットで表現される。I/O アドレスの上位8ビットは 00h になるようにする。上位8ビットが 00h 以外になった場合の動作は保証されない。メモリ空間と異なり16ビットデータの読み書きしかできない。

3.3 MMU (Memory Management Unit)

ページング方式の MMU が使用できる。MMU が働くのはユーザモードで実行中だけである。MMU の制御は図 A.2 の「I/O マップ」に示すポートを IN, OUT 機械語命令で操作して行う。

| 番地 | IN | | OUT | |
|-----|-----------------|----------------|-----------------|----------------|
| | 上位バイト | 下位バイト | 上位バイト | 下位バイト |
| 80h | 00 | TLB[0] 上位 8bit | - | TLB[0] 上位 8bit |
| 82h | TLB[0] 下位 16bit | | TLB[0] 下位 16bit | |
| 84h | 00 | TLB[1] 上位 8bit | - | TLB[1] 上位 8bit |
| 86h | TLB[1] 下位 16bit | | TLB[1] 下位 16bit | |
| ... | ... | | ... | |
| 9Ch | 00 | TLB[7] 上位 8bit | - | TLB[7] 上位 8bit |
| 9Eh | TLB[7] 下位 16bit | | TLB[7] 下位 16bit | |
| A0h | 00 | 00 | - | IPL 切離し |
| A2h | 違反アドレス | | - | MMU 有効化 |
| A4h | 00 | 違反原因 | - | - |
| A6h | 00 | ページ番号 | - | - |

3.3.1 違反アドレス

メモリ保護違反が発生した時、原因となった論理アドレスが記録される。

3.3.2 違反原因 (0000 00BV)

メモリ保護違反が発生した時、奇数アドレスを用いたワードアクセス (Bad Address) の場合 B ビットが '1' になる。ページの保護モード違反 (Memory Violation) の場合 V ビットが '1' になる。これら

のビットは CPU が IN 命令で A4h 番地を読むとクリアされる。

3.3.3 ページ番号

ページ番号が TLB でヒットしなかった場合 TLB ミス例外が発生する。その際、例外の原因となったページ番号が記録される。

3.3.4 IPL 切離し (0000 000I)

I に '1' を書き込むと、物理メモリ空間最後の 8KiB に配置された IPL (ROM) が切り離され RAM に置き換わる。これによりメモリ空間 64KiB 全てが RAM になる。通常、切り離しの操作は IPL 自身が OS をロードした後で自動的に行う。IPL (ROM) がマップされている時でもコンソールからは RAM が見えるので、IPL を最後まで実行していない場合は注意が必要である。

3.3.5 MMU 有効化 (0000 000E)

E に '1' を書き込むと MMU が有効になる。MMU が有効になると CPU の実行モードが「I/O 特権モード」、「ユーザモード」の時、ページからフレームへの変換 ($p \rightarrow f$ 変換) が行われる。

3.3.6 TLB (TLB[0]~TLB[7])

TLB はページ番号で検索されフレーム番号を出力し $p \rightarrow f$ 変換を行う。8 エントリの TLB を IN, OUT 機械語命令で参照・操作することができる。8 つの TLB エントリは I/O 空間の 80h 番地から 9Eh 番地までの範囲に配置される。1 つの TLB エントリは 24bit であるので、上位 8bit と下位 16bit に分けてアクセスする。以下では TLB エントリの各ビット (フィールド) の意味を説明する。

| TLB エントリの構成 | | | | | | | | |
|-------------|---------|----------|----|----|----|----|------|--------|
| I/O ポート | 上位 8bit | 下位 16bit | | | | | | |
| ビット番号 | 23 - 16 | 15 | 14 | 13 | 12 | 11 | 10-8 | 7 - 0 |
| ビットの意味 | ページ番号 | V | U | U | R | D | RWX | フレーム番号 |

ページ番号 (p)

エントリが管理するページの番号を設定する。TLB エントリをページ番号で検索し、該当ページを管理するエントリが見つからない場合、TLB ミス例外が発生する。

V(Valid)

エントリが有効かどうかを表す。TLB を検索する際、 $V = 0$ のエントリは無視される。

U(Undefined)

ハードウェアでは使用していないビットである。単なる 1 ビットの記憶装置として OS が自由に使用できる。

R(Reference)

OS がクリアしたあとユーザプログラムがページを参照すると '1' になる。OS がページの参照に関する統計情報を取得するために使用する。

D(Dirty)

OS がクリアしたあとユーザプログラムがページに書き込みを行うと '1' になる。OS がページをスワップアウトする必要があるか判断するために使用する。

RWX(Read/Write/eXecute)

ページの保護モードを3ビットの組み合わせで表現する。機械語セグメントには101、データセグメントやスタックセグメントには110、読み出し専用のデータセグメントには100をセットする。CPUが保護モードで許可されない種類のアクセスを行った場合、メモリ保護違反例外が発生する。

フレーム番号 (f)

ページ番号に対応するフレーム番号を設定する。MMUはページ番号がヒットしたエントリのフレーム番号を変換結果として出力する。

3.4 IPL プログラム

TaC はリセットされると自動的に IPL プログラム^{*4}の実行を開始する。IPL の第一の役割は、マイクロ SD から OS を読み出し起動することである。しかし、TeC7 の動作モード (1.3.2 参照) によっては、TeC の補助 (1.3.3 参照) を行う。以下では動作モード毎に IPL の役割を説明する。

3.4.1 TeC モード

USB シリアル変換 IC (FT232RL) から受信したデータを TeC の SIO へ送信する。また、TeC の SIO から受信したデータを FT232RL に送信する。FT232RL は PC と USB シリアル接続が確立していればデータを PC に送るが、確立していない場合はデータを無視する。このようにして、TeC のシリアル通信を USB を経由して PC に中継する。

Bluetooth モジュール (RN4020) はシリアル通信でデータだけでなくコマンドも受け付ける。Bluetooth 接続が確立していない状態で TaC が RN4020 に何か送信すると、コマンドとして解釈され不具合が生じる可能性がある。そこで、Bluetooth 接続が確立されている場合だけ TeC の通信を中継する。このようにして TeC が知らない間に、TeC のシリアル通信先が切り換わる。

USB シリアルまたは Bluetooth を通して PC から受信したデータに “\033TWRITE\r\n” の文字列を見つけると、TWRITE プログラムの通信だと判断する。TWRITE プログラムが送ってきた TeC の機械語プログラムを受信し、TeC のコンソールを操作して TeC のメモリに書き込む。

なお、SETA ボタンが押された状態で TeC7 がリセットされた場合は、OS (“kernel.bin”) を読み込み制御を OS に移す。この場合は、コンソールから TeC が操作できるが、裏で TaC が OS を起動した状態になる。TaC の OS 上で TeC のプログラムを開発する場合等に使用することを想定している。

3.4.2 TaC モード

マイクロ SD スロットを確認し、カードが挿入されていれば OS を読み込んで起動する。OS は、マイクロ SD カードの FAT16 ファイルシステムの “\kernel.bin”^{*5} ファイルに格納されている。IPL は OS に制御を移す前に、自身が格納された ROM (E000h - FFFFh) を切り離し RAM に切り換える。

なお、リセット時に SETA ボタンが押されていた場合は、“\kernel.bin” ファイルの代わりに “\kernel0.bin” ファイルから OS を読み込む。カーネルのデバッグ中でも簡単に OS を起動できる。

^{*4} IPL のソースコードは <https://github.com/tctsigemura/TeC7/tree/master/TaC/Ipl> に公開されている。

^{*5} .bin ファイル形式については、「Util--解説書」(<https://github.com/tctsigemura/Util--/raw/master/doc/umm.pdf>) の付録 B 「ファイルフォーマット」を参照のこと。

3.4.3 DEMO モード

「DEMO1 モード」, 「DEMO2 モード」では, IPL が RN4020 と FT232RL の通信を中継する. USB シリアルで接続した PC から, RN4020 の初期設定を行うことができる. 工場出荷時に RN4020 のシリアル通信は 115,200 ボーに設定されているが, FT232RL のデフォルトは 9,600 ボーである. TaC がボーレート変換器の役割を果たす. なお, FT232RL 及び RN4020 のボーレートは変更してはならない.

3.4.4 RESET

RN4020 を工場出荷時の状態に戻す. 通常はシリアル通信でコマンドを送ることで RN4020 を初期化できる. しかし, 間違ってボーレートを変更したり, ハードウェアフロー制御を有効にしたりすると, コマンドを送ることができなくなることがある. そのような場合に, この機能を使用する.

RN4020 は, 電源投入後 5 秒以内に WAKE_HW ピンを 3 回以上フリップすることで, 工場出荷時の状態に戻る. ジャンパーを RESET の設定にして TeC7 に電源を投入すると, TaC がこの操作を行う.

3.5 周辺装置

TaC は, 図 1.4 に示したように, コンソール, MMU, 割り込みコントローラ, タイマー, 入出力装置などの周辺装置を持っている. これらには, 図 A.2 の I/O マップに掲載されたポートを通して, IN, OUT 機械語命令でアクセスする. 以下では, 周辺装置の使用方法を解説する. なお, 特別な説明がないレジスタ等はリセット時に '0' で初期化される.

3.5.1 タイマー

Timer0, Timer1 の 2 チャンネルのインターバルタイマーが使用できる. タイマーは 16 ビットのカウンタと 16 ビットの周期レジスタ等から構成される.

| 番地 | IN | | OUT | |
|-----|-------------|-------|---------------|-------|
| | 上位バイト | 下位バイト | 上位バイト | 下位バイト |
| 00h | Timer0 カウンタ | | Timer0 周期レジスタ | |
| 02h | Timer0 フラグ | | Timer0 制御 | |
| 04h | Timer1 カウンタ | | Timer1 周期レジスタ | |
| 06h | Timer1 フラグ | | Timer1 制御 | |

カウンタ カウンタの現在値を読み出すことができる. タイマー動作中は 1ms 毎にカウントアップされ, カウンタの値と周期レジスタの値が一致するとゼロにリセットされる. リセットされる時, CPU に割り込みを発生する. コンソールから CPU を停止している間はカウンタも停止する.

周期レジスタ 周期レジスタに書き込んだ値によって, カウンタがリセットされる周期が決まる. 単位はミリ秒である.

フラグ (F0000000 00000000) カウンタの値と周期レジスタの値が一致すると F に '1' がセットされる. 同じチャンネルのカウンタまたはフラグが読み出されるとリセットされる.

制御 (I0000000 0000000S) I が割り込み許可ビット, S がカウンタのスタート/ストップ ('1'/'0') を制御する. 制御ワードに書き込みを行うとカウンタがリセットされるので, カウントは必ずリセット状態から開始される.

3.5.2 FT232RL (シリアル I/O)

USB シリアル変換 IC (FT232RL) を通して PC と通信を行うことができる。変調速度は 9,600 ボーに固定されており変更することはできない。送信・受信の両方で割り込みを発生することができる。

| 番地 | IN | | OUT | |
|-----|-------|-------|-------|-------|
| | 上位バイト | 下位バイト | 上位バイト | 下位バイト |
| 08h | 00 | 受信データ | - | 送信データ |
| 0Ah | 00 | ステータス | - | 制御 |

受信データ FT232RL から受信した 1 バイトのデータを読み出す。

送信データ FT232RL へ送信する 1 バイトのデータを書き込む。

ステータス (TR00 0000) 送信回路に送信データを書き込み可能なとき T が '1' になる。受信回路に受信済みデータがあり読み出し可能なとき R が '1' になる。

制御 (TR00 0000) T を '1' にすると次の送信データが書き込み可能になる度に割り込みが発生する。R を '1' にすると次の受信データが読み込み可能になる度に割り込みが発生する。

3.5.3 TeC (シリアル I/O)

TeC とシリアルデータ通信ができる。変調速度は 9,600 ボーに固定されており変更することはできない。送信・受信の両方で割り込みを発生することができる。

| 番地 | IN | | OUT | |
|-----|-------|-------|-------|-------|
| | 上位バイト | 下位バイト | 上位バイト | 下位バイト |
| 0Ch | 00 | 受信データ | - | 送信データ |
| 0Eh | 00 | ステータス | - | 制御 |

受信データ TeC から受信した 1 バイトのデータを読み出す。

送信データ TeC へ送信する 1 バイトのデータを書き込む。

ステータス (TR00 0000) 送信回路に送信データを書き込み可能なとき T が '1' になる。受信回路に受信済みデータがあり読み出し可能なとき R が '1' になる。

制御 (TR00 0000) T を '1' にすると次の送信データが書き込み可能になる度に割り込みが発生する。R を '1' にすると次の受信データが読み込み可能になる度に割り込みが発生する。

3.5.4 マイクロ SD ホストコントローラ

マイクロ SD とメモリの間でセクタ単位の読み書きができる。

| 番地 | IN | | OUT | |
|-----|-----------|-------|-----------|-------|
| | 上位バイト | 下位バイト | 上位バイト | 下位バイト |
| 10h | 00 | ステータス | - | 制御 |
| 12h | メモリアドレス | | メモリアドレス | |
| 14h | セクタアドレス上位 | | セクタアドレス上位 | |
| 16h | セクタアドレス下位 | | セクタアドレス下位 | |

ステータス (IE00 0000) ホストコントローラの状態を表す。I はアイドル状態を表す。E はエラーが発生したことを表す。

制御 (E000 0IRW) I に '1' を書き込むと、マイクロ SD を SPI モードに切り換え使用できるように初期化する動作を開始する。R に '1' を書き込むとマイクロ SD から 1 セクタ読み込む動作を開始する。W に '1' を書き込むとマイクロ SD に 1 セクタ書き込む動作を開始する。E を '1' にすると上記の動作が完了したとき割り込みが発生するようになる。

メモリアドレス セクタから読み込んだデータ、または、セクタに書き込むデータを格納するバッファのメモリアドレスを設定する。ホストコントローラは CPU の力を借りることなく、メモリとマイクロ SD の間でデータの転送を行う。バッファサイズは 512、アドレスは偶数でなければならない。

セクタアドレス上位 データを読み書きするセクタの LBA (Logical Block Addressing) 方式の 32 ビットのアドレスの上位 16 ビットである。

セクタアドレス下位 LBA 方式の 32 ビットのアドレスの下位 16 ビットである。

3.5.5 入出力ポート他

TeC7 の入出力ポート^{*6}にパラレルデータを入出力する。

| 番地 | IN | | OUT | |
|-----|-------|-------|-------|----------|
| | 上位バイト | 下位バイト | 上位バイト | 下位バイト |
| 18h | 00 | 入力ポート | - | 出力ポート |
| 1Ah | 00 | 00 | - | ADC 参照電圧 |
| 1Ch | 00 | 00 | - | 出力ポート上位 |
| 1Eh | 00 | モード | - | - |

入力ポート 入出力ポートの I7～I0^{*7}の 8 ビットの入力値を読み取る

出力ポート 入出力ポートの O7～O0 の 8 ビットに出力する値を設定する。

ADC 参照電圧 プリント基板上の AD コンバータ回路の参照電圧を決定する。TaC モードでは、AD コンバータはソフトウェアで制御する必要がある。リセット時は 0x80 がセットされる。

出力ポート上位 (M000 VVVV) M を '1' にすると入力ポートの I7～I4 が出力ポートに切り換わる。M と同時に書き込んだ VVVV の 4 ビットが、I7～I4 に出力される。

モード (0000 0MMM) TeC7 の動作モード^{*8}を MMM の 3 ビットから知ることができる。MMM の意味は、TeC モード (000)、TaC モード (001)、DEMO1 モード (010)、DEMO2 モード (011)、RN4020 リセット (111) である。

3.5.6 SPI インタフェース

図 3.1 に SPI インタフェースの概略図を示す。入出力ポートの 01 ビットに SCLK、00 ビットに SO を出力し、I6 ビットを SI として入力する SPI インタフェースである。出力の 2 ビットは出力ポートの下位 2 ビットと XOR をとっているので、出力ポートの値で極性を変更することができる。SPI で接続した周辺 LSI が SCLK を誤って認識しないように、出力ポートの値を変更するときは、CS をインアクティブにしなければならない。シフトレジスタにデータが書かれると動作を開始する。(データを受信する際も、シフトレジスタにデータを書き込む。)

^{*6} 図 1.1 参照のこと。

^{*7} 図 1.1 の入出力ポートコネクタ左にピン配置が印刷されている。

^{*8} 詳しくは 1.3.2 を参照のこと。

XOR ここに設定した値は監視するビットと排他的論理和をとるために使用する。

複数のビットを同時に監視する際は、「監視対象ビット全ての論理和をとり、結果が‘0’から‘1’に変化する時に割り込みが発生する。」ことを考慮し、適切な順序で MASK と XOR を操作する必要がある。

3.5.8 RN4020 アダプタ

Bluetooth モジュール（RN4020）を接続するインタフェースである。TeC7a にはない。

| 番地 | IN | | OUT | |
|-----|-------|-------|-------|-------|
| | 上位バイト | 下位バイト | 上位バイト | 下位バイト |
| 28h | 00 | 受信データ | - | 送信データ |
| 2Ah | 00 | ステータス | - | 制御 |
| 2Ch | 00 | 00 | - | コマンド |
| 2Eh | 00 | 接続状況 | - | 接続状況 |

受信データ RN4020 から受信した 1 バイトのデータを読み出す。

送信データ RN4020 へ送信する 1 バイトのデータを書き込む。

ステータス (TR00 0000) 送信回路に送信データを書き込み可能なとき T が‘1’になる。受信回路に受信済みデータがあり読み出し可能なとき R が‘1’になる。

制御 (TR00 0000) T を‘1’にすると次の送信データが書き込み可能になる度に割り込みが発生する。R を‘1’にすると次の送信データが読み込み可能になる度に割り込みが発生する。

コマンド (0000 FHCS) F を‘1’にすると RN4020 と TaC 間のシリアル通信のハードウェアフロー制御が有効になる。H は RN4020 の WAKE_HW ピンを制御する。C は RN4020 の CMD/MLDP ピンを制御する。S は RN4020 の WAKE_SW ピンを制御する。S にはリセット時に‘1’が設定される。

接続状況 (RRRR RRRR) R はリセットされないメモリである。RESET ボタンが押され TaC が再起動しても以前の状態を維持する。C の意味は、TeC7b、TeC7c と TeC7d で異なる。

- TeC7b、TeC7c の場合、C は RESET されない 1 ビットのメモリである。IPL プログラムと OS は RN4020 からの受信データを監視し、BlueTerminal (<https://github.com/tctsigemura/BlueTerminal>) との接続確立・切断が発生したことを判定し C に接続状態を書き込む。
- TeC7d の場合、C は RN4020 の CONNECTION LED ピンの状態を反映する。このビットへの書き込みはできない。（無視される。）

3.5.9 TeC アダプタ

TeC モードで動作中に、TaC のプログラムで TeC のコンソールを操作できる。TaC の IPL が TWRITE の通信内容に応じて TeC を操作するために使用している^{*9}。以下でポートに書き込むビット値は、‘1’がスイッチを上倒した状態、または、ボタンを押した状態を表す。

^{*9} IPL と TWRITE については 3.4 を参照すること。

| 番地 | IN | | OUT | |
|-----|-------|--------|-------|---------|
| | 上位バイト | 下位バイト | 上位バイト | 下位バイト |
| 30h | 00 | データランプ | - | - |
| 32h | 00 | 00 | - | データスイッチ |
| 34h | 00 | 00 | - | 機能スイッチ |
| 36h | 00 | スイッチ状態 | - | 制御 |

データランプ TeC コンソールのデータランプの表示を読み取ることができる。TeC のメモリを読み出す TaC プログラムを作成可能にする。

データスイッチ コンソールのデータスイッチの代わりに TeC に入力する値を設定する。

機能スイッチ (ABCD EFGH) TeC コンソールの一番下の八つのスイッチを操作する。各ビットの意味は下の表の通りである。

スイッチ状態 (0000 00RS) R は RESET ボタン, S は SETA ボタンが押されていることを表す。

制御 (I000 0JKL) I は図 1.2 の MUX1 を操作し, TeC アダプタの機能を有効にするビットである。

J, K, L には, RESET 等のボタンを操作するための値を設定する。各ビットの意味は下の表の通りである。

| ビット | スイッチ |
|-----|--------|
| A | BREAK |
| B | STEP |
| C | RUN |
| D | STOP |
| E | SETA |
| F | INCA |
| G | DECA |
| H | WRITE |
| I | 制御を有効化 |
| J | RESET |
| K | ← |
| L | → |

3.5.10 コンソール

TaC モードでプログラム実行中は, コンソールをプログラムの入出力装置として使用できる。

| 番地 | IN | | OUT | |
|-----|----------|----------|---------|-------|
| | 上位バイト | 下位バイト | 上位バイト | 下位バイト |
| F8h | 00 | データ SW | データレジスタ | |
| FAh | アドレスレジスタ | | - | - |
| FBh | 00 | ロータリー SW | - | - |
| FEh | 00 | 機能レジスタ | - | - |

データ SW データ SW (8 個のトグルスイッチ) の現在の状態を読むことができる。

データレジスタ アドレス・データランプ (合計 16 個の LED) の ON/OFF を制御できる。

アドレスレジスタ 2.2.2 で説明した MA (Memory Address register) の値を読むことができる。

ロータリー SW ロータリースwitchの位置 (G0=0, G1=1 ... MA=17) を読むことができる。

機能レジスタ このポートから WRITE スイッチが押されたことを知ることができる.

付録 A

TaC に関する資料

A.1 データ形式

TaC で使用できるデータの形式と、メモリ空間、I/O 空間、CPU 内部のレジスタ等の構成を図 [A.1](#) に示す.

A.2 メモリマップと I/O マップ

TaC のメモリマップ, I/O マップ, I/O ポート詳細を図 [A.2](#) と図 [A.3](#) に示す.

A.3 機械語命令表

TaC の機械語命令一覧表を図 [A.4](#) に示す.

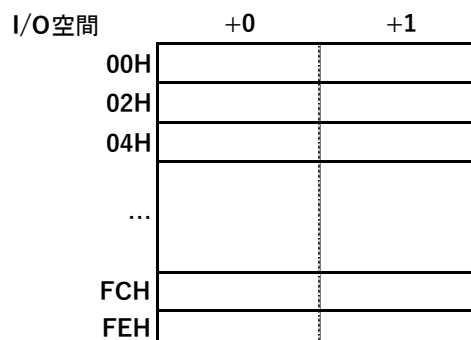
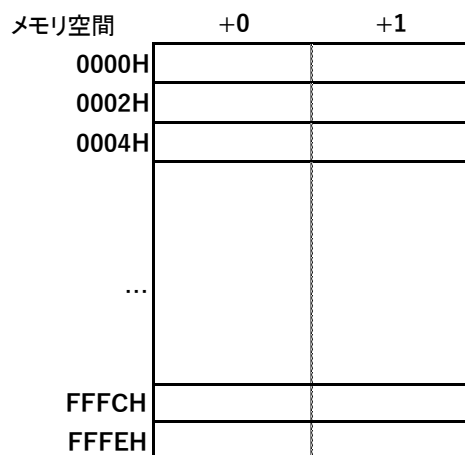
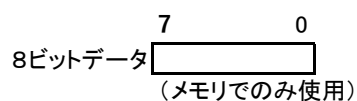
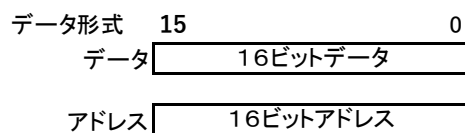
A.4 プリント基板回路図

TaC7d の回路図を図 [A.5](#) に示す.

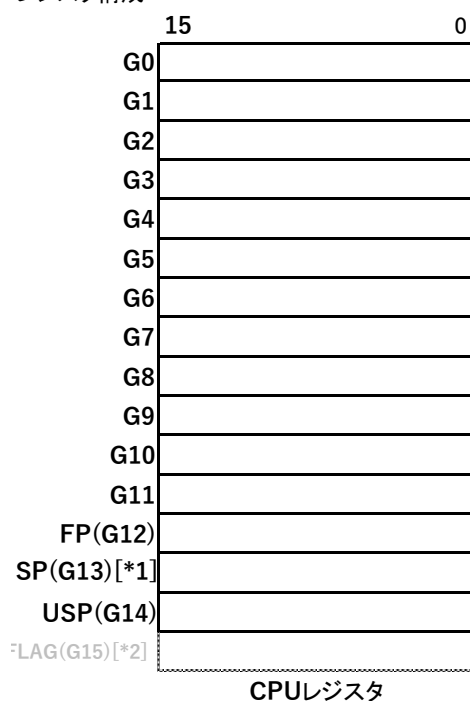
TaC CPU の概要

Ver.10.0.0(TeC7a,b,c,d対応)

2022/3/13



レジスタ構成



*1:SPはカーネルモードではSSP, ユーザモードではUSP

*2:FLAGはCPUレジスタ(G15)として扱うこともできる

*3:Uフラグは単なる1ビットのレジスタ

*4:VフラグはADD, SUB, CMPで有効

*5:CフラグはADD, SUB, CMPで有効, また, SHXXでは1ビットシフトのときだけ有効

図 A.1 TaC CPU の概要

TaCメモリ空間、入出力空間

Ver.10.0.0(TeC7a,b,c,d対応)

2022/3/13

メモリマップ

| | +0番地 | +1番地 | |
|-------|------------|------|---------------------------|
| 0000h | RAM(56KiB) | RAM | ... |
| 0002h | | | |
| 0004h | | | |
| ... | | | |
| DFFEh | RAM(8160B) | RAM | リセット直後はPL(ROM) 割り込みベクタ |
| E000h | | | |
| ... | | | |
| FFDEh | | | |
| FFE0h | | | |
| FFE2h | | | |
| FFE4h | | | |
| FFE6h | | | |
| FFE8h | | | |
| FFEAh | | | |
| FFEC | | | |
| FFEEh | | | |
| FFF0h | | | |
| FFF2h | | | |
| FFF4h | | | |
| FFF6h | | | |
| FFF8h | | | |
| FFFAh | | | |
| FFFCh | | | |
| FFFEh | | | |

※ 1 : 例外 (割り込み禁止の影響を受けない)

I/Oマップ

| +0番地 | | +1番地 | |
|------|---------------------------|-------------------|---------|
| 00h | Timer0(In:現在値/Out:周期) | | タイマー |
| 02h | Timer0(In:フラグ/Out:コントロール) | | |
| 04h | Timer1(In:現在値/Out:周期) | | |
| 06h | Timer1(In:フラグ/Out:コントロール) | | FT232RL |
| 08h | 00H | FT232RL-Data | |
| 0Ah | 00H | FT232RL-Stat/Ctrl | |
| 0Ch | 00H | TeC-Data | TeC |
| 0Eh | 00H | TeC-Stat/Ctrl | |
| 10h | 00H | uSD-Stat/Ctrl | |
| 12h | uSD-MemAddr | | マイクロSD |
| 14h | uSD-BlkAddrH | | |
| 16h | uSD-BlkAddrL | | |
| 18h | 00H | I/Oポート(In/Out) | 入出力ポート |
| 1Ah | 00H | ADC参照電圧(Out) | |
| 1Ch | 00H | 出力ポートHi(Out) | |
| 1Eh | 00H | モード(In) | |
| 20h | 00H | SPI-Data(In/Out) | |
| 22h | 00H | SPI-Stat/Sclk | |
| 24h | 00H | PIO-Mask | |
| 26h | 00H | PIO-Xor | |
| 28h | 00H | RN4020-Data | RN4020 |
| 2Ah | 00H | RN4020-Stat/Ctrl | |
| 2Ch | 00H | RN4020-Cmd | |
| 2Eh | 00H | RN4020-RAM | |
| 30h | 00H | TeC(In:DLed) | |
| 32h | 00H | TeC(Out:DSw) | |
| 34h | 00H | Tec(Out:Fnc) | |
| 36h | 00H | TeC(Ctl) | |
| 38h | 00H | 00H | |
| ... | ... | | |
| 80h | TLB[0]上位8bit | | MMU |
| 82h | TLB[0]下位16bit | | |
| 84h | TLB[1]上位8bit | | |
| 86h | TLB[1]下位16bit | | |
| ... | ... | | |
| 9Ch | TLB[7]上位8bit | | |
| 9Eh | TLB[7]下位16bit | | |
| A0h | b0=IPL切離し(OUT) | | |
| A2h | b0=MMU有効(OUT)/違反アドレス(IN) | | |
| A4h | b1=badAddr,b0=memVio(IN) | | |
| A6h | ページ番号(IN) | | コンソール |
| ... | 空き | | |
| F8h | データレジスタ(Out)/データSW(IN) | | |
| FAh | アドレスレジスタ (IN) | | |
| FCh | 00H | ロータリーSW(IN) | |
| FEh | 00H | 機能レジスタ(IN) | |

出力ポートHi (M000 VVVV)

M (0:入力, 1:出力), VVVV (I7~I4に出力)

RN4020-RAM: リセットの影響を受けない8bitレジスタ

I/Oポート詳細

| 番地 | I/Oポート | ビット | 意味 |
|------|---------------|---------------|--|
| | *-Ctrl(OUT) | TR00 0000 | T=Enable Transmitter Interrupt, R=Enable Reciver Interrupt |
| | *-Stat(IN) | TR00 0000 | T=Transmitter Ready, R=Reciver Ready |
| 02h | Timer0 コントロール | I000 ... 000S | I=Enable Interrupt, S=Start |
| 06h | Timer1 コントロール | I000 ... 000S | I=Enable Interrupt, S=Start |
| 11h | uSD-Ctrl | E000 01RW | E=INT_ENA, I=INIT, R=READ, W=WRITE |
| 11h | uSD-Stat | IE00 000C | I=IDLE, E=ERROR,C=Card Detection(Active=0) |
| 1Fh | モード | 0000 0MMM | MMM : 000=TeC,001=TaC,010=DEMO1,011=DEMO2,111=RN4020FactoryReset |
| 2Dh | RN4020-Cmd | 0000 FHCS | RN4020(F=Flow Control, H=Hw Pin, C=Cmd Pin, S=Sw Pin (初期値=0001)) |
| 30h- | TeCコンソール | - | 詳細は「I/Oマップ詳細」シートに掲載 |
| D0h- | MMU | - | 詳細は「I/Oマップ詳細」シートに掲載 |
| FDh | ロータリーSW(IN) | 000S SSSS | 0=G0,1=G1,...11=G11,12=FP,13=SP,14=PC,15=FLAG,16=MD,17=MA |
| FFh | 機能レジスタ(IN) | 0000 FFFF | 0=ReadReg, 1=WriteReg, 13=ReadMem, 14=WriteMem |

図 A.2 TaC のメモリマップと I/O マップ

I/Oマップ詳細

Ver.10.0.0(TeC7a,b,c,d対応)

2022/3/13

コンソール制御のI/Oポート解説

| TeCコンソールI/Oアドレス | | | |
|-----------------|-------|-----------|------------|
| | | Read | Write |
| データLED | (30h) | データランプ | 空き |
| データSW | (32h) | 00H | データスイッチ |
| 機能SW | (34h) | 00H | ABCD EFGH |
| 制御と機能SW | (36h) | ---- --RS | I---- -JKL |

| TeCコンソールの操作ビット | | | | | | | |
|----------------|----------|---|----------|---|---------|---|----------|
| A | BREAK-SW | B | STEP-SW | C | RUN-SW | D | STOP-SW |
| E | SETA-SW | F | INCA-SW | G | DECA-SW | H | WRITE-SW |
| I | ENABLE | J | RESET-SW | K | LEFT-SW | L | RIGHT-SW |

| TeCコンソールの状態確認ビット | | | | | | | |
|------------------|-------|---|---------|--|--|--|--|
| R | RESET | S | SETA-SW | | | | |

80hから9Fhに配置されるTLBエントリー解説

| TLBエントリーの構成 | | | | | | | |
|------------------|--------------------|----|----|----|----|-------|-------|
| 上位 8 ビット(偶数アドレス) | 下位 1 6 ビット(奇数アドレス) | | | | | | |
| 23-16 | 15 | 14 | 13 | 12 | 11 | 10-8 | 7-0 |
| PAGE | V | * | * | R | D | R/W/X | FRAME |

PAGE:ページ番号 V:Valid *:未定義 R:Reference R/W/X:Read/Write/eXecute
D:Dirty FRAME:フレーム番号

図 A.3 TaC の I/O 詳細

TaC命令表

Ver.10.0.0(TeC7a,b,c,対応)

2022/3/13

| 命令 | ニーモニック | | オペコード | | アドレッシングモード(数値はステート数) | | | | | | | | | フラグ 変化 | 説明 |
|--------------------------|--------|-------|-------|-------|----------------------|-------|-----|--------|-----|------|------|--------|------|-----------|-----------------------------------|
| | 命令 | オペランド | OP | Rd Rx | Drct | Index | Imm | FP Rlt | Reg | Imm4 | Indr | B Indr | Othr | | |
| No Operation | NO | | 00h | 0h 0h | -- | -- | -- | -- | -- | -- | -- | -- | 3 | × | 何もしない |
| Load | LD | Rd,EA | 08h | Rd EA | 7 | 7 | 5 | 5 | 4 | 4 | 5 | 5 | -- | × | Rd ← [EA] |
| Store | ST | Rd,EA | 10h | Rd EA | 7 | 7 | -- | 5 | -- | -- | 5 | 5 | -- | × | [EA] ← Rd |
| Add | ADD | Rd,EA | 18h | Rd EA | 7 | 7 | 5 | 5 | 4 | 4 | 5 | 5 | -- | ○ | Rd ← Rd + [EA] |
| Subtract | SUB | Rd,EA | 20h | Rd EA | 7 | 7 | 5 | 5 | 4 | 4 | 5 | 5 | -- | ○ | Rd ← Rd - [EA] |
| Compare | CMP | Rd,EA | 28h | Rd EA | 7 | 7 | 5 | 5 | 4 | 4 | 5 | 5 | -- | ○ | Rd - [EA] |
| Logical And | AND | Rd,EA | 30h | Rd EA | 7 | 7 | 5 | 5 | 4 | 4 | 5 | 5 | -- | ○ | Rd ← Rd and [EA] |
| Logical Or | OR | Rd,EA | 38h | Rd EA | 7 | 7 | 5 | 5 | 4 | 4 | 5 | 5 | -- | ○ | Rd ← Rd or [EA] |
| Logical Xor | XOR | Rd,EA | 40h | Rd EA | 7 | 7 | 5 | 5 | 4 | 4 | 5 | 5 | -- | ○ | Rd ← Rd xor [EA] |
| Add with Scale | ADDS | Rd,EA | 48h | Rd EA | 7 | 7 | 5 | 5 | 4 | 4 | 5 | 5 | -- | ○ | Rd ← Rd + [EA]*2 |
| Multiply | MUL | Rd,EA | 50h | Rd EA | 7 | 7 | 5 | 5 | 4 | 4 | 5 | 5 | -- | ○ | Rd ← Rd × [EA] |
| Divide | DIV | Rd,EA | 58h | Rd EA | 23 | 23 | 21 | 21 | 20 | 20 | 21 | 21 | -- | ○ | Rd ← Rd / [EA] |
| Modulo | MOD | Rd,EA | 60h | Rd EA | 23 | 23 | 21 | 21 | 20 | 20 | 21 | 21 | -- | ○ | Rd ← Rd % [EA] |
| Shift Left Arithmetic | SHLA | Rd,EA | 80h | Rd EA | 7 | 7 | 5 | 5 | 4 | 4 | 5 | 5 | -- | ○ | Rd ← Rd << [EA] |
| Shift Left Logical | SHLL | Rd,EA | 88h | Rd EA | 7 | 7 | 5 | 5 | 4 | 4 | 5 | 5 | -- | ○ | Rd ← Rd << [EA] |
| Shift Right Arithmetic | SHRA | Rd,EA | 90h | Rd EA | 7 | 7 | 5 | 5 | 4 | 4 | 5 | 5 | -- | ○ | Rd ← Rd >> [EA] |
| Shift Right Logical | SHRL | Rd,EA | 98h | Rd EA | 7 | 7 | 5 | 5 | 4 | 4 | 5 | 5 | -- | ○ | Rd ← Rd >>> [EA] |
| Jump on Zero | JZ | EA | A0h | 0h EA | 5 | 5 | -- | -- | -- | -- | -- | -- | -- | × | If (Z) PC ← EA |
| Jump on Carry | JC | EA | A0h | 1h EA | 5 | 5 | -- | -- | -- | -- | -- | -- | -- | × | If (C) PC ← EA |
| Jump on Minus | JM | EA | A0h | 2h EA | 5 | 5 | -- | -- | -- | -- | -- | -- | -- | × | If (S) PC ← EA |
| Jump on Overflow | JO | EA | A0h | 3h EA | 5 | 5 | -- | -- | -- | -- | -- | -- | -- | × | if (V) PC ← EA |
| Jump on greater than | JGT | EA | A0h | 4h EA | 5 | 5 | -- | -- | -- | -- | -- | -- | -- | × | If (not (Z or (S xor V))) PC ← EA |
| Jump on greater or equal | JGE | EA | A0h | 5h EA | 5 | 5 | -- | -- | -- | -- | -- | -- | -- | × | if (not (S xor V)) PC ← EA |
| Jump on less or equal | JLE | EA | A0h | 6h EA | 5 | 5 | -- | -- | -- | -- | -- | -- | -- | × | If (Z or (S xor V)) PC ← EA |
| Jump on less than | JLT | EA | A0h | 7h EA | 5 | 5 | -- | -- | -- | -- | -- | -- | -- | × | If (S xor V) PC ← EA |
| Jump on Non Zero | JNZ | EA | A0h | 8h EA | 5 | 5 | -- | -- | -- | -- | -- | -- | -- | × | If (not Z) PC ← EA |
| Jump on Non Carry | JNC | EA | A0h | 9h EA | 5 | 5 | -- | -- | -- | -- | -- | -- | -- | × | If (not C) PC ← EA |
| Jump on Non Minus | JNM | EA | A0h | Ah EA | 5 | 5 | -- | -- | -- | -- | -- | -- | -- | × | If (not S) PC ← EA |
| Jump on Non Overflow | JNO | EA | A0h | Bh EA | 5 | 5 | -- | -- | -- | -- | -- | -- | -- | × | If (not V) PC ← EA |
| Jump on higher | JHI | EA | A0h | Ch EA | 5 | 5 | -- | -- | -- | -- | -- | -- | -- | × | If (not (Z or C)) PC ← EA |
| Jump on lower or same | JLS | EA | A0h | Eh EA | 5 | 5 | -- | -- | -- | -- | -- | -- | -- | × | If (Z or C) PC ← EA |
| Jump | JMP | EA | A0h | Fh EA | 5 | 5 | -- | -- | -- | -- | -- | -- | -- | × | PC ← EA |
| Call subroutine | CALL | EA | A8h | 0h EA | 7 | 7 | -- | -- | -- | -- | -- | -- | -- | × | [--SP] ← PC, PC ← EA |
| Input | IN | Rd,EA | B0h | Rd EA | 6 | -- | -- | -- | -- | -- | 4 | 4 | -- | × | Rd ← IO[EA] |
| Output | OUT | Rd,EA | B8h | Rd EA | 5 | -- | -- | -- | -- | -- | 3 | 3 | -- | × | IO[EA] ← Rd |
| Push Register | PUSH | Rd | C0h | Rd 0h | -- | -- | -- | -- | -- | -- | -- | -- | 5 | × | [--SP] ← Rd |
| Pop Register | POP | Rd | C4h | Rd 0h | -- | -- | -- | -- | -- | -- | -- | -- | 5 | × | Rd ← [SP++] |
| Return from Subroutine | RET | | D0h | 0h 0h | -- | -- | -- | -- | -- | -- | -- | -- | 6 | × | PC ← [SP++] |
| Return from Interrupt | RETI | | D4h | Fh 0h | -- | -- | -- | -- | -- | -- | -- | -- | 9 | × | FLAG ← [SP++], PC ← [SP++] |
| Supervisor Call | SVC | | F0h | 0h 0h | -- | -- | -- | -- | -- | -- | -- | -- | 14 | × | システムコール |
| Halt | HALT | | FFh | 0h 0h | -- | -- | -- | -- | -- | -- | -- | -- | 3 | × | CPU停止 |

アドレッシングモード(上の表中EAの詳細)に付いて

| アドレッシングモード | 略記 | ニーモニック (EA部分の標記方法) | 命令フォーマット | | EA(実効アドレス)の決め方 | |
|------------------------|--------|------------------------|-----------|-------|----------------|---|
| | | | 第1ワード | 第2ワード | 略記 | 解説 |
| Direct | Drct | OP Rd, <u>Dsp</u> | OP+0 Rd0h | Dsp | [Dsp] | Dsp番地 |
| Indexed | Index | OP Rd, <u>Dsp, Rx</u> | OP+1 RdRx | Dsp | [Dsp+Rx] | (Dsp+Rxレジスタの内容)番地 |
| Immediate | Imm | OP Rd, <u>#Imm</u> | OP+2 Rd0h | Imm | Imm | Immそのもの |
| FP Rerative | FP Rlt | OP Rd, <u>Dsp4, FP</u> | OP+3 RdD4 | -- | [Dsp4+FP] | (D4を符号拡張した値×2 + FPレジスタの内容)番地(D4=Dsp4/2) |
| Register | Reg | OP Rd, <u>Rs</u> | OP+4 RdRs | -- | Rs | Rsレジスタの内容 |
| 4bit Signed Immediate | Imm4 | OP Rd, <u>#Imm4</u> | OP+5 RdI4 | -- | Imm4 | I4を符号拡張した値そのもの |
| Register Indirect | Indr | OP Rd, <u>@Rx</u> | OP+6 RdRx | -- | [Rx] | Rxレジスタの内容番地 |
| Byte Register Indirect | B Indr | OP Rd, <u>@Rx</u> | OP+7 RdRx | -- | [Rx] | Rxレジスタの内容番地(但し番地の内容は8bitデータ) |
| Other | Othr | OP Rd | OP Rd0h | -- | -- | なし |
| | | OP | OP 0h0h | -- | -- | なし |

注4

※アセンブリ言語でDspとDsp4、ImmとImm4の標記は同じ(値によりアセンブラが自動判定)

※FP相対で、Dsp4は-16～+14の偶数

色付きのセルは特権命令 特権違反が発生時は、スタックに違反を起こす前のPCが保存される

注1: RETI命令は特権モードでのみEPIフラグを変化させる

注2: D4はDsp4(4bitディスプレースメント)の1/2の値

注3: I4はImm4(4bit即値)のこと

注4: アドレッシングモードによりOPの値が変化する

図 A.4 TaC の機械語命令表



図 A.5 TeC7d のプリント基板回路図

TeC7 ハードウェアマニュアル Ver. 1.0.1

発行年月 2020年3月 Ver. 1.0.1

発 行 独立行政法人国立高等専門学校機構
徳山工業高等専門学校
情報電子工学科 重村哲至
〒745-8585 山口県周南市学園台
sigemura@tokuyama.ac.jp