

# 第4章 自作CPUにオンチップ・デバッグ機能を搭載する

続・私はこうしてCPUを開発した！

栗須 基弘



最近の多くのRISC型CPUには、JTAGポート(または、それと同様の機能のポート)がついている。CPUでは、バウンダリ・スキャン・テストだけでなく、オンチップ・デバッグのためにも用いることができる。本誌1999年11月号の特集で取り上げたPLD用CPUコアでは、このオンチップ・デバッグ機能が搭載されていた。ここでは、そのオンチップ・デバッグ機能がどのように実現されていたのかを詳しく説明する。JTAG仕様に準拠しているわけではないが、原理とその実現の観点では、その差は無視できるであろう。(編集部)



## 1. オンチップ・デバッグ機能の概略

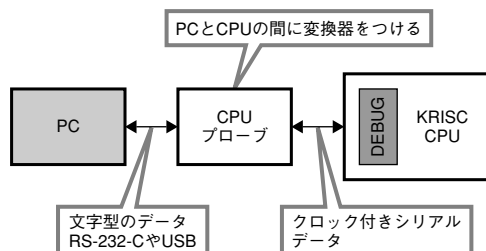
### ■オンチップ・デバッグ機能って何？

1998年末、筆者が開発したCPUコア(KRISC：本誌1999年11月号の特集を参照)の話を米国でしたとき、開発環境の話が頻繁に出てきました。これはどうしても、オンチップ・デバッグ機能を搭載する必要があると強く感じました。

しかし、どうやって設計するのか、参考文献などありませんでした。ただ、JTAGなどのシリアル通信で実現しているということだけはわかっていました。そこでまずは、システム構成から考えました。

### ■システム基本構成 — シリアル通信を使う

図1にデバッグのためのシステム構成を示します。パソコンからの接続には現在、一般的なRS-232-Cを使用することを想



〔図1〕 オンチップ・デバッグ・モジュールとCPUプロープ

定しています。将来、USBやEthernetなど、違った手段で接続する可能性も十分考えられるので、パソコンから直接CPUに接続することを避け、間にCPUプロープというフォーマット変換チップを挿入しました。こうしておけば、将来、いろいろな通信方式に柔軟に対処できると考えたからです。

パソコンとCPUプロープの間がRS-232-Cだとすると、理想のデータは文字型になります。これは、デバッグ・ソフトの観点からも作成やチェックが容易になるからです。しかし、この文字型を直接、RISCエンジンに放り込むには、速度の点で問題が発生します。最終の量産品になった場合には、なるべく高速な転送処理が望まれることを考え、CPUプロープとCPUの間はクロック付きでシリアル転送することにしました。

### ■CPUプロープとオンチップ・デバッグ・モジュールから構成

図2にCPUプロープとオンチップ・デバッグ・モジュールの概略図を記載します。

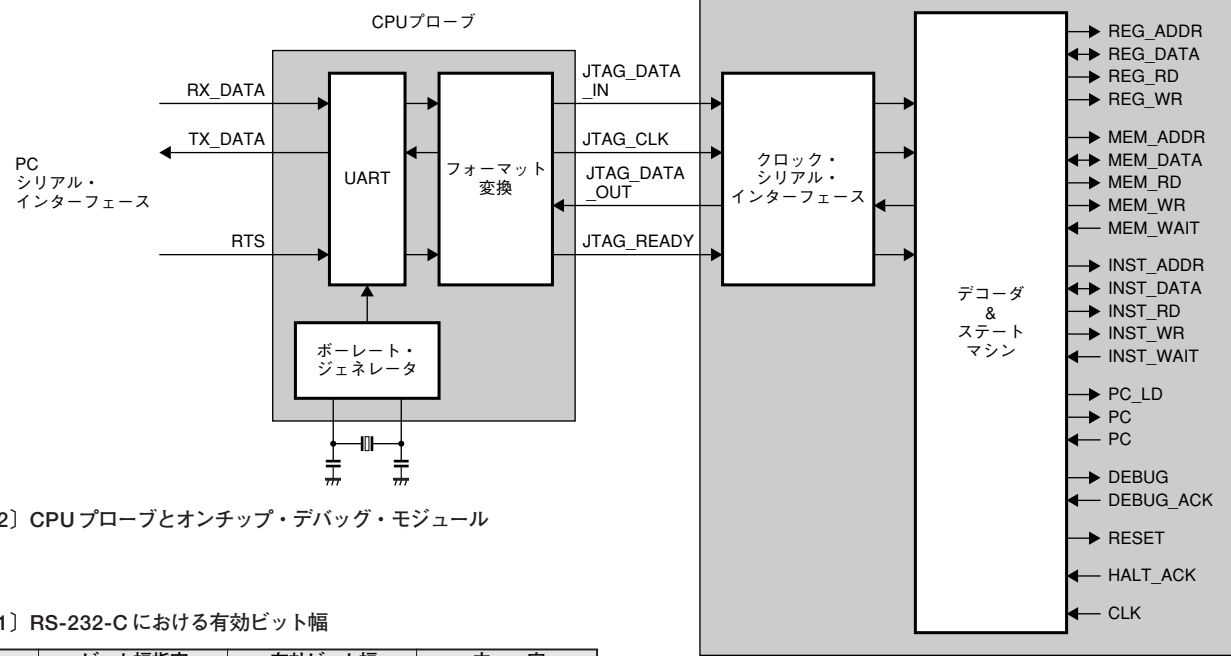
パソコンからの非同期の送信データと受信データ、および受け取る受信データを一時的に止めるためのRTS信号をCPUプロープに接続します。CPUのクロックのほうがRS-232-Cのレートよりもはるかに速いのでCTS信号は使いません。

この文字型データはCPUプロープ内のUART送受信モジュールを使って、シリアル-パラレル変換されます。パラレルになったデータは再度、クロック付きデータに変換されてCPUに転送されます。

### ■非同期データ転送をクロック付きデータに変換

図3にRS-232-Cの非同期シリアル通信フォーマットを示します。8ビット・データがスタート・ビット、ストップ・ビットでサンドイッチされています。パリティ・ビットも追加できます。転送順序はLSBが最初で、MSBが最後です。

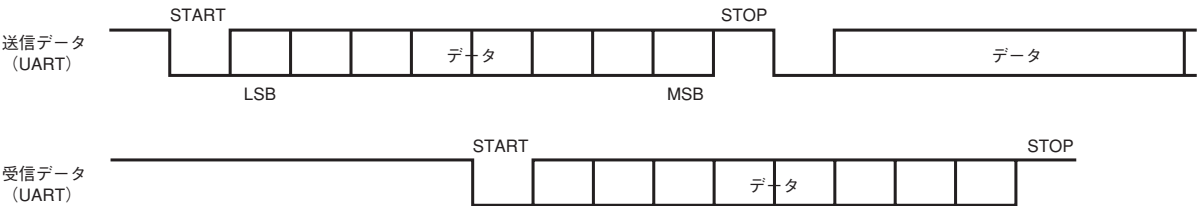
図4がこれらの非同期シリアル・データを18MHzクロック付きのシリアル・データにしたときの波形です。スタート・ビット、有効転送ビット幅の指定(2ビット)、有効データ(5ビット、8ビット、16ビット、32ビット)からなります。転送順序はMSBが最初です。多くのマルチメディア分野のデータ



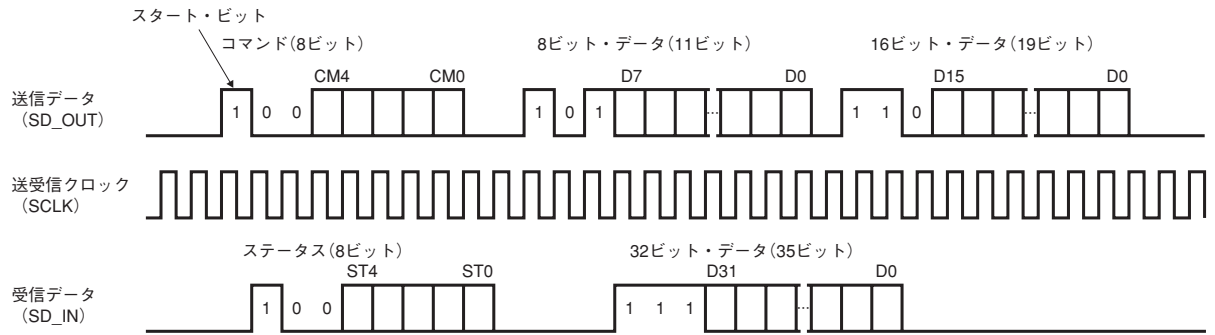
〔図2〕 CPU プローブとオンチップ・デバッグ・モジュール

〔表1〕 RS-232-C における有効ビット幅

	ビット幅指定	有効ビット幅	内 容
1	00	5	5 ビット・コマンド
2	01	8	8 ビット・データ
3	10	16	16 ビット・データ
4	11	32	32 ビット・データ



〔図3〕 RS-232-C 非同期シリアル通信フォーマット



〔図4〕 クロック付きシリアル通信フォーマット

転送はMSBから始まります。

有効データのビット・サイズは4種類あり、その組み合わせを表1に示します。

8, 16, 32 ビット・データは場合により、使い分けます。たとえば、命令空間のデータを読み込む場合には32ビット単位で実行します。アドレスの設定にも32ビットを使用します。今回はデータを非同期形式に似た構造にしていますが、もち

ろん、これ以外の転送フォーマットも考えられます。たとえばUSBのような半2重ですが、シンク・パターンを使ってPID (トークンPacketID) にしたがって、いろいろなものを定義することもできます。

重要なことは、このデバッグ・モジュールが小さく、かつ、転送速度ができるだけ速いということです。今回採用したデータ・フォーマットはシリアル・インターフェース部分の設計