基礎から学ぶ Verilog HDL & FPGA 設計

第10回
CPUを作ろう(1) 基本アーキテクチャの設計

中野浩嗣，伊藤靖朗

デバイスの記事　ビギナーズ

今回は，本連載でこれまでに設計した五つのモジュール，カウンタ(counter)，ステート・マシン(state)，スタック(stack)，算術論理演算回路(alu)，メモリ(ブロックRAM；ram)を構成要素とした小型CPUを設計する．各モジュールの接続には，アドレス・バス(abus)，データ・バス(dbus)という二つのバスを使用する．これらの構成要素を制御する信号線(制御線)を導入し，制御線のロジックを定めることによりCPUが完成する．　(筆者)

ここで作成するCPUは，必要最低限の機能を持ったものなので，「TINYCPU」と呼ぶことにします．今回は，TINYCPUのアーキテクチャ，機械語命令セット，制御線，制御線の論理を決定する手法を学びます．

● 作成する学習用CPU「TINYCPU」のアーキテクチャ

前回(2008年7月号，pp.123-129)命令フェッチ回路fetch.vを設計しました．この回路を拡張して「TINYCPU」を設計します．構成部品として，スタックstackと算術論理演算回路aluを追加します．また，外部からの入力信号を取り込む入力ポートinを持ちます．**図1**はTINYCPUのアーキテクチャの概略です．これらの構成要素を適切なロジックで接続することにより，TINYCPUが完成します．

設計するTINYCPUは完全なスタック・アーキテクチャです．本連載第6回(2008年1月号，pp.115-118)で説明した後置記法で記述された式がそのまま評価できるようになっています．よって，2項演算はスタック・トップとスタックの2番目の間で行われ，演算結果はスタック・トップに格納されます．

単項演算はスタック・トップに対して行われ，演算結果はそのままスタック・トップに書き込まれます．

また，メモリram0のデータの読み書きもスタック・トップとの間で行われます．例えば，以下の機械語プログラム

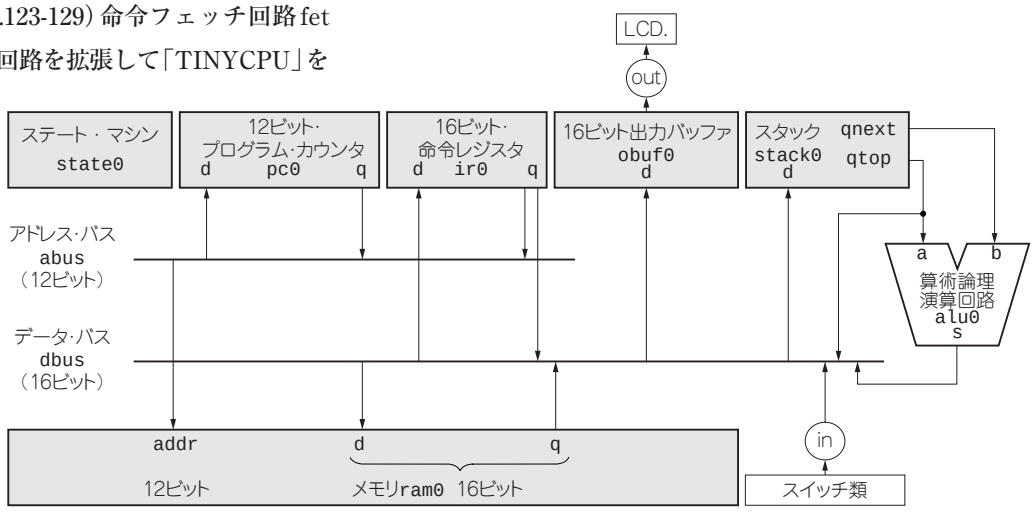

図1
TINYCPUのアーキテクチャ

Keyword　CPU，命令セット，制御線，制御線のロジック，ニーモニック，オペランド，ポップ，プッシュ，カウンタ，ステート・マシン，スタック，算術演算回路，メモリ，TINYCPU

は，メモリの8番地の値に1を加算します．

PUSH 8　メモリの8番地のデータをスタックにプッシュ
PUSHI 1　スタックに1をプッシュ
ADD　スタック・トップと2番目を加算し，スタック・トップに格納
POP 8　スタック・トップの値をメモリの8番目に格納

このように，演算やメモリとのデータのやりとりは，必ずスタック上で行われます．一般のCPUが持つような汎用レジスタがない，単純な構造になっています．

TINYCPUでは，アドレス値は12ビットのアドレス・バスabusを，そのほかのデータは16ビットのデータ・バスdbusを介してやりとりを行います．アドレス・バスabusはメモリram0のアドレス入力addrに，データ・バスdbusはデータ入力dに直結します．

TINYCPUは，入力ポートinと出力ポートoutを備え，外部とデータをやりとりします．inはFPGAボードの三つのプッシュ・スイッチと四つのスライド・スイッチと接続し，スイッチが押されているかどうかの情報を取得します．outは出力バッファobuf0が格納している16ビットの値を外部に出力します．outに出力される値は，FPGAボード上のLCDに4けたの16進数で表示されます．

表1　命令セット，ニーモニックと命令コード

	ニーモニック	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	16進数表示
1	HALT	0	0	0	0	—												0000
2	PUSHI I	0	0	0	1	I（2の補数）												1000+I
3	PUSH A	0	0	1	0	A（符号なし2進数）												2000+A
4	POP A	0	0	1	1	A												3000+A
5	JMP A	0	1	0	0	A												4000+A
6	JZ A	0	1	0	1	A												5000+A
7	JNZ A	0	1	1	0	A												6000+A
8	IN	1	1	0	1	—												D000
9	OUT	1	1	1	0	—												E000
10	OP f	1	1	1	1	—							f（算術論理演回路の機能選択）					F000+f
	ADD	1	1	1	1	—							0	0	0	0	0	F000
	SUB	1	1	1	1	—							0	0	0	0	1	F001
	MUL	1	1	1	1	—							0	0	0	1	0	F002
	SHL	1	1	1	1	—							0	0	0	1	1	F003
	SHR	1	1	1	1	—							0	0	1	0	0	F004
	BAND	1	1	1	1	—							0	0	1	0	1	F005
	BOR	1	1	1	1	—							0	0	1	1	0	F006
	BXOR	1	1	1	1	—							0	0	1	1	1	F007
	AND	1	1	1	1	—							0	1	0	0	0	F008
	OR	1	1	1	1	—							0	1	0	0	1	F009
	EQ	1	1	1	1	—							0	1	0	1	0	F00A
	NE	1	1	1	1	—							0	1	0	1	1	F00B
	GE	1	1	1	1	—							0	1	1	0	0	F00C
	LE	1	1	1	1	—							0	1	1	0	1	F00D
	GT	1	1	1	1	—							0	1	1	1	0	F00E
	LT	1	1	1	1	—							0	1	1	1	1	F00F
	NEG	1	1	1	1	—							1	0	0	0	0	F010
	BNOT	1	1	1	1	—							1	0	0	0	1	F011
	NOT	1	1	1	1	—							1	0	0	1	0	F012

● TINYCPUの命令セット

TINYCPUで用いる命令セットは必要最小限とし，**表1**にある10種類の命令を使用します．命令フォーマットの上位4ビット（つまり［15：12］）で命令の種類を区別します．命令PUSHIは，下位12ビット（つまり［11：0］）がオペランドIです．同様に，PUSH，POP，JMP，JZ，JNZは下位12ビットがオペランドAです．同じ下位12ビットをIとAに区別してあります．Iは2の補数として扱われるのに対し，Aはアドレスを表すため符号なし2進数として扱われるからです．

OP fは算術論理演算のための命令で，下位5ビットf（つまり［4：0］）は，算術論理演算回路alu0の機能選択入力です．よって，OP fは，alu0の機能入力によって定められる算術論理演算命令に対応します．

命令セットの各命令の動作は以下の通りです．

(1) HALT：ステート・マシンstate0の状態がIDLEに遷移し，CPUとしての動作を停止します．

(2) PUSHI I（PUSH Immediate）：オペランドIを即値（Immediate値）としてスタックにプッシュします．ただし，オペランドIは12ビット幅で，スタックは16ビット幅なので，12ビットを16ビットに拡張するときには符号拡張を行います．つまり，オペランドIの最上位ビットが0のときは，4ビットの4'b0000を上位に付加し，1のときは4ビットの4'b1111を付加します．符号拡張により，2の補数表現として値が変わらないことになります．例えば，－3は12ビットの2の補数表現で12'b111111111101ですが，16ビットの2の補数表現は16'b1111111111111101となり，符号拡張で値が変わらないことが確認できます．

(3) PUSH A：メモリのA番地の値をスタックにプッシュします．よって，スタック・トップqtopの値は，メモリのA番地の値になります．PUSHI Iがオペランドの値そのものをプッシュするのに対して，PUSH Aはオペランドの値を番地として，メモリのその番地の値をプッシュします．