CA 25－2
(1977. 3. 18)

演算用ならびに制御用に設計された1チップ4ビットマイクロコンピュータ

高井 昶, 岩先 純一, (日本電気)

§1. はじめに

現在，日本を例にとると，マイクロコンピュータ市場の約70%は4ビットマイクロコンピュータで占められており，将来さらに広範囲の産業分野に応用されようとしている。しかし、今後これら未開拓分野で4ビットマイクロプロセッサが応用される為には，夫々の用途の機能を満たし，システム的に使い易く，しかも経済性に優れたプロセッサが必要とされる。

この思想に基き開発されたマイクロプロセッサがμCOM42／43であり，μCOM42は演算を主とするシステムを，μCOM43は制御を主とするシステムを夫々ねらった1チップ4ビットマイクロコンピュータである。

1チップ化は経済性のみでなく，システムの信頼性を向上させ，各々の専用化はソフトウエア及びハードウエアの使い易さ，機能を提供してくれるものである。

以下に1チップ化及び専用化に焦点を当て，アーキテクチャの概要とその特徴について述べる。

§2. アーキテクチャとその特徴

1チップのマイクロコンピュータは，標準的なマイクロプロセッサ（インテル社の8080，モトローラ社の6800，TI社の9900等）とはかなり異ったアーキテクチャである。これは，1チップでシステムを構成することを考え，ROM, RAM, I/Oポート等を内蔵し，命令体系もそれなりに充分考慮している。

＝(1)＝ プログラムメモリとデータメモリの分離

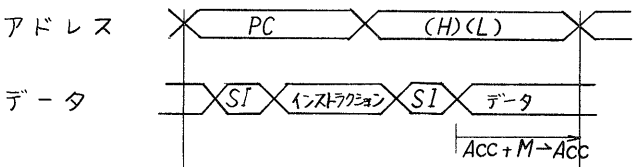
従来のマイクロプロセッサはLSIの端子数，システムの拡張性を考慮してバス方式を採用しており，命令体系もミニコンピュータクラスの汎用性のあるもので，プログラムメモリとデータメモリを同一メモリとして扱うことが出来る。

しかし1チップマイクロコンピュータでは，プログラムメモリ（ROM）とデータメモリ（RAM）を完全に分離して，ROMアドレスは，プログラム・カウンタで，RAMアドレスは，データ・ポインタで指定し，ROM出力は，インストラクション・バスに，RAMデータは4ビット・バスに夫々分離されている。

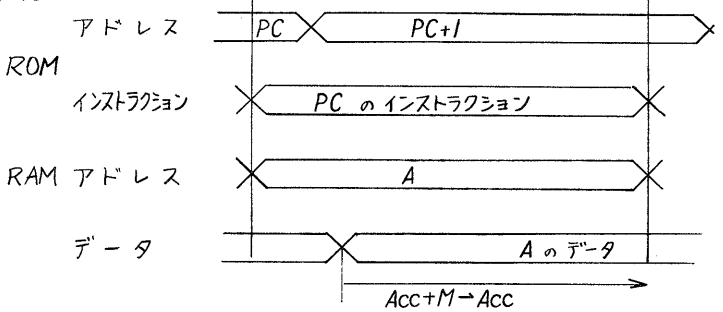
このような構成をすると，プログラムメモリとデータメモリの制御系が分離されて，マイクロプログラム制御方式が容易になり，処理速度の高速化と，複数の命令の同時実行が可能になる。（命令体系の項参照）

ADD(AD) 命令

a. 8080A



b. μCOM42/43



図－1

例としてADD命令（μCOM42／43ではAD命令に相当）で説明すると（図1），8080Aはアドレス・バスに先ずプログラムアドレスを出力し，データバスからインストラクションを受け取り，デコードしてADD命令と解読し，オペランドのアドレスもアドレス・バスに出力し，データを受け取り，Acc+M→Acc を実行する。これは，CPU，メモリ等に高速のデバイスを使用している為可能な方法である。

一方，μCOM42／43では，Low Cost に重点を置いたのでPチャンネル・アルミゲート・プロセスを使用している。このプロセスで最大のスピードを得る為に，ROM／RAMのアドレス及びデータ・バスを分離した。

しかし，プログラムとデータを分離すると，RAMの中にプログラムを入れたり，データによるプログラムの分岐が不可能となるが，μCOM42／43は，アキュムレータ・ジャンプ命令を設けてこの不便を回避している。

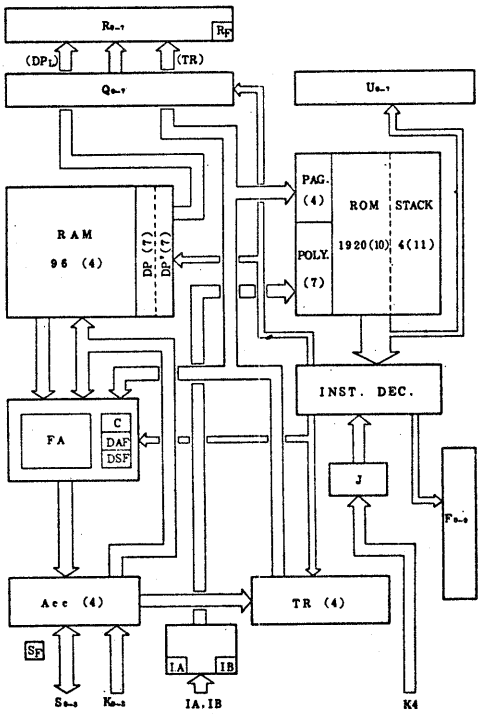
次に，1チップ構成の利点を充分に生かすには，内蔵されているROM，RAMの容量が問題になる。μCOM42／43では，補助的にRAMを外付けする事が出来る為，ROM容量に重点を置いて構成している。内蔵されているROM／RAMを充分に使用した用途が一番コスト・パーフォーマンスが良いので，異なった容量のROM，RAMを内蔵したファミリーを用意している。

(2) きめ細かな命令体系

1チップマイクロコンピュータは命令機能的にも従来のマイクロプロセッサと異なっている。

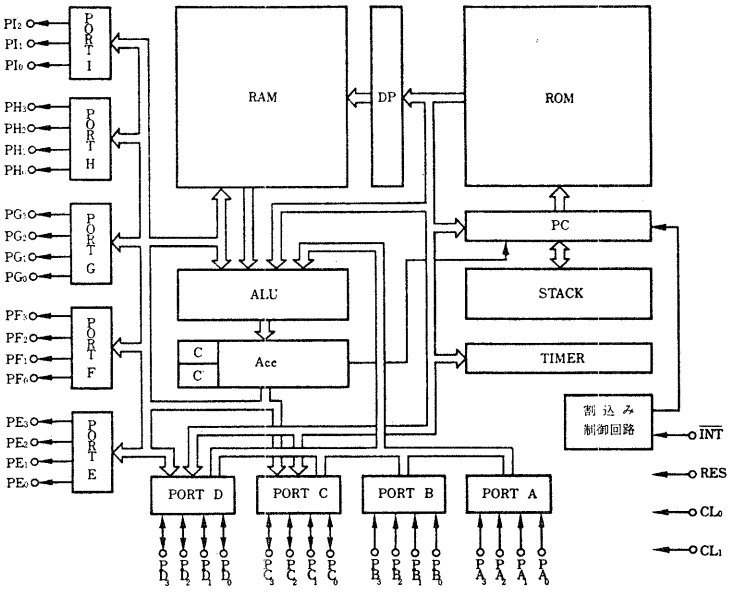
従来のプロセッサは汎用性に重点を置いた命令体系の為，2～3バイト命令が多いのに反し，1チップマイクロコンピュータは夫々の分野に適した命令体

■COM-42の機能ブロック図



POLY：ポリノミナル・カウンタ
FA　：フル・アダー
TR　：テンポラリ・レジスタ
IA　：割込みポート
IB　：割込みポート
DAF：10進加算モード F/F
DSF：10進減算モード F/F
C　：キャリー F/F
Acc：アキュムレータ
DP　：データ・ポインタ
DP'：退避用データ・ポインタ

UCOM-43の機能ブロック図



DP　：データ・ポインタ　　PC：プログラム・カウンタ
ALU：演算論理ユニット　　C　：キャリー　F/F
Acc　：アキュムレータ　　C'　：キャリー退避　F/F

系の為と，内蔵されたＲＯＭを出来るだけ効率良く利用する為，１バイト命令が主体になっている。

特にＲＡＭのデータ操作に対する複合命令は強力な１バイト命令である。これは，プログラムメモリとデータメモリを分離している為，データ操作に時間的余裕があり，ロード，ストア命令等と同じサイクルにデータ・ポインタの増減とジャッジを行っている。

この例をＸＭＤ命令を利用して説明すると，次に示す各動作を同一命令時間に実行するものであり，ミニコンピュータの場合のオートインクリメントとスキップに相当する高度な命令である。

(1) アキュムレータの内容と，データポインタで示すメモリの内容の交換
(2) データポインタの上位ビットを命令の一部でEXOR 修飾する。
（μＣＯＭ４２と４３でその修飾ビットは若干異なる。）
(3) データポインタの下位ビットをデクリメントする。
(4) データポインタの下位ビットが指定数になると STATUS をセットし，次の命令を SKIP する様にする。

これら複合命令はＲＡＭのデータを操作する命令群に効用され，１チップによるデータ処理スピード向上に役立っている。

```
MOVE:  LDI   17H
       LM    3
       XMD   3
       JCP   $-2
       RT
```

コール前

	7	6	5	4	3	2	1	0
10H								
20H	X	X	X	X	X	X	X	X

コール後

	7	6	5	4	3	2	1	0
10H								
20H								

図－２

図２のプログラムは，レジスタ転送サブルーチンの例である。順を追って説明する。

(1) ＬＤＩ命令によりデータポインタを１７に設定する。
(2) ＬＭ命令でアキュムレータにデータをロードする。この時データポインタを"3"で修飾する。
(3) ＸＭＤ命令を実行する時点では(2)によりデータポインタは２７を指定している。
(4) データポインタの値は(3)でデクリメントされるが，この値がハードウェアで指定される値になる迄はＪＣＰ命令でＬＭ命令の番地にジャンプし，データポインタが指定値になるとＪＣＰ命令をＮＯＰ命令に変えてサブルーチンを終了する。

(3) 割り込みについて

従来，割り込み処理機能は４ビット系マイクロコンピュータでは無く，８ビット系以上のシステムが主力であった。μＣＯＭ４２／４３は広範囲の分野で使用される事を考慮し，夫々特徴の有る割り込み処理機能を有している。

割り込み機能には，(1) NMI (NON-MASKABLE INTERRUPT) のように，ハードウェア

で決められた固定番地からスタートする方法と，(2)RST（RE-START）のように，外部割り込みの起因によりスタートする番地を決める事が出来る方法が有る。μCOM42/43では，システム的に使い易い前者の固定番地方式を採用しエッジトリガにて割り込みを受け付けている。

次にμCOM42と43の間の割り込みに於る相違点を説明する。

(A) μCOM42は割り込みの主対象をプリンタとした為に，入力としてはA，B2系統，レベル的には1レベルとした。又，A，Bの入力には互いに優先度を設け，Aが入るとA，B共に割り込み禁止とし，A，B両者が同時に入るとAを優先し，Bのみが入るとBのみ割り込み禁止としてありA，Bに夫々別の割り込み番地を用意してある。割り込みレベルはデータポインタが2系統なので1レベルの制限となっているが，データポインタが破壊されても良いプログラムの場合は2レベルが許される。

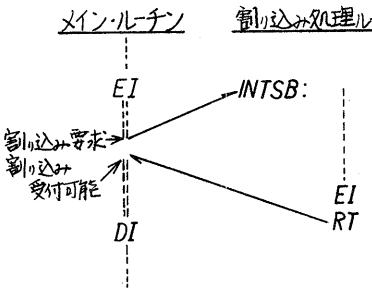


図--3

(B) μCOM43は制御用であり各種の割り込みを考えて1系統としてあるが，割り込み処理期間中に次の割り込みを記憶するフリップフロップが有り，命令でリセットテストする事が出来る。この機能を使用するとプログラム的に2レベルの割り込みが可能になる。

=(4)= 専用化によるALUの形態

8ビットマイクロプロセッサは汎用的なデータ処理に便利なロジック計算が主であるが，1チップマイクロコンピュータは専用化されている為，ALUの機能も汎用的な機能を削り，有効な専用機能を組み込んでいる。

(A) μCOM42は，10進演算のプログラムを容易にする為に，10進補正の為のハードウェアを有している。10進加算はDAM（DECIMAL ADD MODE），減算はDSM（DECIMAL SUBTRUCT）のモード設定命令の次にADC（ADD WITH CARRY）命令をプログラムする事により自動的に10進加減算を実行する。ビット操作に関しては，データポインタで示されるRAMのビットチェック，セット，リセットが可能である。又，基本演算処理桁数は8桁とし，その為のデータポインタのハードウェアストッパを有する。

(B) μCOM43は，4ビット数値データ以外に，1ビットデータを取り扱う場合が多いので，アキュムレータのローテーション，アキュムレータとメモリ内容の1ビット比較，アキュムレータの1ビットテストが可能である。又，入力ポートもメモリと同様に1ビットテストが出来る様に考慮してある。

=(5)= 専用化によるI/Oポートの形態

1チップマイクロコンピュータのメリットの1つは小規模のシステムを構成

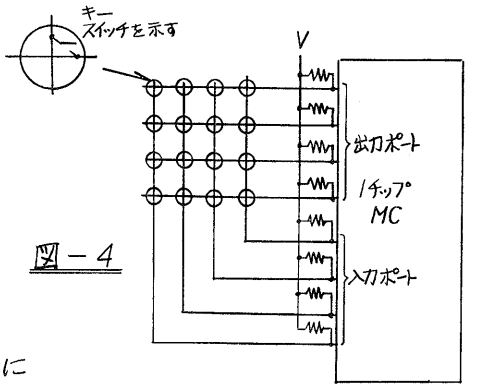
した場合，外付部品（ＳＳＩ，Tr，抵抗等）が減少する事である。この目的の為，I/Oポートがかなり専用化されており μＣＯＭ４２はＥＣＲ，電子秤等を考えて，μＣＯＭ４３は機器制御を考えて，I/Oポートを設計している。

(1) 表示及びパターン出力用ポート

このポートにＲＯＭのイミディエートデータを出力する命令を設け，表示をする場合にはセグメント・デコーダとして用い，パターンを発生する場合にはパターン発生器として用いる。ＲＯＭに必要情報（データ）を入れておける為，種々の表示に対応出来る。

(2) キー及びスイッチ

キー及びスイッチの認識をプログラムで行なえるようになっている。プログラムでキー及びスイッチで調べる方式は，キー及びスイッチの拡張性が有り，種々の仕様に対応出来るものであり，外付けのキー・デコーダ・インターフェースチップは不要になる。図４にその例を示す。



図－４

(3) ＲＡＭ拡張用ポート

内蔵されたＲＡＭで不充分な場合は外部にＲＡＭを接続して，ＲＡＭ容量を拡張する必要が有る。この為に外付ＲＡＭとのデータの授受が出来る入出力ポート，ＲＡＭのアドレスを指定するポートを設けてある。

(4) ビット処理用ポート

１ビット毎に独立に操作する事の出来る命令とポートを有する。このポートを使用する事により，プランジャー，リレー，ランプ等の操作が容易に出来る。

(5) μＣＯＭ４２特有のポート

μＣＯＭ４２はプリンタ制御用のＱレジスタを持っている，これは，一命令づつセット/リセット及びシフトが可能であるのでデータ一致検出により，Ｑレジスタを決定し，これをＲポートに並列出力する事でマグネット制御をする。一方，プリンタの同期信号は割り込み処理信号としてＩＡ及びＩＢポートに入力する事でプリンタの処理を可能にした。

§3. プログラム開発とテスト

ＲＯＭ／ＲＡＭ等を内蔵した１チップマイクロコンピュータになると，従来の汎用プロセッサと異なり，プログラム開発とＬＳＩのテスト方法が重要問題になってくる。

=(1)= テスト方法

従来のプロセッサは、プログラムメモリが内蔵されていない為，ＬＳＩのテスト時に外部から種々の組み合せのインストラクション・プログラムを入力し

て検出率良くテスト出来た。しかし1チップになるとROMが内蔵されている為テストが困難になってくる。そこでμCOM42/43は以下の方法でこの問題を解決した。

(1) ROMの内容を外部にダンプする。
(2) 外部からインストラクションコードを挿入する。
(3) ROMの一部をテスト用に使い(1),(2)で実行不可能な命令をテストする。

これら(1),(2),(3)を実現するハードウェアを有することにより，従来のプロセッサ並の故障検出率が得られた。

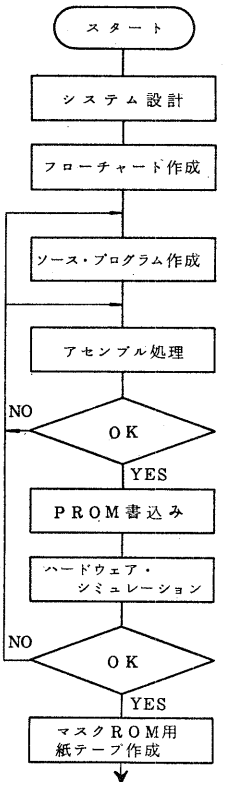
═(2)═ プログラム開発

1チップ化によりROMが内蔵された為に，システムシュミレーションが困難になる。これを実現するには以下の方法が考えられる。

(1) ディスクリート部品によるシュミレータ製作。
(2) 大型システムによるエミュレータ開発。
(3) ROM部を外に取り出したハードウェアシュミレータ製作。

しかし(1)についてはメーカに於る多量生産が不可能。
(2)については共通上位コンピュータの問題がある上にソフト的なシュミレートしか出来ず，テストランには不適。
(3)についてはROMのアドレス及び出力ビット用端子が増える為，新らしく端子数の多いLSIを開発しなければならない。

と，夫々欠点は有るが，(3)は大量に製造が可能，使い易い，場合によってはROM外付のCPUとして利用出来る，等が考えられるのでμCOM42/43では本方式を採用し，64ピンセラミックDIPケースに収めたハードウェアシュミレータを開発した。この方式を採用するに当っても当然テスト方式と同様に，アーキテクチャ上の制限，特徴を生ずる。つまりROMのアドレス系を出力した時に，外部にラッチを設けなくても外部ROMに接続出来る様に考慮しなければならない。この点についてμCOM42/43はROM系，RAM系のアドレスを分離してあるのでハードウェア上の設計が簡略化出来ている。図5に開発フローの例を示す。



図－5

謝辞 1チップマイクロコンピュータを開発するに当たりプロセス及び試作を担当していただいた集積回路事業部，山本宏彰主任，武川藤次郎氏，奥村孝一郎氏，ならびにシステム開発を指導していただいた高島二郎主任に深謝致します。

μCOM-42　インストラクション活用表

命令群	ニーモニック	命令コード		マシンサイクル	オペレーション	スキップ条件
アキュムレータ操作命令	CMA	000000	0001	1	Acc←(Acc)	
	CIA	000011	0000	1	Acc←(Acc)+1	
	INA	000011	0001	½	Acc←(Acc)+1	Carry=1
	DEA	000011	1111	½	Acc←(Acc)−1	Borrow=1
	RFC	000111	0010	1	C←0	
	SFC	000111	0011	1	C←1	
演算命令	DSM	000001	1101	1	Decimal Subtract Mode	
	DAM	000001	1100	1	Decimal Add Mode	
	AD	000001	0000	½	Acc←(Acc)+(DP)	Carry=1
	ADC	000001	0010	1	Acc,C←(Acc)+(DP)+(C)	
	ADI	000011	$I_3I_2I_1I_0$	½	Acc←(Acc)+$I_3I_2I_1I_0$	Carry=1
ロード・ストア命令	LM	000101	$0M_2M_1M_0$	1	Acc←(DP) DP_H←(DP_H)∀$M_2M_1M_0$	
	XM	000101	$1M_2M_1M_0$	1	(Acc)↔(DP) DP_H←(DP_H)∀$M_2M_1M_0$	
	XMI	000100	$0M_2M_1M_0$	½	(Acc)↔(DP) DP_H←(DP_H)∀$M_2M_1M_0$ DP_L←(DP_L)+1	(DP_L)=8 or (DP_L)=0
	XMD	000100	$1M_2M_1M_0$	½	(Acc)↔(DP) DP_H←(DP_H)∀$M_2M_1M_0$ DP_L←(DP_L)−1	(DP_L)=F or (DP_L)=7
	LI*	001111	$I_3I_2I_1I_0$	1	Acc←$I_3I_2I_1I_0$	
データポインタ操作命令	LDI*	001$I_6I_5I_4$	$I_3I_2I_1I_0$	1	DP←I_6~I_0	
	IND	000110	0001	½	DP_L←(DP_L)+1	(DP_L)=8 or (DP_L)=0
	DED	000110	1001	½	DP_L←(DP_L)−1	(DP_L)=F or (DP_L)=7
	XDP	000001	1111	1	(DP)↔(DP')	
	ZAG	000001	1110	1	000DP_L← (DP)'	
レジスタ操作命令	XTA	000110	0000	1	(Acc)↔(TR)	
	LTI*	001110	$I_3I_2I_1I_0$	1	TR←$I_3I_2I_1I_0$	
	QS1	000111	0001	1	Q_{n+1}←Q_n, Q_0←1	
	QS0	000111	0000	1	Q_{n+1}←Q_n, Q_0←0	
ビット操作命令	SB	000111	$11B_1B_0$	1	(DP, B_1, B_0)←1	
	RB	000111	$10B_1B_0$	1	(DP, B_1, B_0)←0	
	SBT	000111	$01B_1B_0$	½	Skip if (DP, B_1, B_0)=1	指定ビット=1
スキップ命令	SC	000110	1000	½	Skip if (C)=1	(C)=1
	SEM	000001	0001	½	Skip if (Acc)=(DP)	(Acc)=(DP)
	SEI	000010	$I_3I_2I_1I_0$	½	Skip if (Acc)=$I_3I_2I_1I_0$	(Acc)=$I_3I_2I_1I_0$
	SK4	000001	0011	½	Skip if K4=1	K4=1
ジャンプ命令	JPT	101$P_6P_5P_4$	$P_3P_2P_1P_0$	1	PC←(TR), P_{6-0}	
	JPA	011$P_6P_5P_4$	$P_3P_2P_1P_0$	1	PC_{6-4}←P_{6-4} PC_{3-0}←P_{3-0}∨(Acc)	
	JCP	010$P_6P_5P_4$	$P_3P_2P_1P_0$	1	PC_{6-0}←P_{6-0}	
サブルーチン制御命令	CAL	100$P_6P_5P_4$	$P_3P_2P_1P_0$	1	(PC)↦(STACK) 1000$P_6P_5P_4P_3P_2P_1P_0$→PC	
	RT	000000	1110	1	PC←(STACK)	
	RTS	000000	1111	2	PC←(STACK) PC←(PC)+1	無条件
割込制御命令	EIA	000000	1000	1	Enable IA port	
	DIA	000000	1001	1	Disable IA port	
	EIB	000110	1110	1	Enable IB port	
	DIB	000110	1111	1	Disable IB port	

命令群	ニーモニック	命令コード		マシンサイクル	オペレーション
入出力命令	OIU	$11I_7I_6I_5I_4$	$I_3I_2I_1I_0$	1	U_{7-0}←I_{7-0} R_{7-0}←(Q)
	ERO	000001	0111	1	Enable R port
	DRO	000001	0110	1	Disable R port
	OQR	000001	1000	1	R←(Q)
	OTR	000001	1001	1	R_{7-4}←(TR), R_{3-0}←(DP_L)
	SFS	000001	0101	1	S ← (Acc)
	RFS	000001	0100	1	S port Input Mode
	IS	000001	1011	1	Acc←S
	IK	000001	1010	1	Acc←K
	RF1	000000	0010	1	F_1←0
	SF1	000000	0011	1	F_1←1
	RF2	000000	0100	1	F_2←0
	SF2	000000	0101	1	F_2←1
	RF3	000000	0110	1	F_3←0
	SF3	000000	0111	1	F_3←1
	RF4	000000	1010	1	F_4←0
	SF4	000000	1011	1	F_4←1
	RF5	000000	1100	1	F_5←0
	SF5	000000	1101	1	F_5←1
	RF6	000110	0010	1	F_6←0
	SF6	000110	0011	1	F_6←1
	RF7	000110	0100	1	F_7←0
	SF7	000110	0101	1	F_7←1
	RF8	000110	0110	1	F_8←0
	SF8	000110	0111	1	F_8←1
	RF9	000110	1010	1	F_9←0
	SF9	000110	1011	1	F_9←1
	RF0	000110	1100	1	F_0←0
	SF0	000110	1101	1	F_0←1
その他	NOP	000000	0000	1	No Operation

*：同種の命令が続けて実行されると，2番目以降の命令はNOPになります。

μCOM-43 インストラクション活用表

命令群	ニーモニック	命令コード D7D6D5D4	命令コード D3D2D1D0	バイト	マシンサイクル	オペレーション	スキップ条件
アキュムレータ操作命令	CLA	1001	0000	1	1	Acc←0	
	CMA	0001	0000	1	1	Acc←($\overline{Acc}$)	
	CIA	0001	0001	1	1	Acc←($\overline{Acc}$)+1	
	INC	0000	1101	1	1/2~3	Acc←(Acc)+1 skip if Carry	Carry
	DEC	0000	1111	1	1/2~3	Acc←(Acc)−1 skip if Borrow	Borrow
	CLC	0000	1011	1	1	C←0	
	STC	0001	1011	1	1	C←1	
	XC	0001	1010	1	1	(C)⇄(C')	
	RAR	0011	0000	1	1	(Acc_{n-1})←(Acc_n) C←(Acc0), (Acc3)←(C)	
メモリ操作命令	INM	0001	1101	1	1/2~3	{(DP)}←{(DP)}+1 skip if {(DP)}=0	{(DP)}=0
	DEM	0001	1111	1	1/2~3	{(DP)}←{(DP)}−1 skip if {(DP)}=F	{(DP)}=F
演算命令	AD	0000	1000	1	1/2~3	Acc←(Acc)+{(DP)} skip if Carry	Carry
	ADS	0000	1001	1	1/2~3	Acc, C←(Acc)+{(DP)}+(C) skip if Carry	Carry
	ADC	0001	1001	1	1	Acc, C←(Acc)+{(DP)}+(C)	
	DAA	0000	0110	1	1	Acc←(Acc)+6	
	DAS	0000	1010	1	1	Acc←(Acc)+10	
	EXL	0001	1000	1	1	Acc←(Acc)∨{(DP)}	
ロード・ストア命令	LI	1001	I3I2I1I0	1	1	Acc←I3I2I1I0	
	S	0000	0010	1	1	{(DP)}←(Acc)	
	L	0011	1000	1	1	Acc←{(DP)}	
	LM	0011	10M1M0	1	1	Acc←{(DP)} DPH←(DPH)∨0M1M0	
	X	0010	1000	1	1	(Acc)⇄{(DP)}	
	XM	0010	10M1M0	1	1	(Acc)⇄{(DP)} DPH←(DPH)∨0M1M0	
	XD	0010	1100	1	1/2~3	(Acc)⇄{(DP)} DPL←(DPL)−1 skip if (DPL)=F	(DPL)=F
	XMD	0010	11M1M0	1	1/2~3	(Acc)⇄{(DP)} DPH←(DPH)∨0M1M0 DPL←(DPL)−1 skip if (DPL)=F	(DPL)=F
	XI	0011	1100	1	1/2~3	(Acc)⇄{(DP)} DPL←(DPL)−1 skip if (DPL)=0	(DPL)=0
	XMI	0011	11M1M0	1	1/2~3	(Acc)⇄{(DP)} DPH←(DPH)∨0M1M0 DPL←(DPL)+1 skip if (DPL)=0	(DPL)=0
データ・ポインタ操作命令	LDI	0001 0I5I4I3	0101 I2I1I0	2	2	DP←I5~I0	
	LDZ	1000	I3I2I1I0	1	1	DPH←0 DPL←I3I2I1I0	
	DED	0001	0011	1	1/2~3	DPL←(DPL)−1 skip if (DPL)=F	(DPL)=F
	IND	0011	0011	1	1/2~3	DPL←(DPL)+1 skip if (DPL)=0	(DPL)=0
	TAL	0000	0111	1	1	DPL←(Acc)	
	TLA	0001	0010	1	1	Acc←(DPL)	
レジスタ操作命令・ワーキング	XHX	0100	1111	1	2	(X)⇄(DPH)	
	XLY	0100	1110	1	2	(Y)⇄(DPL)	
	THX	0100	0111	1	2	X←(DPH)	
	TLY	0100	0110	1	2	Y←(DPL)	

命令群	ニーモニック	命令コード D7D6D5D4	命令コード D3D2D1D0	バイト	マシンサイクル	オペレーション	スキップ条件
レジスタ操作命令・ワーキング	XAZ	0100	1010	1	2	(Z)⇄(Acc)	
	XAW	0100	1011	1	2	(W)⇄(Acc)	
	TAZ	0100	0010	1	2	Z←(Acc)	
	TAW	0100	0011	1	2	W←(Acc)	
	XHR	0100	1101	1	2	(R)⇄(DPH)	
	XLS	0100	1100	1	2	(S)⇄(DPL)	
ビット操作命令	SMB	0111	10B1B0	1	1	{(DP, B1B0)}←1	
	RMB	0110	10B1B0	1	1	{(DP, B1B0)}←0	
	TMB	0101	10B1B0	1	1/2~3	skip if {(DP, B1B0)}=1	{(DP, B1B0)}=1
	TAB	0010	01B1B0	1	1/2~3	skip if (Acc(B1B0))=1	(Acc(B1B0))=1
	CMB	0011	01B1B0	1	1/2~3	skip if (Acc(B1B0)) ={(DP, B1B0)}	(Acc(B1B0)) ={(DP, B1B0)}
フラグ操作命令	SFB	0111	11B1B0	1	2	FLAG (B1B0)←1	
	RFB	0110	11B1B0	1	2	FLAG (B1B0)←0	
	FBT	0101	11B1B0	1	2/3~4	skip if (FLAG (B1B0))=1	(FLAG (B1B0))=1
	FBF	0010	00B1B0	1	2/3~4	skip if (FLAG (B1B0))=0	(FLAG (B1B0))=0
比較命令	CM	0000	1100	1	1/2~3	skip if (Acc)={(DP)}	(Acc)={(DP)}
	CI	0001 1100	0111 I3I2I1I0	2	2/3~4	skip if (Acc)=I3I2I1I0	(Acc)=I3I2I1I0
	CLI	0001 1110	0110 I3I2I1I0	2	2/3~4	skip if (DPL)=I3I2I1I0	(DPL)=I3I2I1I0
スキップ命令	TC	0000	0100	1	1/2~3	skip if (C)=1	(C)=1
	TIT	0000	0011	1	1/2~3	skip if (INT F/F)=1 INT F/F←0	(INT F/F)=1
ジャンプ命令	JCP	11P5P4	P3P2P1P0	1	1	PC5~6←P5~P0	
	JMP	1010 P7P6P5P4	0P10P9P8 P3P2P1P0	2	2	PC←P10~P0	
	JPA	0100	0001	1	2	PC5~4←A2A2A1A000	
割込み制御命令	EI	0011	0001	1	1	INTE F/F←1	
	DI	0000	0001	1	1	INTE F/F←0	
サブルーチン命令	CZP	1011	P5P4P3P2	1	1	STACK←(PC) PC←00000P5P4P3P200	
	CAL	1010 P7P6P5P4	1P10P9P8 P3P2P1P0	2	2	STACK←(PC) PC←P10~P0	
	RT	0100	1000	1	2	PC←(STACK)	
	RTS	0100	1001	1	3~4	PC←(STACK) PC←(PC)+1, 2	無条件
割込み/タイマ	STM	0001 10I5I4	0100 I3I2I1I0	2	2	TM F/F←0 TIMER←I5~I0	
	TTM	0000	0101	1	1/2~3	skip if (TM F/F)=1	(TM F/F)=1
入出力命令	SEB	0111	01B1B0	1	2	PORT E (B1B0)←1	
	REB	0110	01B1B0	1	1	PORT E (B1B0)←0	
	SPB	0111	00B1B0	1	1	PORT (DPL, B1B0)←1	
	RPB	0110	00B1B0	1	1	PORT (DPL, B1B0)←0	
	TPA	0101	01B1B0	1	2/3~4	skip if (PORT A (B1B0))=1	(PORT A (B1B0)=1
	TPB	0101	00B1B0	1	1/2~3	skip if (PORT (DPL, B1B0))=1	(PORT (DPL, B1B0))=1
	OE	0100	0100	1	2	PORT E←(Acc)	
	OP	0000	1110	1	1	PORT (DPL)←(Acc)	
	OCD	0001 I7I6I5I4	1110 I3I2I1I0	2	2	PORT C, D←I7~I0	
	IA	0100	0000	2	2	Acc←(PORT A)	
	IP	0011	0010	1	1	Acc←(PORT (DPL))	
その他	NOP	0000	0000	1	1	No Operation	