

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		Drcr	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	7	4	4	6	6	--	×	×	Rd ← [EA]
Load	LD Rd,FLAG	10h Rd 0h	--	--	--	--	4	--	--	--	--	×	×	Rd ← FLAG
Store	ST Rd,EA	10h Rd EA	6	6	--	6	--	--	5	5	--	×	×	[Dsp] ← EA
Add	ADD Rd,EA	18h Rd EA	7	7	5	7	5	4	6	6	--	○	○	Rd ← Rd + [EA]
Subtract	SUB Rd,EA	20h Rd EA	7	7	5	7	5	4	6	6	--	○	○	Rd ← Rd - [EA]
Compare	CMP Rd,EA	28h Rd EA	7	7	5	7	5	4	6	6	--	○	○	Rd - [EA]
Logical And	AND Rd,EA	30h Rd EA	7	7	5	7	5	4	6	6	--	○	○	Rd ← Rd and [EA]
Logical Or	OR Rd,EA	38h Rd EA	7	7	5	7	5	4	6	6	--	○	○	Rd ← Rd or [EA]
Logical Xor	XOR Rd,EA	40h Rd EA	7	7	5	7	5	4	6	6	--	○	○	Rd ← Rd xor [EA]
Add with Scale	ADDS Rd,EA	48h Rd EA	8	8	6	8	6	5	7	7	--	○	○	Rd ← Rd + [EA]*2
Multiply	MUL Rd,EA	50h Rd EA	57	57	55	57	55	54	56	56	--	○	○	Rd ← Rd × [EA]
Divide	DIV Rd,EA	58h Rd EA	73	73	71	73	71	70	72	72	--	○	○	Rd ← Rd / [EA]
Modulo	MOD Rd,EA	60h Rd EA	73	73	71	73	71	70	72	72	--	○	○	Rd ← Rd % [EA]
Multiply Long	MULL Rd,EA	680h Rd EA	57	57	55	57	55	54	56	56	--	○	○	(Rd+1,Rd) ← Rd × [EA]
Divide Long	DIVL Rd,EA	70h Rd EA	73	73	71	73	71	70	72	72	--	○	○	Rd ← (Rd+1,Rd) / [EA], Rd+1 ← (Rd+1,Rd) % [EA]
Shift Left Arithmetic	SHLA Rd,EA	80h Rd EA	8+n	8+n	6+n	8+n	6+n	5+n	7+n	7+n	--	○	○	Rd ← Rd << [EA]
Shift Left Logical	SHLL Rd,EA	88h Rd EA	8+n	8+n	6+n	8+n	6+n	5+n	7+n	7+n	--	○	○	Rd ← Rd << [EA]
Shift Right Arithmetic	SHRA Rd,EA	90h Rd EA	8+n	8+n	6+n	8+n	6+n	5+n	7+n	7+n	--	○	○	Rd ← Rd >> [EA]
Shift Right Logical	SHRL Rd,EA	98h Rd EA	8+n	8+n	6+n	8+n	6+n	5+n	7+n	7+n	--	○	○	Rd ← Rd >>> [EA]
Jump on Zero	JZ EA	A0h 0h EA	4/5	4/5	--	--	--	--	4/5	--	--	×	×	If (Z) PC ← EA
Jump on Carry	JC EA	A0h 1h EA	4/5	4/5	--	--	--	--	4/5	--	--	×	×	If (C) PC ← EA
Jump on Minus	JM EA	A0h 2h EA	4/5	4/5	--	--	--	--	4/5	--	--	×	×	If (S) PC ← EA
Jump on Overflow	JO EA	A0h 3h EA	4/5	4/5	--	--	--	--	4/5	--	--	×	×	if (V) PC ← EA
Jump on greater than	JGT EA	A0h 4h EA	4/5	4/5	--	--	--	--	4/5	--	--	×	×	If (not (Z or (S xor V))) PC ← EA
Jump on greater or equal	JGE EA	A0h 5h EA	4/5	4/5	--	--	--	--	4/5	--	--	×	×	if (not (S xor V)) PC ← EA
Jump on less or equal	JLE EA	A0h 6h EA	4/5	4/5	--	--	--	--	4/5	--	--	×	×	If (Z or (S xor V)) PC ← EA
Jump on less than	JLT EA	A0h 7h EA	4/5	4/5	--	--	--	--	4/5	--	--	×	×	If (S xor V) PC ← EA
Jump on Non Zero	JNZ EA	A0h 8h EA	4/5	4/5	--	--	--	--	4/5	--	--	×	×	If (not Z) PC ← EA
Jump on Non Carry	JNC EA	A0h 9h EA	4/5	4/5	--	--	--	--	4/5	--	--	×	×	If (not C) PC ← EA
Jump on Non Minus	JNM EA	A0h Ah EA	4/5	4/5	--	--	--	--	4/5	--	--	×	×	If (not S) PC ← EA
Jump on Non Overflow	JNO EA	A0h Bh EA	4/5	4/5	--	--	--	--	4/5	--	--	×	×	If (not V) PC ← EA
Jump on higher	JHI EA	A0h Ch EA	4/5	4/5	--	--	--	--	4/5	--	--	×	×	If (not (Z or C)) PC ← EA
Jump on lower or same	JLS EA	A0h Eh EA	4/5	4/5	--	--	--	--	4/5	--	--	×	×	If (Z or C) PC ← EA
Jump	JMP EA	A0h Fh EA	5	5	--	--	--	--	5	--	--	×	×	PC ← EA
Call subroutine	CALL EA	A8h 0h EA	6	6	--	--	--	--	6	--	--	×	×	[--SP] ← PC, PC ← EA
Input	IN Rd,EA	B0h Rd EA	7	--	--	--	--	--	6	6	--	×	×	Rd ← IO[EA]
Output	OUT Rd,EA	B8h Rd EA	6	--	--	--	--	--	5	5	--	×	×	IO[EA] ← Rd
Push Register	PUSH Rd	C0h Rd 0h	--	--	--	--	--	--	--	--	--	5	×	[--SP] ← Rd
Pop Register	POP Rd	C4h Rd 0h	--	--	--	--	--	--	--	--	--	6	×	Rd ← [SP++]
Return from Subroutine	RET	D0h 0h 0h	--	--	--	--	--	--	--	--	--	6	×	PC ← [SP++]
Return from Interrupt	RETI	D4h 0h 0h	--	--	--	--	--	--	--	--	--	9	×	FLAG ← [SP++], PC ← [SP++]
Enable Interrupt	EI	E0h 0h 0h	--	--	--	--	--	--	--	--	--	5	×	割り込み許可
Disable Interrupt	DI	E4h 0h 0h	--	--	--	--	--	--	--	--	--	5	×	割り込み禁止
Supervisor Call	SVC	F0h 0h 0h	--	--	--	--	--	--	--	--	--	12	×	システムコール
Halt	HALT	FFh 0h 0h	--	--	--	--	--	--	--	--	--	5	×	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	Drcr	OP Rd,Dsp	OP+0 Rd0h	Dsp	[Dsp]	Dsp番地
Indexed	Index	OP Rd,Dsp,Rx	OP+1 RdRx	Dsp	[Dsp+Rx]	(Dsp+Rxレジスタの内容) 番地
Immediate	Imm	OP Rd,#Imm	OP+2 Rd0h	Imm	Imm	Immそのもの
FP Rerative	FP Rlt	OP Rd,Dsp4,FP	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=Dsp
Register	Reg	OP Rd,Rs	OP+4 RdRs	--	Rs	Rsレジスタの内容
4bit Signed Immediate	Imm4	OP Rd,#Imm4	OP+5 RdI4	--	Imm4	I4を符号拡張した値そのもの
Register Indirect	Indr	OP Rd,0,Rx	OP+6 RdRx	--	[Rx]	Rxレジスタの内容番地
Byte Regsiter Indirect	B Indr	OP Rd,@Rx	OP+7 RdRx	--	[Rx]	Rxレジスタの内容番地 (但し番地の内容は 8 bit 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

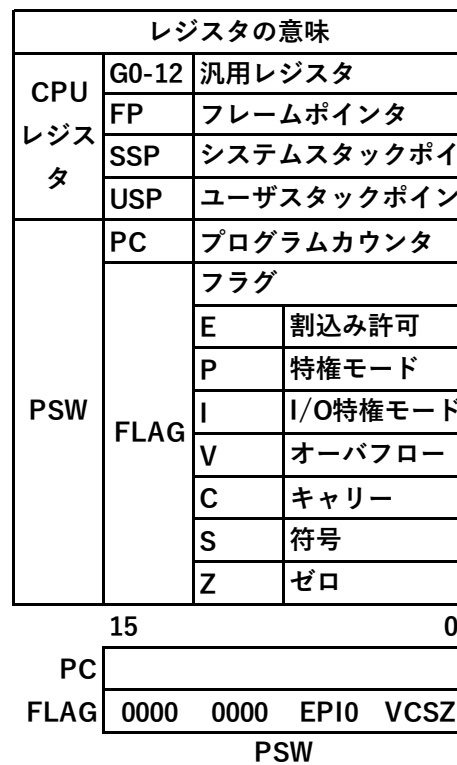
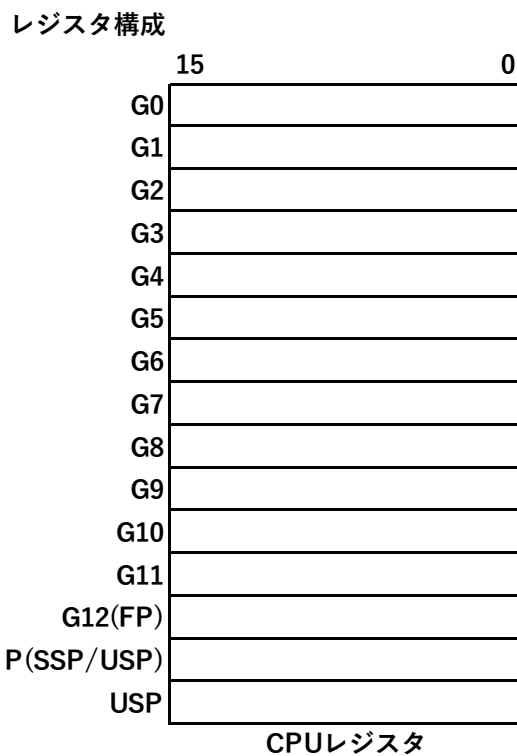
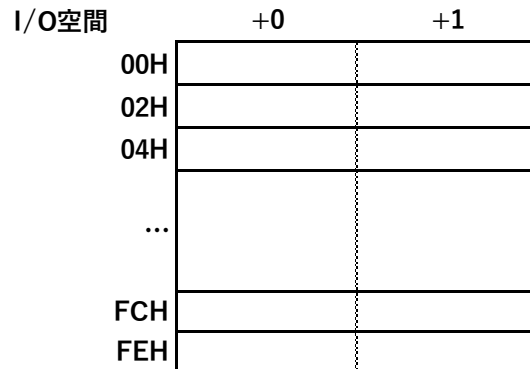
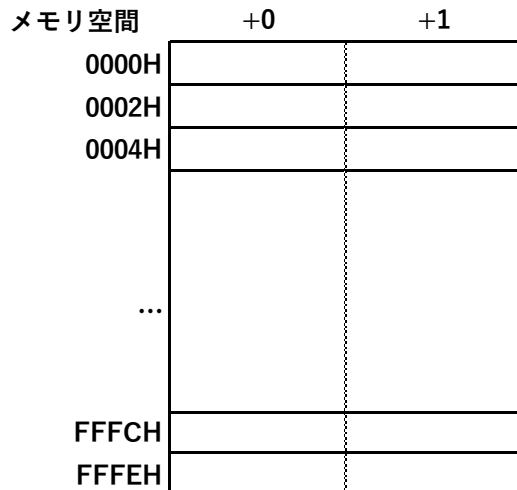
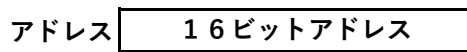
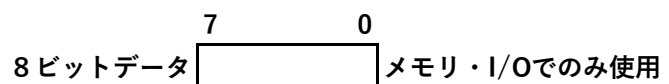
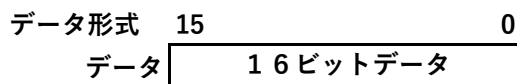
注 0 : フラグからレジスタへの転送命令、オペコードは14h

注 1 : MUL、DIV命令ではRdは偶数番号のレジスタ

注 2 : D4はDsp4(4bitディシプレースメント)の1/2の値

注 3 : I4はImm4 (4 bit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



# TaC命令フォーマット

Ver.8.8.4

2019/2/10

ダイレクト(\*0)

OP	Rd	0H	Dsp
----	----	----	-----

ショートイミディエイト(\*5)

OP	Rd	Imm4
----	----	------

インデクスド(\*1)

OP	Rd	Rx	Dsp
----	----	----	-----

レジスタインダイレクト(\*6)

OP	Rd	Rx
----	----	----

イミディエイト(\*2)

OP	Rd	0H	Imm
----	----	----	-----

バイト・レジスタインダイレクト(\*7)

OP	Rd	Rx
----	----	----

FP相対(\*3)

OP	Rd	Dsp4
----	----	------

レジスタ(\*8)

OP	Rd	0H
----	----	----

レジスタレジスタ(\*4)

OP	Rd	Rs
----	----	----

オペランドなし(\*9)

OP	00H
----	-----

OP

		OP 下位3ビット							
		0	1	2	3	4	5	6	7
OP 上位5ビット	00000	NO(*9)							
	00001	LD(*0)	LD(*1)	LD(*2)	LD(*3)	LD(*4)	LD(*5)	LD(*6)	LD(*7)
	00010	ST(*0)	ST(*1)		ST(*3)	LD(*8)※1		ST(*6)	ST(*7)
	00011	ADD(*0)	ADD(*1)	ADD(*2)	ADD(*3)	ADD(*4)	ADD(*5)	ADD(*6)	ADD(*7)
	00100	SUB(*0)	SUB(*1)	SUB(*2)	SUB(*3)	SUB(*4)	SUB(*5)	SUB(*6)	SUB(*7)
	00101	CMP(*0)	CMP(*1)	CMP(*2)	CMP(*3)	CMP(*4)	CMP(*5)	CMP(*6)	CMP(*7)
	00110	AND(*0)	AND(*1)	AND(*2)	AND(*3)	AND(*4)	AND(*5)	AND(*6)	AND(*7)
	00111	OR(*0)	OR(*1)	OR(*2)	OR(*3)	OR(*4)	OR(*5)	OR(*6)	OR(*7)
	01000	XOR(*0)	XOR(*1)	XOR(*2)	XOR(*3)	XOR(*4)	XOR(*5)	XOR(*6)	XOR(*7)
	01001	ADDS(*0)	ADDS(*1)	ADDS(*2)	ADDS(*3)	ADDS(*4)	ADDS(*5)	ADDS(*6)	ADDS(*7)
	01010	MUL(*0)	MUL(*1)	MUL(*2)	MUL(*3)	MUL(*4)	MUL(*5)	MUL(*6)	MUL(*7)
	01011	DIV(*0)	DIV(*1)	DIV(*2)	DIV(*3)	DIV(*4)	DIV(*5)	DIV(*6)	DIV(*7)
	01100	MOD(*0)	MOD(*1)	MOD(*2)	MOD(*3)	MOD(*4)	MOD(*5)	MOD(*6)	MOD(*7)
	01101	MULL(*0)	MULL(*1)	MULL(*2)	MULL(*3)	MULL(*4)	MULL(*5)	MULL(*6)	MULL(*7)
	01110	DIVL(*0)	DIVL(*1)	DIVL(*2)	DIVL(*3)	DIVL(*4)	DIVL(*5)	DIVL(*6)	DIVL(*7)
	01111								
	10000	SHLA(*0)	SHLA(*1)	SHLA(*2)	SHLA(*3)	SHLA(*4)	SHLA(*5)	SHLA(*6)	SHLA(*7)
	10001	SHLL(*0)	SHLL(*1)	SHLL(*2)	SHLL(*3)	SHLL(*4)	SHLL(*5)	SHLL(*6)	SHLL(*7)
	10010	SHRA(*0)	SHRA(*1)	SHRA(*2)	SHRA(*3)	SHRA(*4)	SHRA(*5)	SHRA(*6)	SHRA(*7)
	10011	SHRL(*0)	SHRL(*1)	SHRL(*2)	SHRL(*3)	SHRL(*4)	SHRL(*5)	SHRL(*6)	SHRL(*7)
	10100	JMP(*0)	JMP(*1)					JMP(*6)	
	10101	CALL(*0)	CALL(*1)					CALL(*6)	
	10110	IN(*0)						IN(*6)	IN(*7)
	10111	OUT(*0)						OUT(*6)	OUT(*7)
	11000	PUSH(*8)				POP(*8)			
	11001								
	11010	RET(*9)				RETI(*9)			
	11011								
	11100	EI(*9)				DI(*9)			
	11101								
	11110	SVC(*9)							
	11111								HALT(*9)

特権命令

※1：フラグからレジスタへの転送命令

	>	>=	=	!=	<=	<
符号あり	JGT	JGE	JZ	JNZ	JLE	JLT
符号無し	JHI	JNC	JZ	JNZ	JLS	JC

FLAGのビット割り  
(00000000EP00VCSZ)

Rd/Rs/Rx	
値	意味
0	G0
1	G1
2	G2
3	G3
4	G4
5	G5
6	G6
7	G7
8	G8
9	G9
A	G10
B	G11
C	G12(FP)
D	SP(SSP/USP)
E	USP
F	PC

SPの意味はPフラグで変化

JMP命令のRd	
値	意味
0	JZ
1	JC
2	JM
3	JO
4	JGT
5	JGE
6	JLE
7	JLT
8	JNZ
9	JNC
A	JNM
B	JNO
C	JHI
D	
E	JLS
F	JMP

# TaCメモリ空間、入出力空間

Ver.8.8.4  
2019/2/10

## メモリマップ

+0番地		+1番地		
0000h	RAM(56kB)			RAM
0002h				
0004h				
...				
DFFEh				
E000h	予約	VRAM(2kB)		
...	(アトリビュート)			
EF FEh				
F000h	IPL(4064B)			ROM
...				
FF DEh				
FFE0h	Timer0			割り込みベクタ
FFE2h	Timer1			
FFE4h	INT2			
FFE6h	INT3			
FFE8h	SIO 受信			
FFEAh	SIO 送信			
FF ECh	PS2 受信			
FF EEh	PS2 送信			
FFF0h	uSD			
FFF2h	ADC			
FFF4h	不正 (奇数) アドレス			
FFF6h	上下限アドレス違反			
FFF8h	ゼロ除算(※1)			
FFFAh	特権違反 (※1)			
FFFCh	未定義命令 (※1)			
FF FEh	SVC (※1)			

※1：マイクロプログラムにより発生

## I/Oマップ

+0番地		+1番地	
00h	Timer0(In:現在値/Out:周期)		I/O装置
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)		
08h	00H	SIO-Data	
0Ah	00H	SIO-Stat/Ctrl	
0Ch	00H	PS2-Data	
0Eh	00H	PS2-Stat/Ctrl	
10h	00H	uSD-Stat/Ctrl	
12h	00H	uSD-MemAddr	
14h	00H	uSD-BlkAddrH	
16h	00H	uSD-BlkAddrL	
18h	00H	拡張ポート(In/Out)	
1Ah	00H	ADC参照電圧(Out)	
1Ch	00H	拡張ポートHi(Out)	
1Eh	00H	モード(In)	
20h	00H	ACD予約	
...	...	...	
28h	00H	RN4020予約	
...	...	...	
30h	00H	TeC(DLed)	
32h	00H	TeC(DSw)	
34h	00H	Tec(Fnc)	
36h	00H	TeC(Ctl)	
38h	00H	00H	
...	...	...	
F0h	00H	b0=Enable MMU	
F2h	00H	00H	
F4h	ベースレジスタ(Out)/0000H(IN)		
F6h	リミットレジスタ(Out)/0000H(IN)		
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)	
			コンソール

※2：拡張ポートHi (M000 VVVV)

M(0:入力, 1:出力), V 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
04h	Timer1 コントロール	I000 ... 000S	I=Enable Interrupt, S=Start
11h	uSD-Ctrl	E000 0IRW	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
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

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

TeCコンソールI/Oアドレス		
	Read	Write
Dled (30h)	データランプ	空き
Dsw (32h)	00H	データスイッチ
Fnc (34h)	00H	ABCD EFGH
Ctl (36h)	---- --RS	I--- -JKL

R=Reset-SW(IN), S=SETA-SW(IN)