Cortex-M0 ARMマシン語表（抜粋）

代入	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	cycles
Rd = u8	0	0	1	0	0	Rd			u8								1
Rd = Rm	0	1	0	0	0	1	1	0	0	0	Rm			Rd			1
Rd = Rm	0	1	0	0	0	1	1	0	Rd3	Rm				Rd2-0			1 or 3

※Rd3とRd2-0の4bitでRdを指定する、Rd=PCの時3cycles

演算	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	cycles
Rd = Rd + u8	0	0	1	1	0	Rd			u8								1
Rd = Rd - u8	0	0	1	1	1	Rd			u8								1
Rd = Rm << u5	0	0	0	0	0	u5					Rm			Rd			1
Rd = Rm >> u5	0	0	0	0	1	u5					Rm			Rd			1
Rd = Rn + Rm	0	0	0	1	1	0	0	Rm			Rn			Rd			1
Rd = Rd + Rm	0	1	0	0	0	1	0	0	Rd3	Rm				Rd2-0			1 or 3
Rd = Rn - Rm	0	0	0	1	1	0	1	Rm			Rn			Rd			1
Rd = Rn + u3	0	0	0	1	1	1	0	u3			Rn			Rd			1
Rd = Rn - u3	0	0	0	1	1	1	1	u3			Rn			Rd			1
Rd = Rd & Rm	0	1	0	0	0	0	0	0	0	0	Rm			Rd			1
Rd = Rd ^ Rm	0	1	0	0	0	0	0	0	0	1	Rm			Rd			1
Rd = Rd << Rs	0	1	0	0	0	0	0	0	1	0	Rs			Rd			1
Rd = Rd >> Rs	0	1	0	0	0	0	0	0	1	1	Rs			Rd			1
Rd = -Rm	0	1	0	0	0	0	1	0	0	1	Rm			Rd			1
Rd = Rd | Rm	0	1	0	0	0	0	1	1	0	0	Rm			Rd			1
Rd = Rd * Rm	0	1	0	0	0	0	1	1	0	1	Rm			Rd			1
Rd = Rd & ~Rm	0	1	0	0	0	0	1	1	1	0	Rm			Rd			1
Rd = ~Rm	0	1	0	0	0	0	1	1	1	1	Rm			Rd			1
Rd = rev(Rm)	1	0	1	1	1	0	1	0	0	0	Rm			Rd			1
Rd = rev16(Rm)	1	0	1	1	1	0	1	0	0	1	Rm			Rd			1
Rd = revSH(Rm)	1	0	1	1	1	0	1	0	1	1	Rm			Rd			1

※rev:byteオーダー反転、rev16:byteオーダー反転(2byteずつ)、revSH:符号付き16bitを反転32bit化

※Rd=PCの時3cycles

メモリアクセス	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	cycles
Rd = [Rn + u5]	0	1	1	1	1	u5					Rn			Rd			2
Rd = [Rn + u5]2	1	0	0	0	1	u5/2					Rn			Rd			2
Rd = [Rn + u5]4	0	1	1	0	1	u5/4					Rn			Rd			2
[Rn + u5] = Rd	0	1	1	1	0	u5					Rn			Rd			2
[Rn + u5]2 = Rd	1	0	0	0	0	u5/2					Rn			Rd			2
[Rn + u5]4 = Rd	0	1	1	0	0	u5/4					Rn			Rd			2
Rd = [Rn + Rm]	0	1	0	1	1	1	0	Rm			Rn			Rd			2
Rd = [Rn + Rm]2	0	1	0	1	1	0	1	Rm			Rn			Rd			2
Rd = [Rn + Rm]4	0	1	0	1	1	0	0	Rm			Rn			Rd			2
[Rn + Rm] = Rd	0	1	0	1	0	1	0	Rm			Rn			Rd			2
[Rn + Rm]2 = Rd	0	1	0	1	0	0	1	Rm			Rn			Rd			2
[Rn + Rm]4 = Rd	0	1	0	1	0	0	0	Rm			Rn			Rd			2

条件判断	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	cycles
Rn - u8	0	0	1	0	1	Rn			u8								1
Rn - Rm	0	1	0	0	0	0	1	0	1	0	Rm			Rn			1
Rn & Rm	0	1	0	0	0	0	1	0	0	0	Rm			Rn			1

分岐	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	cycles
IF 0 GOTO n8 << 1	1	1	0	1	0	0	0	0	n8								1 or 3
IF !0 GOTO n8 << 1	1	1	0	1	0	0	0	1	n8								1 or 3
GOTO n11 << 1	1	1	1	0	0	n11											3
GOTO Rd	0	1	0	0	0	1	1	1	0	Rm				0	0	0	3
RET (=#4770)	0	1	0	0	0	1	1	1	0	1	1	1	0	0	0	0	3
CALL Rd	0	1	0	0	0	1	1	1	1	Rm				0	0	0	3

※GOTOに指定する値(n8/n11)は飛ばしたい差分から更に-2した数を指定する、分岐するとき3cycles

スタック	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	cycles
PUSH regs	1	0	1	1	0	1	0	LR	R7	R6	R5	R4	R3	R2	R1	R0	1+N
POP regs	1	0	1	1	1	1	0	PC	R7	R6	R5	R4	R3	R2	R1	R0	1+N

※N:指定したレジスタの数、例）PUSH R5,R4,R3

その他	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	cycles
CPSID (=#B672)	1	0	1	1	0	1	1	0	0	1	1	1	0	0	1	0	1
CPSIE (=#B662)	1	0	1	1	0	1	1	0	0	1	1	0	0	0	1	0	1
WFI (=#BF30)	1	0	1	1	1	1	1	1	0	0	1	1	0	0	0	0	2
NOP (=0)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

※CPSID:割込禁止、CPSIE:割込許可、WIF:割込待ち、NOP:なにもしない(no operation) R0=R0<<0

- 連載、IchigoJamではじめる、ARMマシン語入門

1. はじめてのマシン語
2. ハンドアセンブルで超速計算！
3. マシン語メモリアクセスで画面超速表示！
4. マシン語でLEDを光らせよう！
5. 楽しさ広がるマルチバイトメモリアクセスとスタック

DATA: Cortex-M0プロセッサ - ARM