マシン語はそのコードを逆アセンブルして命令を見ることができる。ＣＡＳＬはその仕様書１．１－（６）、「命令語の構成」の説明でその構成を定義しないとしている。だから自分が解いた練習問題を実行して結果を確かめることができない。このことは、受験者にとって少しだけ不満を呈するものである。

しかし、そこを逆手にとって自分なりの命令構成を仕立てあげる事もできる。

§1 Jacuzzi流ＣＡＳＬ命令語の構成!!

1. 確定事項（恒常的条件）

ａｄｒ（オペランド中の）は16ビットである。且つ、命令の第２ワード目である。

2. 最低限度（必要十分条件）

ＧＲとＸＲはそれぞれ３ビット必要で、第１ワードの中に存在する。

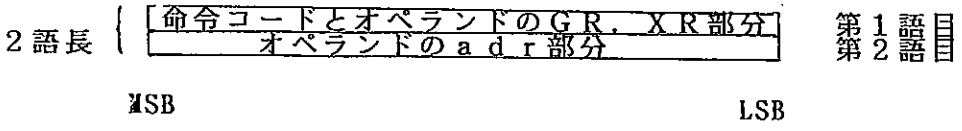
3. 単純計算

３２－（１６＋２×３）＝１０

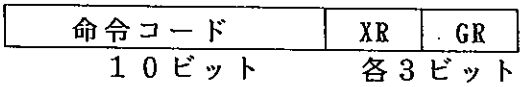
１０ビットが命令コードとして使える。　実際には２３しか命令がないから、５ビットで十分。ＧＲやＸＲの冗長部を考慮すると命令の「ａｄｒ」の部分を除いて８ビットで足るのではないかと思う。

§2 構成の骨子

1. 前章の条件より、ＣＡＳＬ仕様書に示さなくてはいけない命令の語長（２語長）は次のように分ける。



2. 第１語（ワード）目のビット割り当て

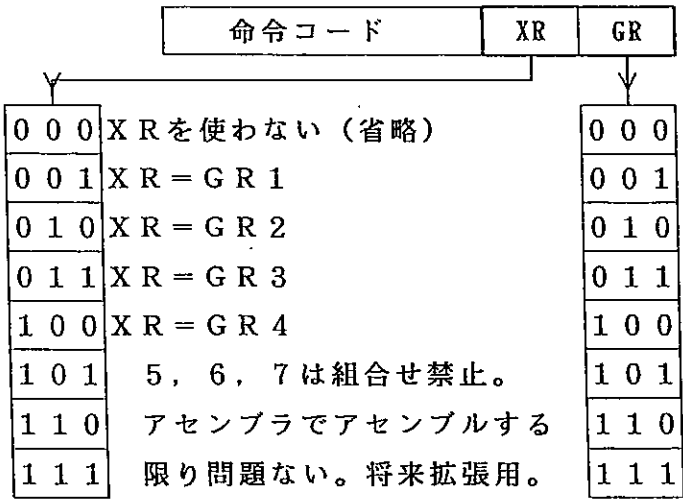


基本命令は２３しかないので５ビットで十分だが、１０ビットすべてをビットごとに意味を持たせ、命令コード部の構成を行った。

各ビットは試行錯誤の結果、次の章で解説する通りである。

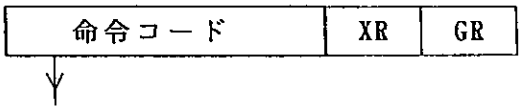
§3 第1ワード目の解説

1. 汎用レジスタ指定部



0 0 0	XRを使わない（省略）	0 0 0	GR0
0 0 1	XR＝GR1	0 0 1	GR1
0 1 0	XR＝GR2	0 1 0	GR2
0 1 1	XR＝GR3	0 1 1	GR3
1 0 0	XR＝GR4	1 0 0	GR4
1 0 1	5，6，7は組合せ禁止。	1 0 1	5，6，7は組合せ禁止。
1 1 0	アセンブラでアセンブルする	1 1 0	XRの場合と同様。
1 1 1	限り問題ない。将来拡張用。	1 1 1	

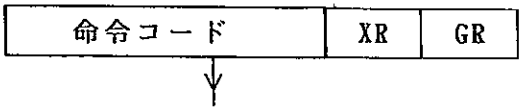
2. 命令コード部（命令系）



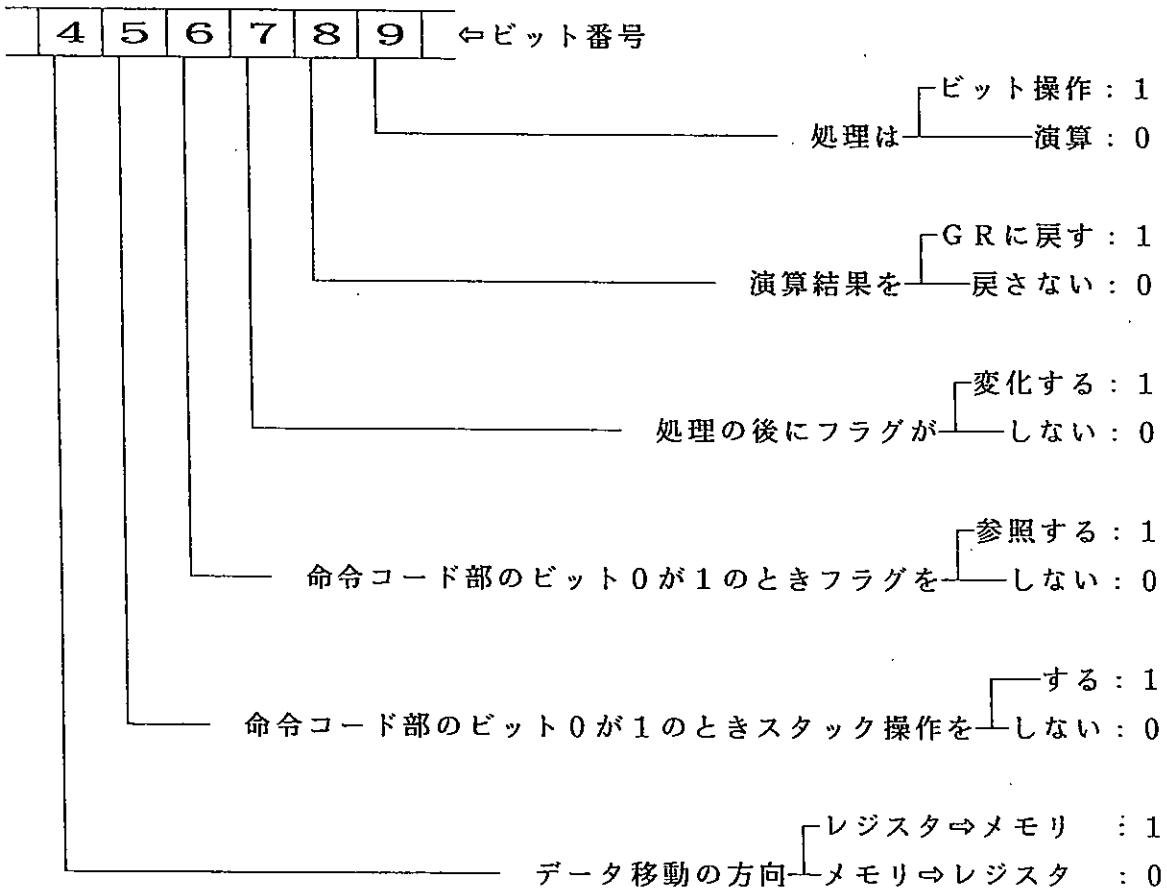
MSB から第3ビットまでの4ビットについてのビット割り当て。

0 0 0 0	LD，ST，PUSH，POP	データ移動系1
0 0 0 1	LEA	データ移動系2
0 0 1 0	ADD，SLA	
0 0 1 1	SUB，SRA，CPA	算術演算系
0 1 0 0	AND	
0 1 0 1	OR	
0 1 1 0	EOR，SLL	論理演算系
0 1 1 1	CPL，SRL	
1 0 0 0	JMP	
1 0 0 1	未定義　INTなどの隠し命令だったりする	
1 0 1 0	未定義	
1 0 1 1	CALL，RET	プログラム制御系
1 1 0 0	JPZ	
1 1 0 1	JMI	
1 1 1 0	JNZ	
1 1 1 1	JZE	

3. 命令コード部（処理指示部）



第４ビットから第９ビットまでの残り６ビットについてのビット割り当て。



4　補足説明

ＧＲのない命令ではＣＡＳＬはＸＲ，ＧＲともに０００をビット列を生成する。ハンドアセンブルするときは、これに従う方が精神衛生上好ましい。　学校での専門科目の講義や実習以外そんなことはありえないと思うが・・・

命令の体系は見て分かるように、ビット０が１ならばＰＣが処理の対象となり、そうでなければＧＲが処理の対象となるような命令が定義されている。

またＧＲ処理命令系でビット１が１なら論理演算系、０なら算術演算系となっていることに気づいてくれただろうか。Jac 4202 が勝手にそう定義したのだが）

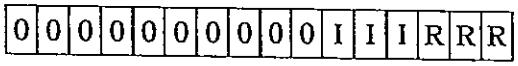
命令によって該当しないビットは×で示し、ＣＡＳＬは０のビットを生成する。

（1）ロード・ストア

LD GR, adr[,XR]

命令: LD GR4,#FFF0

ｺｰﾄﾞ: $0004FFF0_{16}$



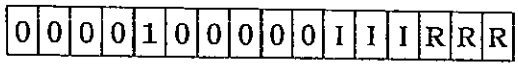
RRR＝１００

ＩＩＩ＝０００

ST GR, adr[,XR]

命令: ST GR3,#0000,GR2

ｺｰﾄﾞ: 08130000_{16}



RRR＝０１１

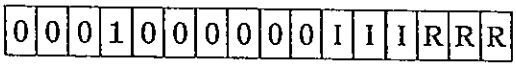
ＩＩＩ＝０１０

（2）ロードアドレス（ロードイミディエート）

LEA GR, adr[,XR]

命令: LEA GR2,'A'

ｺｰﾄﾞ: 10020041_{16}



RRR＝０１０

ＩＩＩ＝０００　（第2ﾜｰﾄﾞ目＝#0041）

（3）算術・論理演算

ADD GR, adr[,XR]

命令: ADD GR1,32767

ｺｰﾄﾞ: $21817FFF_{16}$

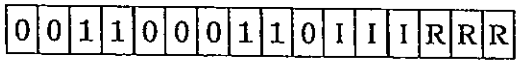
RRR＝００１

ＩＩＩ＝０００　（第2ﾜｰﾄﾞ目＝#7FFF）

SUB GR, adr[,XR]

命令: SUB GR0,-1,GR4

ｺｰﾄﾞ: $31A0FFFF_{16}$



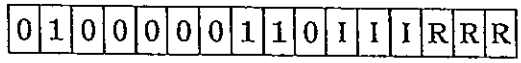
RRR＝０００

ＩＩＩ＝１００　（第2ﾜｰﾄﾞ目＝#FFFF）

AND GR, adr[,XR]

命令: AND GR2,MASK

ｺｰﾄﾞ: 41802345_{16}



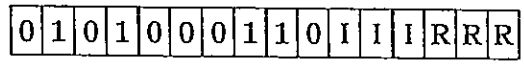
RRR＝０１０

ＩＩＩ＝０００　（MASK=#2345とする）

OR GR, adr[,XR]

命令: OR GR4,OFFSET,GR3

ｺｰﾄﾞ: $519C0000_{16}$



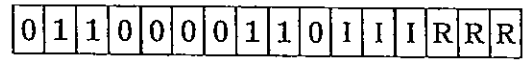
RRR＝１００

ＩＩＩ＝０１１（OFFSET=#0000とする）

EOR GR, adr[,XR]

命令: EOR GR4,TEST

ｺｰﾄﾞ: $6183CDEF_{16}$



RRR＝０１１

ＩＩＩ＝０００　（TEST=#CDEFとする）

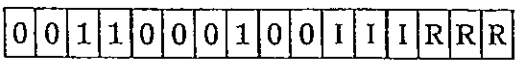
注意）論理演算で使ったラベルは全て、番地を示すもので即値ではない。

（４）比較演算

ＣＰＡ　ＧＲ，ａｄｒ[,ＸＲ]

命令:　CPA GR0,-1,GR4

ｺｰﾄﾞ:　３１４０ＦＦＦＦ$_{16}$



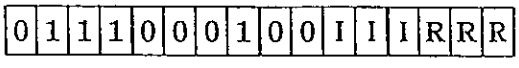
ＲＲＲ＝０００

ＩＩＩ＝１００　（第2ﾜｰﾄﾞ目＝#FFFF）

ＣＰＬ　ＧＲ，ａｄｒ[,ＸＲ]

命令:　CPL GR0,#3388

ｺｰﾄﾞ:　７１００３３８８$_{16}$



ＲＲＲ＝０００

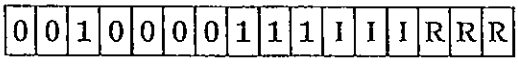
ＩＩＩ＝０００

（５）シフト演算

ＳＬＡ　ＧＲ，ａｄｒ[,ＸＲ]

命令:　SLA GR1,32767

ｺｰﾄﾞ:　２１Ｃ１７ＦＦＦ$_{16}$



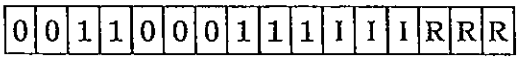
ＲＲＲ＝００１

ＩＩＩ＝０００　（第2ﾜｰﾄﾞ目＝#7FFF）

ＳＲＡ　ＧＲ，ａｄｒ[,ＸＲ]

命令:　SRA GR0,-1,GR4

ｺｰﾄﾞ:　３１Ｅ０ＦＦＦＦ$_{16}$



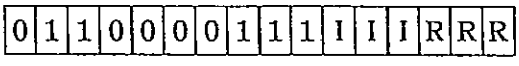
ＲＲＲ＝０００

ＩＩＩ＝１００　（第2ﾜｰﾄﾞ目＝#FFFF）

ＳＬＬ　ＧＲ，ａｄｒ[,ＸＲ]

命令:　SLL GR4,REPEAT

ｺｰﾄﾞ:　６１Ｃ３０００４$_{16}$



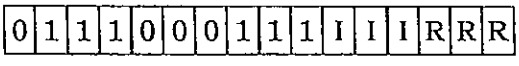
ＲＲＲ＝０１１

ＩＩＩ＝０００（REPEAT=#0004とする）

ＳＲＬ　ＧＲ，ａｄｒ[,ＸＲ]

命令:　SRL GR2,0,GR3

ｺｰﾄﾞ:　７１ＤＡ０００$_{16}$



ＲＲＲ＝０１０

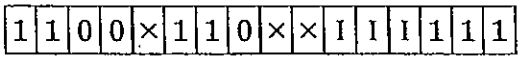
ＩＩＩ＝０１１

（６）分岐

ＪＰＺ　ＧＲ，ａｄｒ[,ＸＲ]

命令:　JPZ BUNKI

ｺｰﾄﾞ:　Ｃ６０７ＦＦＦ０$_{16}$



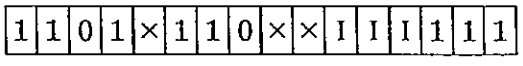
(BUNKI=#FFF0とする)

ＩＩＩ＝０００

ＪＭＩ　ＧＲ，ａｄｒ[,ＸＲ]

命令:　JMI OFFSET,GR1

ｺｰﾄﾞ:　Ｄ６０Ｆ０１００$_{16}$



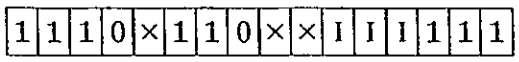
(OFFSET=#0100とする)

ＩＩＩ＝００１

ＪＮＺ　ＧＲ，ａｄｒ[,ＸＲ]

命令:　JNZ BUNKI

ｺｰﾄﾞ:　Ｅ６０７ＦＦＦ０$_{16}$



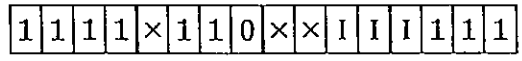
(BUNKI=#FFF0とする)

ＩＩＩ＝０００

ＪＺＥ　ＧＲ，ａｄｒ[,ＸＲ]

命令:　JZE OFFSET,GR3

ｺｰﾄﾞ:　Ｆ６１Ｆ０１００$_{16}$



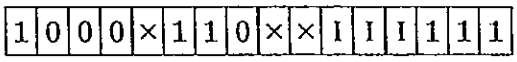
(OFFSET=#0100とする)

ＩＩＩ＝０１１

ＪＭＰ　ＧＲ，ａｄｒ[,ＸＲ]

命令:　JMP BUNKI,GR2

ｺｰﾄﾞ:　８６１７ＦＦＦ０$_{16}$



(BUNKI=#FFF0とする)

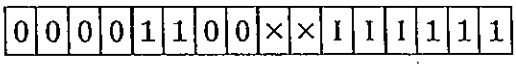
ＩＩＩ＝０１０

（７）スタック操作

ＰＵＳＨ　ａｄｒ[,ＸＲ]

命令:　PUSH 0,GR1

ｺｰﾄﾞ:　０Ｃ０Ｆ０００００$_{16}$

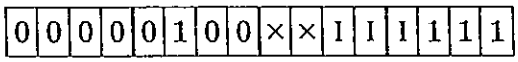


ＩＩＩ＝００１

ＰＯＰ　ＧＲ

命令:　POP GR3

ｺｰﾄﾞ:　０４１Ｆ××××$_{16}$



（１ワードで済む命令）

ＩＩＩ＝０１１

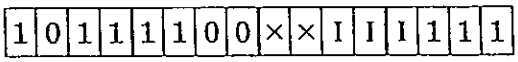
第２ワード目に隠しアドレスをセットしておき、プログラムの別の場所でこの命令の第１ワードを書き変え（＝自分で開腹手術をして）、その後ここが実行されるとスペシャルファンクションを実行する。
てなことも、アセンブラしか知らない者にはできないが、マシンコードを知っていると、暴走しない２重人格プログラムが書けるかもね。

（８）コール・リターン

ＣＡＬＬ　ａｄｒ[,ＸＲ]

命令:　CALL SUB1

ｺｰﾄﾞ:　ＢＣ１７ＦＦＦ０$_{16}$



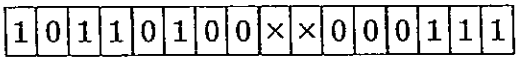
(SUB1=#FFF0とする)

ＩＩＩ＝０１０

ＲＥＴ

命令:　RET

ｺｰﾄﾞ:　Ｂ４００××××$_{16}$



（１ワード以下で済む命令）

ＰＯＰ命令と同じトリックができる