M16C Befehlsübersicht

Mov:

LDC src,dest (16bit) (rs4)	- (FLG !)
LDCTX abs16,abs20 (Context)	-
LDE.B.W src,dest (extra far) (rs3) ♣	SZ
LDINTB src (20bit)	-
LDIPL src (Int.Level)	-
MOV.B.W src,dest	SZ
MOVA src,dest (16bit)	-
MOV{HH,LL,HL,LH} src,dest (rs2)	-
POP.B.W dest	-
PUSH.B.W src	-
POPC dest [FB,SB,SP,ISP,FLG,INTBH,L]	-
PUSHC src [FB,SB,SP......]	-
POPM dest [A0,A1,R0,R1,R2,R3]	-
PUSHM src	-
SMOVB.B.W (String move backwards)	-
SMOVF.B.W (String move forward)	-
SSTR.B.W (String store)	-
STC src,dest (16bit) (rs4)	-
STCTX abs16,abs20 (Context)	-
STE.B.W src,dest (extra far) (rs3) ♣	SZ
XCHG.B.W src,dest (r4)	-

Register Satz rs1:

.B	R0L,R0H,dsp:8[SB],dsp[FB],abs16
.W	A0,A1

Register Satz rs2:

src	*dest*
R0L	R0L,R0H,R1L,R1H,[A0],[A1],dsp:[A0],dsp:[A1],dsp:[SB],dsp:[FB],abs16
R0L,R0H,R1L,R1H,[A0],[A1],dsp:[A0],dsp:[A1],dsp:[SB],dsp:[FB],abs16	R0L

Register Satz rs3:

src	*dest*
dsp:20[A0],abs20,[A1A0] *(LDE.B.W)*	R0L,R0H,R1L,R1H,R0,R1,R2,R3,A0,A1,[A0],[A1],abs16,dsp:8[X],dsp:16[X] *X=A0,A1,SB,FB*
R0L,R0H,R1L,R1H,R0,R1,R2,R3,A0,A1,[A0],[A1],abs16,dsp:8[X],dsp:16[X] *X=A0.A1,SB,FB*	dsp:20[A0],abs20,[A1A0] *(STE.B.W)*

Register Satz rs4:

	FLG,FB,SB,SP(U-Flag),ISP,INTBH,INTBL
	U-Flag: 0= ISP 1=USP

Condition {cond}:

GEU	C=1	Equal or greater than
EQ	Z=1	Equal to
GTU	C^/Z=1	Greater than
PZ	S=0	Positiv or zero
GE	SAO=0	Equal or greater than (signed)
GT	S,O,Z	Greater than (signed)
O	O=1	O flag = 1
LTU	NC C=0	Smaller than
NE	NZ Z=0	Not equal
LEU	C^/Z=0	Equal to or smaller than
N	S=1	Negativ
LE	S,O,Z	Equal to or smaller than (signed)
LT	SAO=1	Smaller than (signed)
NO	O=0	O flag = 0

Sascha Pypke (saschaspeedy@hotmail.com)

Bit Operation:

BAND src (C)	C
BCLR dest	-
BNAND src (C)	C
BNOR stc (C)	C
BNOT dest	-
BNXOR src (C)	C
BOR src (C)	C
BSET dest	-
BXOR src (C)	C
FCLR dest (UIOBSZDC)	UIOBSZDC
FSET dest (UIOBSZDC)	UIOBSZDC

Logic:

AND.B.W src,dest		SZ
NOT.B.W dest (XOR FFh)		SZ
OR.B.W src,dest		SZ
XOR.B.W src,dest		SZ
ROLC.B.W dest (C←MSB←LSB←C)		SZC
RORC.B.W dest (C→MSB→LSB→C)		SZC
ROT.B.W src,dest (⟳)	↓	SZC
SHA.B.W.L src,dest (x=>C)	(+x = left shift)	OSZC
SHL.B.W.L src,dest (0=>C)	(-x = right shift)	SZC

Condition:

BM{cond} dest (bit move , true = 1)	C if dest
BNTST src (Z,C) (bit not test)	/Z /C
BTST src (Z,C) (bit test)	/Z C
BTSTC dest (bit test and clear)	/Z C
BTSTS dest (bit test and set)	/Z C
CMP.B.W src,dest (compare auf dest)	OSZC
J{cond} Label (+128 -127)	-
STNZ im8,dest (Store on not zero)	-
STZ im8,dest (Store on zero)	-
STZX im8[src1 Z],im8[src2 NZ],dest	-
TST.B.W src,dest (Bitweise)	SZ

Loop:

ADJNZ.B.W src,dest,label (+128 -127)	-
SBJNZ.B.W src,dest,label (+128 -127)	-

Sonstige:

ABS.B.W dest	OSZC
EXTS.B.W dest	SZ
DEC.B.W dest (rs1)	SZ
INC.B.W dest (rs1)	SZ
NOP	-

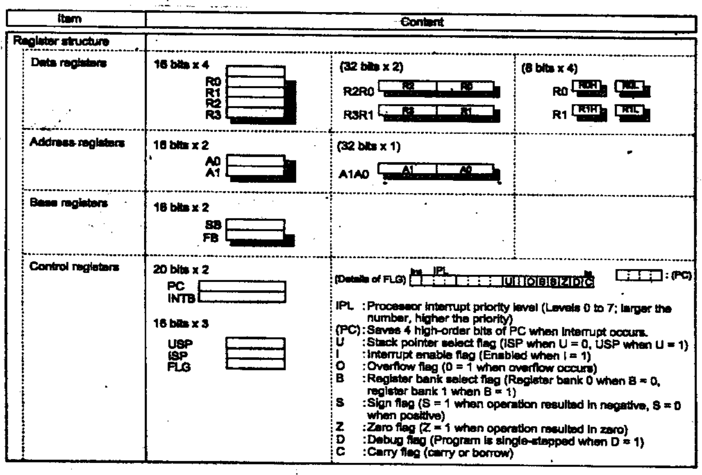
Mathe:

ADC.B.W src,dest	OSZC
ADCF.B.W dest	OSZC
ADD.B.W src,dest	OSZC
SBB.B.W src.dest	OSZC
SUB.B.W src,dest	OSZC
DIV.B.W src (dest=R0L,R0H,R2R0)	O
DIVU.B.W src (dest=R0L,R0H,R2R0)	O
DIVX.B.W src (dest=R0L,R0H,R2R0)	O
MUL.B.W src,dest	-
MULU.B.W src,dest	-
NEG dest (0-dest)	OSZC
RMPA.B.W (sonder)	O
DADC.B.W src,dest(im|R0H|R1,R0L|R0)	SZC
DADD.B.W src,dest(im|R0H|R1,R0L|R0)	SZC
DSBB.B.W src,dest(im|R0H|R1,R0L|R0)	SZC
DSUB.B.W src,dest(im|R0H|R1,R0L|R0)	SZC

Jump:

BRK (BRK Interrupt)	UID = 0
ENTER #bytes	- (FB,SP)
EXITD	- (FB,SP)
INT src (src=0-31 U=0 | src=32-63 U=1)	UID
INTO	UID = 0
JMP.S.B.W.A Label	-
JMPI.W.A src (.A = A1A0,R2R0,R3R1)	-
JMPS.W.A src ♣	-
JSR.W.A Label	-
JSRI.W.A src	-
JSRS src ♣	-
REIT	(FLG)
RTS	-
UND	UID = 0
WAIT	-

♣ Einschränkung bei R8C



IPL :Processor interrupt priority level (Levels 0 to 7; larger the number, higher the priority)
(PC):Saves 4 high-order bits of PC when interrupt occurs.
U :Stack pointer select flag (ISP when U = 0, USP when U = 1)
I :Interrupt enable flag (Enabled when I = 1)
O :Overflow flag (0 = 1 when overflow occurs)
B :Register bank select flag (Register bank 0 when B = 0, register bank 1 when B = 1)
S :Sign flag (S = 1 when operation resulted in negative, S = 0 when positive)
Z :Zero flag (Z = 1 when operation resulted in zero)
D :Debug flag (Program is single-stepped when D = 1)
C :Carry flag (carry or borrow)