```
\ ------------------------------------------------------------
\ 6809 assembler: utilities             (c) 14 11 85 BJR
\
VOCABULARY ASSEMBLER IMMEDIATE  ASSEMBLER DEFINITIONS  HEX
: WITHIN   ROT SWAP OVER   \ n lo hi -- f | test within limits
   < ROT ROT > OR 0= ;
: 8BIT?   -80  7F  WITHIN ;
: 5BIT?   -10 0F WITHIN ;
: (,)  , ;
\ : ALIGN ;  ( for byte-addressing hosts)

\ Buffoonery to allow byte-oriented assembly on word machines.
\ Assembler does ALIGNing, thus always knows when to de-ALIGN!
\ Include if host is word-aligned.
0 VARIABLE ALIGNED         \ flag indicating HERE was adjusted
: ALIGN    HERE 1 AND DUP ALLOT ALIGNED ! ;     \ align HERE
: DEALIGN  ALIGNED @ MINUS ALLOT  0 ALIGNED ! ; \ cancel ALIGN
: HERE     HERE ALIGNED @ - ;  \ true HERE for address calc'n
: C,   DEALIGN C, ALIGN ;       \ redefine C, for assembler

\ ------------------------------------------------------------
\   6809 assembler: addressing modes    (c) 03 06 85 BJR
\
: W,   DUP >< C, C, ;      \ store word as hibyte, lobyte
: OPCODE,                  \ store opcode with prefix (if any)
   DUP FF00 AND  IF W, ELSE C, THEN  ;

30 VARIABLE MODE   \ 0=immed,10=direct,20=indexed,30=extended
: #   0 MODE ! ;
: <>   10 MODE ! ;

: INDEXREG   20 MODE !   \ rval postbyte -- postbyte |
   SWAP  1-  DUP 0 3 WITHIN 0= 3 ?ERROR  \ must be x,y,u, or s
   20 * OR ;                           \ put reg # in postbyte

: XMODE   <BUILDS (,)   \ postbyte --  | Simple Indexed Modes
   DOES>  @ INDEXREG ;  \ rval -- postbyte

84 XMODE 0,    86 XMODE A,    85 XMODE B,    8B XMODE D,
80 XMODE ,+    81 XMODE ,++   82 XMODE -,    83 XMODE --,

\ ------------------------------------------------------------
\   6809 assembler: addressing modes    (c) 04 06 85 BJR
\
: ,      SWAP 89 INDEXREG ;   \ rval n -- n postbyte |
: ,PCR   20 MODE !  8D ;      \ n -- n postbyte |

: []   MODE @ 20 =    \ Indexed:  postbyte -- postbyte
                      \ Extended:        n -- n postbyte
   IF  DUP 9D AND 80 = 3 ?ERROR  10 +    \ Indexed Indirect
   ELSE  20 MODE !  9F THEN ;            \ Extended Indirect

: RESET   30 MODE ! ;

\ register definitions
0 CONSTANT D    1 CONSTANT X    2 CONSTANT Y    3 CONSTANT U
4 CONSTANT S    5 CONSTANT PC   8 CONSTANT A    9 CONSTANT B
0A CONSTANT CCR    0B CONSTANT DPR

Y CONSTANT IP   U CONSTANT SP   S CONSTANT RP   X CONSTANT W

\ ------------------------------------------------------------
\   6809 assembler: inherent instruction(c) 03 06 85 BJR
\
: INHOP   <BUILDS (,)      \ opcode -- | Inherent Addressing
   DOES>   @ OPCODE, RESET ; \ -- | lay one or two bytes

3A INHOP ABX,   48 INHOP ASLA,  58 INHOP ASLB,  47 INHOP ASRA,
57 INHOP ASRB,  4F INHOP CLRA,  5F INHOP CLRB,  43 INHOP COMA,
53 INHOP COMB,  19 INHOP DAA,   4A INHOP DECA,  5A INHOP DECB,
4C INHOP INCA,  5C INHOP INCB,  48 INHOP LSLA,  58 INHOP LSLB,
44 INHOP LSRA,  54 INHOP LSRB,  3D INHOP MUL,   40 INHOP NEGA,
50 INHOP NEGB,  12 INHOP NOP,   49 INHOP ROLA,  59 INHOP ROLB,
```

```
46 INHOP RORA,  56 INHOP RORB,  3B INHOP RTI,   39 INHOP RTS,
1D INHOP SEX,  3F INHOP SWI, 103F INHOP SWI2, 113F INHOP SWI3,
13 INHOP SYNC,  4D INHOP TSTA,  5D INHOP TSTB,

\ ------------------------------------------------------------
\   6809 assembler: immediate instructio(c) 03 06 85 BJR
\
: IMMOP   <BUILDS (,)     \ opcode -- | Immediate Only (8-bit)
   DOES>   MODE @ 3 ?ERROR  @ C, C, RESET ;  \ operand --

3C IMMOP CWAI,  34 IMMOP PSHS,  36 IMMOP PSHU,  35 IMMOP PULS,
37 IMMOP PULU,  1C IMMOP ANDCC, 1A IMMOP ORCC,

: RROP   <BUILDS (,)       \ opcode -- | Register-Register
   DOES>   @ C,  SWAP 10 * + C, RESET ;  \ srcrval dstrval --

1E RROP EXG,    1F RROP TFR,

\ ------------------------------------------------------------
\   6809 assembler: +mode              (c) 03 06 85 BJR
\
: +MODE    \ operand -- operand | modify operand per mode
   MODE @ +  DUP 0F0 AND 50 = IF 0F AND THEN ; \ chng 5x to 0x

\ ------------------------------------------------------------
\   6809 assembler: pcrel, cofset       (c) 29 03 85 BJR
\
: PCREL   \ operand postbyte -- | lay PC relative
   SWAP  HERE 2+ -  DUP 8BIT?  \ try 8 bit relative offset
   IF  SWAP 0FE AND C, C,      \ it fits...lay postbyte,offset
   ELSE  1- SWAP C, W, THEN ;  \ no good...use 16 bit relative

: NOTINDIR?  10 AND 0= ;  \ postbyte -- f | test for indirect

: COFSET   \ operand postbyte -- | lay constant offset
   OVER 0= IF  0F0 AND 4 OR C, DROP     \ no offset
   ELSE OVER 5BIT? OVER NOTINDIR? AND IF
      60 AND SWAP 1F AND OR C,          \ 5 bit offset
   ELSE OVER 8BIT? IF  0FE AND C, C,    \ 8 bit offset
   ELSE  C, W,  THEN THEN THEN ;        \ 16 bit offset

\ ------------------------------------------------------------
\   6809 assembler: indexed, immed      (c) 03 06 85 BJR
\
: EXTIND    \ operand postbyte -- | lay extended indirect
   C, W, ;      \ lay postbyte and operand

: INDEXED   \ operand? postbyte -- | lay indexed poststuff
   DUP 8F AND CASE      \ check postbyte for modes w/ operands
      89 OF COFSET   ENDOF      \ const.offset
      8D OF PCREL    ENDOF      \ PC relative
      8F OF EXTIND   ENDOF      \ extended indir
            SWAP C,  ENDCASE ;  \ simple modes, postbyte only

: IMMED   \ operand opcode-pfa -- | lay immediate poststuff
   2+ @  DUP 0= 3 ?ERROR    \ test immedsize
   1- IF W, ELSE C, THEN ;  \ lay immed. operand in reqd.size

\ ------------------------------------------------------------
\   6809 assembler: general addr instr  (c) 03 06 85 BJR
\
: GENOP   <BUILDS (,) (,)   \ immedsize opcode -- | Gen'l Addr
   DOES>   DUP @ +MODE OPCODE,   \ [see below] | lay opcode
      MODE @ CASE 0 OF  IMMED         ENDOF   \ immediate
                  10 OF  DROP C,      ENDOF   \ direct
                  20 OF  DROP INDEXED ENDOF   \ indexed
                  30 OF  DROP W,      ENDOF   \ extended
      ENDCASE RESET ;

: INXOP   <BUILDS (,)       \ opcode -- | Indexed Only
   DOES>   MODE @ 20 - 3 ?ERROR  @ OPCODE, INDEXED RESET ;

\ Stack action of general addressing instructions
\ (1) immediate, direct, extended:                operand --
```

```
\ (2) all indexed except (3):                    postbyte --
\ (3) const.offset, PCR, extended indir:  operand postbyte --

\ ------------------------------------------------------------
\   6809 assembler: general addr instr  (c) 29 03 85 BJR
\
1   89 GENOP ADCA,   1   C9 GENOP ADCB,   1   8B GENOP ADDA,
1   CB GENOP ADDB,   2   C3 GENOP ADDD,   1   84 GENOP ANDA,
1   C4 GENOP ANDB,   1   85 GENOP BITA,   1   C5 GENOP BITB,
0   48 GENOP ASL,    0   47 GENOP ASR,    0   4F GENOP CLR,
1   81 GENOP CMPA,   1   C1 GENOP CMPB,   2 1083 GENOP CMPD,
2 118C GENOP CMPS,   2 1183 GENOP CMPU,   2   8C GENOP CMPX,
2 108C GENOP CMPY,   1   88 GENOP EORA,   1   C8 GENOP EORB,
0   43 GENOP COM,    0   4A GENOP DEC,    0   4C GENOP INC,
1   86 GENOP LDA,    1   C6 GENOP LDB,    2   CC GENOP LDD,
2 10CE GENOP LDS,    2   CE GENOP LDU,    2   8E GENOP LDX,
2 108E GENOP LDY,    0   4E GENOP JMP,    0   8D GENOP JSR,
0   48 GENOP LSL,    0   44 GENOP LSR,    0   40 GENOP NEG,
1   8A GENOP ORA,    1   CA GENOP ORB,    0   49 GENOP ROL,

\ ------------------------------------------------------------
\   6809 assembler: general addr instr  (c) 29 03 85 BJR
\
0   46 GENOP ROR,    1   82 GENOP SBCA,   1   C2 GENOP SBCB,
0   87 GENOP STA,    0   C7 GENOP STB,    0   CD GENOP STD,
0 10CF GENOP STS,    0   CF GENOP STU,    0   8F GENOP STX,
0 108F GENOP STY,    1   80 GENOP SUBA,   1   C0 GENOP SUBB,
2   83 GENOP SUBD,   0   4D GENOP TST,

32 INXOP LEAS,  33 INXOP LEAU,  30 INXOP LEAX,  31 INXOP LEAY,

\ ------------------------------------------------------------
\   6809 assembler: branches            (c) 03 06 85 BJR
\
: CONDBR   <BUILDS (,)      \ opcode -- | Conditional Branch
   DOES>   @  SWAP HERE 2+ -     \ addr --
   DUP 8BIT? IF  SWAP C, C,                  \ 8 bit
   ELSE  10 C, SWAP C, 2- W,  THEN RESET ;    \ 16 bit

: UNCBR   <BUILDS (,)       \ short:long -- | Uncondit'l Bran
   DOES>   @  SWAP HERE 2+ -     \ addr --
   DUP 8BIT? IF  SWAP >< C, C,        \ 8 bit: use short opcod
   ELSE  SWAP C, 1- W,  THEN RESET ;  \ 16 bit: use long opcod

\ ------------------------------------------------------------
\   6809 assembler: branch instructions (c) 29 03 85 BJR
\
24 CONDBR BCC,  25 CONDBR BCS,  27 CONDBR BEQ,  2C CONDBR BGE,
2E CONDBR BGT,  22 CONDBR BHI,  24 CONDBR BHS,  2F CONDBR BLE,
25 CONDBR BLO,  23 CONDBR BLS,  2D CONDBR BLT,  2B CONDBR BMI,
26 CONDBR BNE,  2A CONDBR BPL,  21 CONDBR BRN,  28 CONDBR BVC,
29 CONDBR BVS,  2016 UNCBR BRA,  8D17 UNCBR BSR,

\ ------------------------------------------------------------
\   6809 assembler: conditions          (c) 03 06 85 BJR
\
24 CONSTANT CS  25 CONSTANT CC  27 CONSTANT NE  2C CONSTANT LT
2E CONSTANT LE  22 CONSTANT LS  24 CONSTANT LO  2F CONSTANT GT
25 CONSTANT HS  23 CONSTANT HI  2D CONSTANT GE  2B CONSTANT PL
26 CONSTANT EQ  2A CONSTANT MI  21 CONSTANT ALW 28 CONSTANT VS
29 CONSTANT VC  20 CONSTANT NVR

\ ------------------------------------------------------------
\   6809 assembler: structured cond'ls  (c) 03 06 85 BJR
\
: IF,     \ br.opcode -- adr.next.instr 2  | reserve space
   C, 0 C, HERE 2 ;
: ENDIF,  \ adr.instr.after.br 2 -- | patch the forward ref.
   2 ?PAIRS   HERE OVER -  DUP 8BIT? 0= 3 ?ERROR  SWAP 1- C! ;
: ELSE,   \ adr.after.br 2 -- adr.after.this.br 2
   2 ?PAIRS   NVR C, 0 C, HERE SWAP  2 ENDIF, 2 ;
: BEGIN,  \ -- dest.adr 1
   HERE 1 ;
: UNTIL,  \ dest.adr 1 br.opcode --
```

```
    SWAP 1 ?PAIRS   C,  HERE 1+ -  DUP 8BIT? 0= 3 ?ERROR  C, ;
: WHILE,  \ dest.adr 1 br.opcod -- adr.after.this 2 dest.adr 1
   IF, 2SWAP ;
: REPEAT, \ adr.after.while 2 dest.adr.of.begin 1 --
   NVR UNTIL, ENDIF, ;
: THEN,  ENDIF, ;
: END,  UNTIL, ;

\ ------------------------------------------------------------
\   6809 assembler: code, ;code, ;c     (c) 14 11 85 BJR
\
FORTH DEFINITIONS ASSEMBLER
: ENTERCODE [COMPILE] ASSEMBLER   ASSEMBLER ALIGN  !CSP ;
: CODE CREATE ENTERCODE ;
: ;CODE  ?CSP COMPILE (;CODE) [COMPILE] [ ENTERCODE ;
   IMMEDIATE
ASSEMBLER DEFINITIONS

: ;C   CURRENT @ CONTEXT !   ?CSP   SMUDGE ;
: NEXT,   Y ,++ LDX,  X 0, [] JMP, ;
: NEXT    NEXT, ;
FORTH DEFINITIONS DECIMAL
□
```