Exhibit U
Part 4

Operation codes

G.SUB.H. 8.C	Group subtract halve signed bytes ceiling
G.SUB.H. 8.F	Group subtract halve signed bytes floor
G.SUB.H. 8.N	Group subtract halve signed bytes nearest
G.SUB.H. 8.Z	Group subtract halve signed bytes zero
G.SUB.H. 16.C	Group subtract halve signed doublets ceiling
G.SUB.H. 16.F	Group subtract halve signed doublets floor
G.SUB.H. 16.N	Group subtract halve signed doublets nearest
G.SUB.H. 16.Z	Group subtract halve signed doublets zero
G.SUB.H. 32.C	Group subtract halve signed quadlets ceiling
G.SUB.H. 32.F	Group subtract halve signed quadlets floor
G.SUB.H. 32.N	Group subtract halve signed quadlets nearest
G.SUB.H. 32.Z	Group subtract halve signed quadlets zero
G.SUB.H. 64.C	Group subtract halve signed octlets ceiling
G.SUB.H. 64.F	Group subtract halve signed octlets floor
G.SUB.H. 64.N	Group subtract halve signed octlets nearest
G.SUB.H. 64.Z	Group subtract halve signed octlets zero
G.SUB.H.128.C	Group subtract halve signed hexlet ceiling
G.SUB.H.128.F	Group subtract halve signed hexlet floor
G.SUB.H.128.N	Group subtract halve signed hexlet nearest
G.SUB.H.128.Z	Group subtract halve signed hexlet zero
G.SUB.H.U. 8.C	Group subtract halve unsigned bytes ceiling
G.SUB.H.U. 8.F	Group subtract halve unsigned bytes floor
G.SUB.H.U. 8.N	Group subtract halve unsigned bytes nearest
G.SUB.H.U. 8.Z	Group subtract halve unsigned bytes zero
G.SUB.H.U. 16.C	Group subtract halve unsigned doublets ceiling
G.SUB.H.U. 16.F	Group subtract halve unsigned doublets floor
G.SUB.H.U. 16.N	Group subtract halve unsigned doublets nearest
G.SUB.H.U. 16.Z	Group subtract halve unsigned doublets zero
G.SUB.H.U. 32.C	Group subtract halve unsigned quadlets ceiling
G.SUB.H.U. 32.F	Group subtract halve unsigned quadlets floor
G.SUB.H.U. 32.N	Group subtract halve unsigned quadlets nearest
G.SUB.H.U. 32.Z	Group subtract halve unsigned quadlets zero
G.SUB.H.U. 64.C	Group subtract halve unsigned octlets ceiling
G.SUB.H.U. 64.F	Group subtract halve unsigned octlets floor
G.SUB.H.U. 64.N	Group subtract halve unsigned octlets nearest
G.SUB.H.U. 64.Z	Group subtract halve unsigned octlets zero
G.SUB.H.U.128.C	Group subtract halve unsigned hexlet ceiling
G.SUB.H.U.128.F	Group subtract halve unsigned hexlet floor

FIG. 93A

G.SUB.H.U.128.N	Group subtract halve unsigned hexlet nearest
G.SUB.H.U.128.Z	Group subtract halve unsigned hexlet zero

Redundancies

G.SUB.H.size.rnd rd=rc,rc	⇔	G.ZERO rd
G.SUB.H.U.size.rnd rd=rc,rc	⇔	G.ZERO rd

FIG. 93A *continued*

Format

G.op.size.rnd rd=rb,rc

rd=gopsizernd(rb,rc)

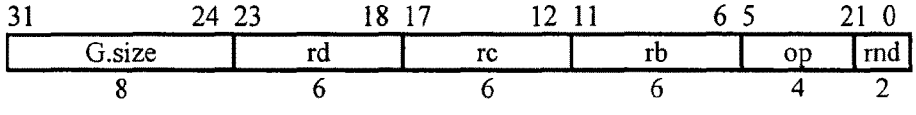


FIG. 93B

Definition

```
def GroupSubtractHalve(op,rnd,size,rd,rc,rb)
    c ← RegRead(rc, 128)
    b ← RegRead(rb, 128)
    case op of
        G.SUB.H.C, G.SUB.H.F, G.SUB.H.N, G.SUB.H.Z:
            as ← cs ← bs ← 1
        G.SUB.H.U.C, G.SUB.H.U.F, G.SUB.H.U.N, G.SUB.H.U.Z
            as ← 1
            cs ← bs ← 0
    endcase
    for i ← 0 to 128-size by size
        p ← ((bs and b_size-1) || b_size-1+i..i) - ((cs and c_size-1) || c_size-1+i..i)
        case rnd of
            none, N:
                s ← 0^size || ~p1
            Z:
                s ← 0^size || p_size
            F:
                s ← 0^size+1
            C:
                s ← 0^size || 1^1
        endcase
        v ← ((as & p_size)||p) + (0||s)
        if v_size+1 = (as & v_size) then
            a_size-1+i..i ← v_size..1
        else
            a_size-1+i..i ← as ? (v_size+1 || ~v^size-1_size+1) : 1^size
        endif
    endfor
    RegWrite(rd, 128, a)
enddef
```

Exceptions

none

FIG. 93C

Operation codes

E.CON. 8	Ensemble convolve signed bytes
E.CON.16	Ensemble convolve signed doublets
E.CON.32	Ensemble convolve signed quadlets
E.CON.64	Ensemble convolve signed octlets
E.CON.C. 8	Ensemble convolve complex bytes
E.CON.C.16	Ensemble convolve complex doublets
E.CON.C.32	Ensemble convolve complex quadlets
E.CON.M. 8	Ensemble convolve mixed-signed bytes
E.CON.M.16	Ensemble convolve mixed-signed doublets
E.CON.M.32	Ensemble convolve mixed-signed quadlets
E.CON.M.64	Ensemble convolve mixed-signed octlets
E.CON.U. 8	Ensemble convolve unsigned bytes
E.CON.U.16	Ensemble convolve unsigned doublets
E.CON.U.32	Ensemble convolve unsigned quadlets
E.CON.U.64	Ensemble convolve unsigned octlets
E.DIV.64	Ensemble divide signed octlets
E.DIV.U.64	Ensemble divide unsigned octlets
E.MUL. 8	Ensemble multiply signed bytes
E.MUL.16	Ensemble multiply signed doublets
E.MUL.32	Ensemble multiply signed quadlets
E.MUL.64	Ensemble multiply signed octlets
E.MUL.SUM. 8	Ensemble multiply sum signed bytes
E.MUL.SUM.16	Ensemble multiply sum signed doublets
E.MUL.SUM.32	Ensemble multiply sum signed quadlets
E.MUL.SUM.64	Ensemble multiply sum signed octlets
E.MUL.C. 8	Ensemble complex multiply bytes
E.MUL.C.16	Ensemble complex multiply doublets
E.MUL.C.32	Ensemble complex multiply quadlets
E.MUL.M. 8	Ensemble multiply mixed-signed bytes
E.MUL.M.16	Ensemble multiply mixed-signed doublets
E.MUL.M.32	Ensemble multiply mixed-signed quadlets
E.MUL.M.64	Ensemble multiply mixed-signed octlets
E.MUL.P. 8	Ensemble multiply polynomial bytes
E.MUL.P.16	Ensemble multiply polynomial doublets
E.MUL.P.32	Ensemble multiply polynomial quadlets
E.MUL.P.64	Ensemble multiply polynomial octlets
E.MUL.SUM.C. 8	Ensemble multiply sum complex bytes
E.MUL.SUM.C.16	Ensemble multiply sum complex doublets

FIG. 94A

E.MUL.SUM.C.32	Ensemble multiply sum complex quadlets
E.MUL.SUM.M. 8	Ensemble multiply sum mixed-signed bytes
E.MUL.SUM.M.16	Ensemble multiply sum mixed-signed doublets
E.MUL.SUM.M.32	Ensemble multiply sum mixed-signed quadlets
E.MUL.SUM.M.64	Ensemble multiply sum mixed-signed octlets
E.MUL.SUM.U. 8	Ensemble multiply sum unsigned bytes
E.MUL.SUM.U.16	Ensemble multiply sum unsigned doublets
E.MUL.SUM.U.32	Ensemble multiply sum unsigned quadlets
E.MUL.SUM.U.64	Ensemble multiply sum unsigned octlets
E.MUL.U. 8	Ensemble multiply unsigned bytes
E.MUL.U.16	Ensemble multiply unsigned doublets
E.MUL.U.32	Ensemble multiply unsigned quadlets
E.MUL.U.64	Ensemble multiply unsigned octlets

FIG. 94A *continued*

Selection

class	op	type	size
multiply	E.MUL	NONE M U P	8 16 32 64
		C	8 16 32
multiply sum	E.MUL.SUM	NONE M U	8 16 32 64
		C	8 16 32
divide	E.DIV	NONE U	64

Format

E.op.size rd=rc,rb

rd=eopsize(rc,rb)

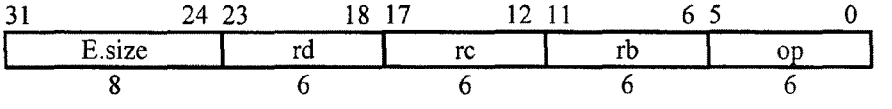


FIG. 94B

Definition

```
def mul(size,h,vs,v,i,ws,w,j) as
    mul ← ((vs&v_size-1+i)^h-size || v_size-1+i..i) * ((ws&w_size-1+j)^h-size || w_size-1+j..j)
enddef

def c ← PolyMultiply(size,a,b) as
    p[0] ← 0^2*size
    for k ← 0 to size-1
        p[k+1] ← p[k] ^ a_k ? (0^size-k || b || 0^k) : 0^2*size
    endfor
    c ← p[size]
enddef

def Ensemble(op,size,rd,rc,rb)
    c ← RegRead(rc, 128)
    b ← RegRead(rb, 128)
    case op of
        E.MUL:, E.MUL.C:, EMUL.SUM, E.MUL.SUM.C, E.CON, E.CON.C, E.DIV:
            cs ← bs ← 1
        E.MUL.M:, EMUL.SUM.M, E.CON.M:
            cs ← 0
            bs ← 1
        E.MUL.U:, EMUL.SUM.U, E.CON.U, E.DIV.U, E.MUL.P:
            cs ← bs ← 0
    endcase
    case op of
        E.MUL, E.MUL.U, E.MUL.M:
            for i ← 0 to 64-size by size
                d2*(i+size)-1..2*i ← mul(size,2*size,cs,c,i,bs,b,i)
            endfor
        E.MUL.P:
            for i ← 0 to 64-size by size
                d2*(i+size)-1..2*i ← PolyMultiply(size,c_size-1+i..i,b_size-1+i..i)
            endfor
        E.MUL.C:
            for i ← 0 to 64-size by size
                if (i and size) = 0 then
                    p ← mul(size,2*size,1,c,i,1,b,i) - mul(size,2*size,1,c,i+size,1,b,i+size)
                else
```

FIG. 94C

```
            p ← mul(size,2*size,1,c,i,1,b,i+size) + mul(size,2*size,1,c,i,1,b,i+size)
        endif
        d2*(i+size)-1..2*i ← p
    endfor
E.MUL.SUM, E.MUL.SUM.U, E.MUL.SUM.M:
    p[0] ← 0^128
    for i ← 0 to 128-size by size
        p[i+size] ← p[i] + mul(size,128,cs,c,i,bs,b,i)
    endfor
    a ← p[128]
E.MUL.SUM.C:
    p[0] ← 0^64
    p[size] ← 0^64
    for i ← 0 to 128-size by size
        if (i and size) = 0 then
            p[i+2*size] ← p[i] + mul(size,64,1,c,i,1,b,i)
                               - mul(size,64,1,c,i+size,1,b,i+size)
        else
            p[i+2*size] ← p[i] + mul(size,64,1,c,i,1,b,i+size)
                               + mul(size,64,1,c,i+size,1,b,i)
        endif
    endfor
    a ← p[128+size] || p[128]
E.CON, E.CON.U, E.CON.M:
    p[0] ← 0^128
    for j ← 0 to 64-size by size
        for i ← 0 to 64-size by size
            p[j+size]2*(i+size)-1..2*i ← p[j]2*(i+size)-1..2*i +
                mul(size,2*size,cs,c,i+64-j,bs,b,j)
        endfor
    endfor
    a ← p[64]
E.CON.C:
    p[0] ← 0^128
    for j ← 0 to 64-size by size
        for i ← 0 to 64-size by size
            if ((~i) and j and size) = 0 then
                p[j+size]2*(i+size)-1..2*i ← p[j]2*(i+size)-1..2*i +
                    mul(size,2*size,1,c,i+64-j,1,b,j)
            else
                p[j+size]2*(i+size)-1..2*i ← p[j]2*(i+size)-1..2*i -
                    mul(size,2*size,1,c,i+64-j+2*size,1,b,j)
```

FIG. 94C *continued*

```
                    endif
                endfor
            endfor
            a ← p[64]
        E.DIV:
            if (b = 0) or ( (c = (1||0^63)) and (b = 1^64) ) then
                a ← undefined
            else
                q ← c / b
                r ← c - q*b
                a ← r63..0 || q63..0
            endif
        E.DIV.U:
            if b = 0 then
                a ← undefined
            else
                q ← (0 || c) / (0 || b)
                r ← c - (0 || q)*(0 || b)
                a ← r63..0 || q63..0
            endif
    endcase
    RegWrite(rd, 128, a)
enddef
```

Exceptions

none

FIG. 94C *continued*

Operation codes

E.CON.X.I.C. 8.C.B	Ensemble convolve extract immediate signed complex bytes big-endian ceiling
E.CON.X.I.C. 8.F.B	Ensemble convolve extract immediate signed complex bytes big-endian floor
E.CON.X.I.C. 8.N.B	Ensemble convolve extract immediate signed complex bytes big-endian nearest
E.CON.X.I.C. 8.Z.B	Ensemble convolve extract immediate signed complex bytes big-endian zero
E.CON.X.I.C.16.C.B	Ensemble convolve extract immediate signed complex doublets big-endian ceiling
E.CON.X.I.C.16.F.B	Ensemble convolve extract immediate signed complex doublets big-endian floor
E.CON.X.I.C.16.N.B	Ensemble convolve extract immediate signed complex doublets big-endian nearest
E.CON.X.I.C.16.Z.B	Ensemble convolve extract immediate signed complex doublets big-endian zero
E.CON.X.I.C.32.C.B	Ensemble convolve extract immediate signed complex quadlets big-endian ceiling
E.CON.X.I.C.32.F.B	Ensemble convolve extract immediate signed complex quadlets big-endian floor
E.CON.X.I.C.32.N.B	Ensemble convolve extract immediate signed complex quadlets big-endian nearest
E.CON.X.I.C.32.Z.B	Ensemble convolve extract immediate signed complex quadlets big-endian zero
E.CON.X.I.C.64.C.B	Ensemble convolve extract immediate signed complex octlets big-endian ceiling
E.CON.X.I.C.64.F.B	Ensemble convolve extract immediate signed complex octlets big-endian floor
E.CON.X.I.C.64.N.B	Ensemble convolve extract immediate signed complex octlets big-endian nearest
E.CON.X.I.C.64.Z.B	Ensemble convolve extract immediate signed complex octlets big-endian zero
E.CON.X.I.C. 8.C.L	Ensemble convolve extract immediate signed complex bytes little-endian ceiling
E.CON.X.I.C. 8.F.L	Ensemble convolve extract immediate signed complex bytes little-endian floor
E.CON.X.I.C. 8.N.L	Ensemble convolve extract immediate signed complex bytes little-endian nearest
E.CON.X.I.C. 8.Z.L	Ensemble convolve extract immediate signed complex bytes little-endian zero

FIG. 95A

E.CON.X.I.C.16.C.L	Ensemble convolve extract immediate signed complex doublets little-endian ceiling
E.CON.X.I.C.16.F.L	Ensemble convolve extract immediate signed complex doublets little-endian floor
E.CON.X.I.C.16.N.L	Ensemble convolve extract immediate signed complex doublets little-endian nearest
E.CON.X.I.C.16.Z.L	Ensemble convolve extract immediate signed complex doublets little-endian zero
E.CON.X.I.C.32.C.L	Ensemble convolve extract immediate signed complex quadlets little-endian ceiling
E.CON.X.I.C.32.F.L	Ensemble convolve extract immediate signed complex quadlets little-endian floor
E.CON.X.I.C.32.N.L	Ensemble convolve extract immediate signed complex quadlets little-endian nearest
E.CON.X.I.C.32.Z.L	Ensemble convolve extract immediate signed complex quadlets little-endian zero
E.CON.X.I.C.64.C.L	Ensemble convolve extract immediate signed complex octlets little-endian ceiling
E.CON.X.I.C.64.F.L	Ensemble convolve extract immediate signed complex octlets little-endian floor
E.CON.X.I.C.64.N.L	Ensemble convolve extract immediate signed complex octlets little-endian nearest
E.CON.X.I.C.64.Z.L	Ensemble convolve extract immediate signed complex octlets little-endian zero
E.CON.X.I. 8.C.B	Ensemble convolve extract immediate signed bytes big-endian ceiling
E.CON.X.I. 8.F.B	Ensemble convolve extract immediate signed bytes big-endian floor
E.CON.X.I. 8.N.B	Ensemble convolve extract immediate signed bytes big-endian nearest
E.CON.X.I. 8.Z.B	Ensemble convolve extract immediate signed bytes big-endian zero
E.CON.X.I.16.C.B	Ensemble convolve extract immediate signed doublets big-endian ceiling
E.CON.X.I.16.F.B	Ensemble convolve extract immediate signed doublets big-endian floor
E.CON.X.I.16.N.B	Ensemble convolve extract immediate signed doublets big-endian nearest
E.CON.X.I.16.Z.B	Ensemble convolve extract immediate signed doublets big-endian zero
E.CON.X.I.32.C.B	Ensemble convolve extract immediate signed quadlets big-endian ceiling

FIG. 95A *continued*

E.CON.X.I.32.F.B	Ensemble convolve extract immediate signed quadlets big-endian floor
E.CON.X.I.32.N.B	Ensemble convolve extract immediate signed quadlets big-endian nearest
E.CON.X.I.32.Z.B	Ensemble convolve extract immediate signed quadlets big-endian zero
E.CON.X.I.64.C.B	Ensemble convolve extract immediate signed octlets big-endian ceiling
E.CON.X.I.64.F.B	Ensemble convolve extract immediate signed octlets big-endian floor
E.CON.X.I.64.N.B	Ensemble convolve extract immediate signed octlets big-endian nearest
E.CON.X.I.64.Z.B	Ensemble convolve extract immediate signed octlets big-endian zero
E.CON.X.I. 8.C.L	Ensemble convolve extract immediate signed bytes little-endian ceiling
E.CON.X.I. 8.F.L	Ensemble convolve extract immediate signed bytes little-endian floor
E.CON.X.I. 8.N.L	Ensemble convolve extract immediate signed bytes little-endian nearest
E.CON.X.I. 8.Z.L	Ensemble convolve extract immediate signed bytes little-endian zero
E.CON.X.I.16.C.L	Ensemble convolve extract immediate signed doublets little-endian ceiling
E.CON.X.I.16.F.L	Ensemble convolve extract immediate signed doublets little-endian floor
E.CON.X.I.16.N.L	Ensemble convolve extract immediate signed doublets little-endian nearest
E.CON.X.I.16.Z.L	Ensemble convolve extract immediate signed doublets little-endian zero
E.CON.X.I.32.C.L	Ensemble convolve extract immediate signed quadlets little-endian ceiling
E.CON.X.I.32.F.L	Ensemble convolve extract immediate signed quadlets little-endian floor
E.CON.X.I.32.N.L	Ensemble convolve extract immediate signed quadlets little-endian nearest
E.CON.X.I.32.Z.L	Ensemble convolve extract immediate signed quadlets little-endian zero
E.CON.X.I.64.C.L	Ensemble convolve extract immediate signed octlets little-endian ceiling
E.CON.X.I.64.F.L	Ensemble convolve extract immediate signed octlets little-endian floor

FIG. 95A *continued*

E.CON.X.I.64.N.L	Ensemble convolve extract immediate signed octlets little-endian nearest
E.CON.X.I.64.Z.L	Ensemble convolve extract immediate signed octlets little-endian zero
E.CON.X.I.M. 8.C.B	Ensemble convolve extract immediate mixed-signed bytes big-endian ceiling
E.CON.X.I.M. 8.F.B	Ensemble convolve extract immediate mixed-signed bytes big-endian floor
E.CON.X.I.M. 8.N.B	Ensemble convolve extract immediate mixed-signed bytes big-endian nearest
E.CON.X.I.M. 8.Z.B	Ensemble convolve extract immediate mixed-signed bytes big-endian zero
E.CON.X.I.M.16.C.B	Ensemble convolve extract immediate mixed-signed doublets big-endian ceiling
E.CON.X.I.M.16.F.B	Ensemble convolve extract immediate mixed-signed doublets big-endian floor
E.CON.X.I.M.16.N.B	Ensemble convolve extract immediate mixed-signed doublets big-endian nearest
E.CON.X.I.M.16.Z.B	Ensemble convolve extract immediate mixed-signed doublets big-endian zero
E.CON.X.I.M.32.C.B	Ensemble convolve extract immediate mixed-signed quadlets big-endian ceiling
E.CON.X.I.M.32.F.B	Ensemble convolve extract immediate mixed-signed quadlets big-endian floor
E.CON.X.I.M.32.N.B	Ensemble convolve extract immediate mixed-signed quadlets big-endian nearest
E.CON.X.I.M.32.Z.B	Ensemble convolve extract immediate mixed-signed quadlets big-endian zero
E.CON.X.I.M.64.C.B	Ensemble convolve extract immediate mixed-signed octlets big-endian ceiling
E.CON.X.I.M.64.F.B	Ensemble convolve extract immediate mixed-signed octlets big-endian floor
E.CON.X.I.M.64.N.B	Ensemble convolve extract immediate mixed-signed octlets big-endian nearest
E.CON.X.I.M.64.Z.B	Ensemble convolve extract immediate mixed-signed octlets big-endian zero
E.CON.X.I.M. 8.C.L	Ensemble convolve extract immediate mixed-signed bytes little-endian ceiling
E.CON.X.I.M. 8.F.L	Ensemble convolve extract immediate mixed-signed bytes little-endian floor
E.CON.X.I.M. 8.N.L	Ensemble convolve extract immediate mixed-signed bytes little-endian nearest

FIG. 95A *continued*

E.CON.X.I.M. 8.Z.L	Ensemble convolve extract immediate mixed-signed bytes little-endian zero
E.CON.X.I.M.16.C.L	Ensemble convolve extract immediate mixed-signed doublets little-endian ceiling
E.CON.X.I.M.16.F.L	Ensemble convolve extract immediate mixed-signed doublets little-endian floor
E.CON.X.I.M.16.N.L	Ensemble convolve extract immediate mixed-signed doublets little-endian nearest
E.CON.X.I.M.16.Z.L	Ensemble convolve extract immediate mixed-signed doublets little-endian zero
E.CON.X.I.M.32.C.L	Ensemble convolve extract immediate mixed-signed quadlets little-endian ceiling
E.CON.X.I.M.32.F.L	Ensemble convolve extract immediate mixed-signed quadlets little-endian floor
E.CON.X.I.M.32.N.L	Ensemble convolve extract immediate mixed-signed quadlets little-endian nearest
E.CON.X.I.M.32.Z.L	Ensemble convolve extract immediate mixed-signed quadlets little-endian zero
E.CON.X.I.M.64.C.L	Ensemble convolve extract immediate mixed-signed octlets little-endian ceiling
E.CON.X.I.M.64.F.L	Ensemble convolve extract immediate mixed-signed octlets little-endian floor
E.CON.X.I.M.64.N.L	Ensemble convolve extract immediate mixed-signed octlets little-endian nearest
E.CON.X.I.M.64.Z.L	Ensemble convolve extract immediate mixed-signed octlets little-endian zero
E.CON.X.I.U. 8.C.B	Ensemble convolve extract immediate unsigned bytes big-endian ceiling
E.CON.X.I.U. 8.F.B	Ensemble convolve extract immediate unsigned bytes big-endian floor
E.CON.X.I.U. 8.N.B	Ensemble convolve extract immediate unsigned bytes big-endian nearest
E.CON.X.I.U.16.C.B	Ensemble convolve extract immediate unsigned doublets big-endian ceiling
E.CON.X.I.U.16.F.B	Ensemble convolve extract immediate unsigned doublets big-endian floor
E.CON.X.I.U.16.N.B	Ensemble convolve extract immediate unsigned doublets big-endian nearest
E.CON.X.I.U.32.C.B	Ensemble convolve extract immediate unsigned quadlets big-endian ceiling
E.CON.X.I.U.32.F.B	Ensemble convolve extract immediate unsigned quadlets big-endian floor

FIG. 95A *continued*

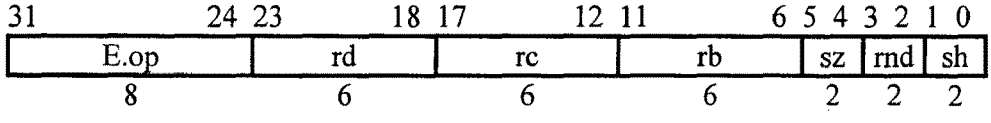
E.CON.X.I.U.32.N.B	Ensemble convolve extract immediate unsigned quadlets big-endian nearest
E.CON.X.I.U.64.C.B	Ensemble convolve extract immediate unsigned octlets big-endian ceiling
E.CON.X.I.U.64.F.B	Ensemble convolve extract immediate unsigned octlets big-endian floor
E.CON.X.I.U.64.N.B	Ensemble convolve extract immediate unsigned octlets big-endian nearest
E.CON.X.I.U. 8.C.L	Ensemble convolve extract immediate unsigned bytes little-endian ceiling
E.CON.X.I.U. 8.F.L	Ensemble convolve extract immediate unsigned bytes little-endian floor
E.CON.X.I.U. 8.N.L	Ensemble convolve extract immediate unsigned bytes little-endian nearest
E.CON.X.I.U.16.C.L	Ensemble convolve extract immediate unsigned doublets little-endian ceiling
E.CON.X.I.U.16.F.L	Ensemble convolve extract immediate unsigned doublets little-endian floor
E.CON.X.I.U.16.N.L	Ensemble convolve extract immediate unsigned doublets little-endian nearest
E.CON.X.I.U.32.C.L	Ensemble convolve extract immediate unsigned quadlets little-endian ceiling
E.CON.X.I.U.32.F.L	Ensemble convolve extract immediate unsigned quadlets little-endian floor
E.CON.X.I.U.32.N.L	Ensemble convolve extract immediate unsigned quadlets little-endian nearest
E.CON.X.I.U.64.C.L	Ensemble convolve extract immediate unsigned octlets little-endian ceiling
E.CON.X.I.U.64.F.L	Ensemble convolve extract immediate unsigned octlets little-endian floor
E.CON.X.I.U.64.N.L	Ensemble convolve extract immediate unsigned octlets little-endian nearest

FIG. 95A *continued*

Format

E.op.size.rnd rd@rc,rb,i

rd=eopsizernd(rd,rc,rb,i)



sz ← log(size) - 3
sh ← size + 7 - log(size) - i

FIG. 95B

Definition

```
def mul(size,h,vs,v,i,ws,w,j) as
    mul ← ((vs&v_{size-1+i})^{h-size} || v_{size-1+i..i}) * ((ws&w_{size-1+j})^{h-size} || w_{size-1+j..j})
enddef

def EnsembleConvolveExtractImmediate(op,rnd,gsize,rd,rc,rb,sh)
    d ← RegRead(rd, 128)
    c ← RegRead(rd, 128)
    b ← RegRead(rb, 128)
    lgsize ← log(gsize)
    wsize ← 128
    msize ← 256
    vsize ← 128
    case op of
        E.CON.X.I.B, E.CON.X.I.U.B, E.CON.X.I.M.B, E.CON.X.I.C.B:
            m ← d || c
        E.CON.X.I.L, E.CON.X.I.U.L, E.CON.X.I.M.L, E.CON.X.I.C.L:
            m ← c || d
    endcase
    case op of
        E.CON.X.I.U.B, E.CON.X.I.U.L:
            as ← ms ← bs ← false
        E.CON.X.I.M.B, E.CON.X.I.M.L:
            ms ← false
            as ← bs ← true
        E.CON.X.I.B, E.CON.X.I.L, E.CON.X.I.C.B, E.CON.X.I.C.L:
            as ← ms ← bs ← true
    endcase
    h ← (2*gsize) + 7 - lgsize
    r ← h - size - sh
    for i ← 0 to wsize-gsize by gsize
        q[0] ← 0^{2*gsize+7-lgsize}
        for j ← 0 to vsize-gsize by gsize
            case op of
                E.CON.X.I.B, E.CON.X.I.L, E.CON.X.I.M.B, E.CON.X.I.M.L,
                E.CON.X.I.U.B, E.CON.X.I.U.L:
                    q[j+gsize] ← q[j] + mul(gsize,h,ms,m,i+128-j,bs,b,j)
                E.CON.X.I.C.B, E.CON.X.I.C.L:
                    if (~i) & j & gsize = 0 then
                        q[j+gsize] ← q[j] + mul(gsize,h,ms,m,i+128-j,bs,b,j)
                    else
```

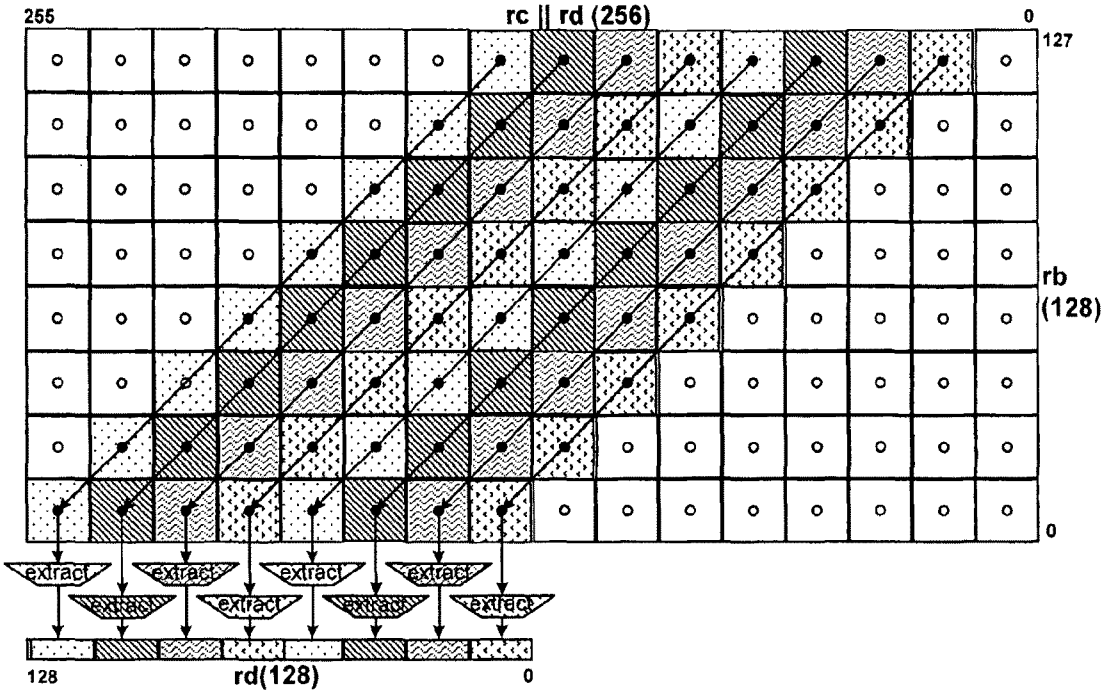
FIG. 95C

```
                        q[j+gsize] ← q[j] - mul(gsize,h,ms,m,i+128-j+2*gsize,bs,b,j)
                    endif
                endcase
            endfor
            p ← q[vsize]
            case rnd of
                none, N:
                    s ← 0^(h-r) || ~p_r || p_(r-1)^(r-1)
                Z:
                    s ← 0^(h-r) || p_(h-1)^r
                F:
                    s ← 0^h
                C:
                    s ← 0^(h-r) || 1^r
            endcase
            v ← ((as & p_(h-1))||p) + (0||s)
            if (v_(h..r+gsize) = (as & v_(r+gsize-1))^(h+1-r-gsize) then
                a_(gsize-1+i..i) ← v_(gsize-1+r..r)
            else
                a_(gsize-1+i..i) ← as ? (v_h || ~v_h^(gsize-1)) : 1^gsize
            endif
        endfor
        a_(127..wsize) ← 0
        RegWrite(rd, 128, a)
enddef
```

Exceptions

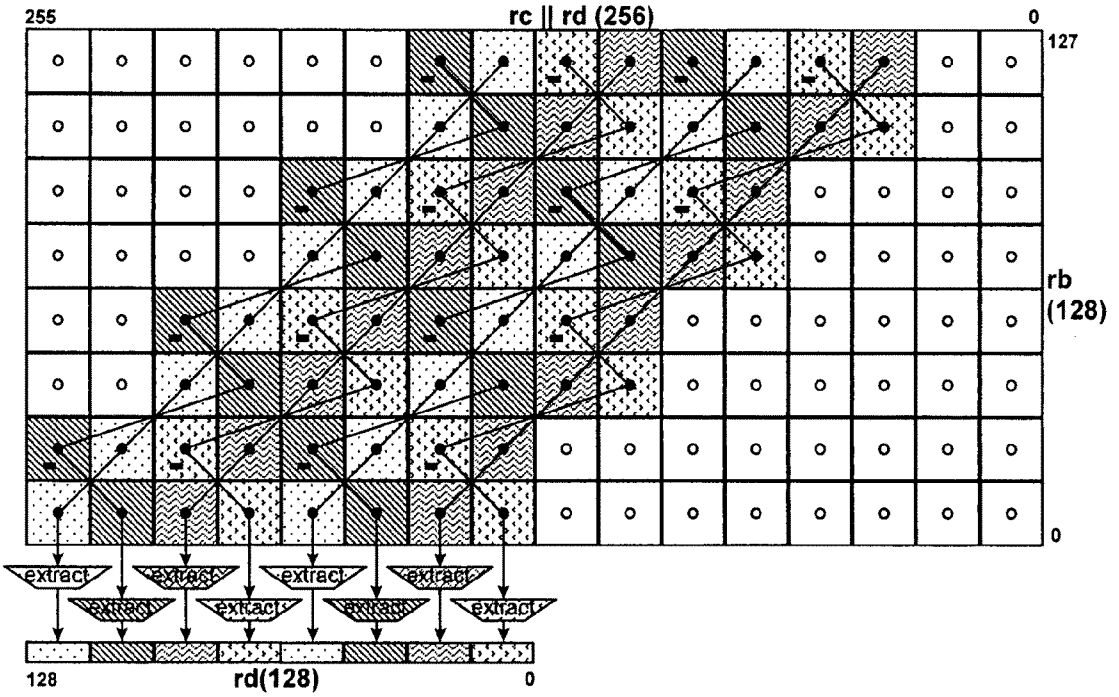
none

FIG. 95C *continued*



Ensemble convolve extract immediate doublets

FIG. 95D



Ensemble convolve extract immediate complex doublets

FIG. 95E

Operation codes

E.CON.F.16.B	Ensemble convolve floating-point half big-endian
E.CON.F.16.L	Ensemble convolve floating-point half little-endian
E.CON.F.32.B	Ensemble convolve floating-point single big-endian
E.CON.F.32.L	Ensemble convolve floating-point single little-endian
E.CON.F.64.B	Ensemble convolve floating-point double big-endian
E.CON.F.64.L	Ensemble convolve floating-point double little-endian
E.CON.C.F.16.B	Ensemble convolve complex floating-point half big-endian
E.CON.C.F.16.L	Ensemble convolve complex floating-point half little-endian
E.CON.C.F.32.B	Ensemble convolve complex floating-point single big-endian
E.CON.C.F.32.L	Ensemble convolve complex floating-point single little-endian
E.CON.C.F.64.B	Ensemble convolve complex floating-point double big-endian
E.CON.C.F.64.L	Ensemble convolve complex floating-point double little-endian

FIG. 96A

Format

E.op.size.order rd=rc,rb

rd=eopsizeorder(rd,rc,rb)

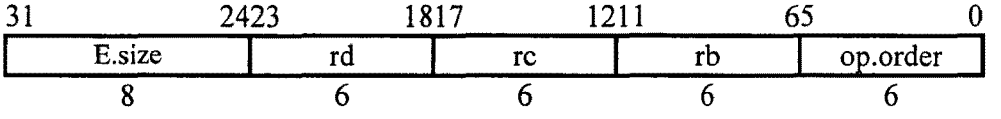


FIG. 96B

Definition

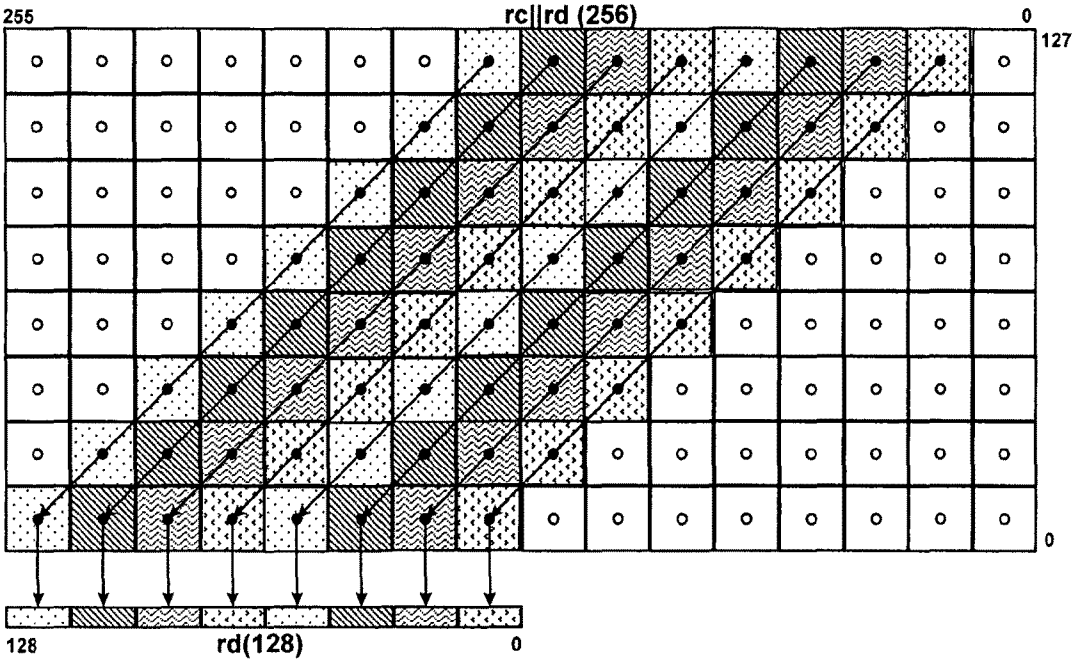
```
def mul(size,v,i,w,j) as
    mul ← fmul(F(size,v_size-1+i..i),F(size,w_size-1+j..j))
enddef

def EnsembleConvolveFloatingPoint(op,gsize,rd,rc,rb)
    d ← RegRead(rd, 128)
    c ← RegRead(rc, 128)
    b ← RegRead(rb, 128)
    lgsize ← log(gsize)
    wsize ← 128
    msize ← 256
    vsize ← 128
    case op of
        E.CON.F.B, E.CON.C.F.B:
            m ← d || c
        E.CON.F.L, E.CON.C.F.L :
            m ← c || d
    endcase
    for i ← 0 to wsize-gsize by gsize
        //NULL value doesn't combine with zero to alter sign bit
        q[0].t ← NULL
        for j ← 0 to vsize-gsize by gsize
            case op of
                E.CONF.L, E.CONF.B:
                    q[j+gsize] ← fadd(q[j], mul(gsize,m,i+128-j,b,j))
                E.CONCF.L, E.CONCF.B:
                    if (~i) & j & gsize = 0 then
                        q[j+gsize] ← fadd(q[j], mul(gsize,m,i+128-j,b,j))
                    else
                        q[j+gsize] ← fsub(q[j], mul(gsize,m,i+128-j+2*gsize,b,j))
                    endif
            endcase
        endfor
        a_gsize-1+i..i ← PackF(gsize,q[vsize],N)
    endfor
    a_127..wsize ← 0
    RegWrite(rd, 128, a)
enddef
```

Exceptions

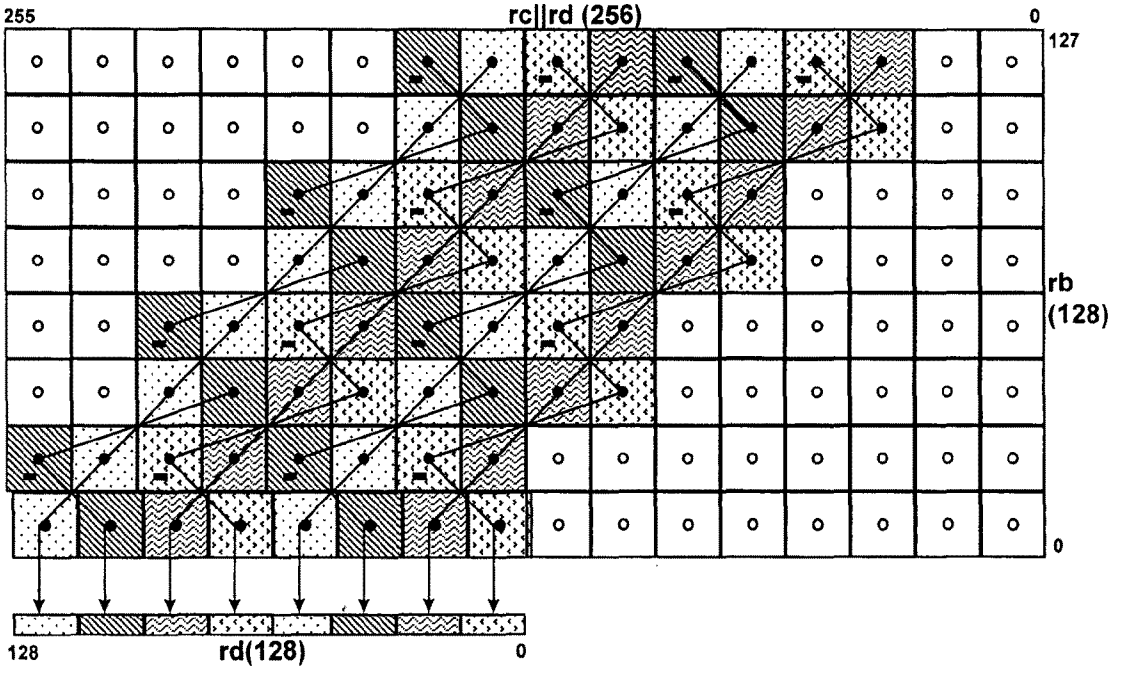
none

FIG. 96C



Ensemble convolve floating-point half little-endian

FIG. 96D



Ensemble convolve complex floating-point half little-endian

FIG. 96E

Operation codes

E.EXTRACT.I. 8.C	Ensemble extract immediate signed bytes ceiling
E.EXTRACT.I. 8.F	Ensemble extract immediate signed bytes floor
E.EXTRACT.I. 8.N	Ensemble extract immediate signed bytes nearest
E.EXTRACT.I. 8.Z	Ensemble extract immediate signed bytes zero
E.EXTRACT.I.16.C	Ensemble extract immediate signed doublets ceiling
E.EXTRACT.I.16.F	Ensemble extract immediate signed doublets floor
E.EXTRACT.I.16.N	Ensemble extract immediate signed doublets nearest
E.EXTRACT.I.16.Z	Ensemble extract immediate signed doublets zero
E.EXTRACT.I.32.C	Ensemble extract immediate signed quadlets ceiling
E.EXTRACT.I.32.F	Ensemble extract immediate signed quadlets floor
E.EXTRACT.I.32.N	Ensemble extract immediate signed quadlets nearest
E.EXTRACT.I.32.Z	Ensemble extract immediate signed quadlets zero
E.EXTRACT.I.64.C	Ensemble extract immediate signed octlets ceiling
E.EXTRACT.I.64.F	Ensemble extract immediate signed octlets floor
E.EXTRACT.I.64.N	Ensemble extract immediate signed octlets nearest
E.EXTRACT.I.64.Z	Ensemble extract immediate signed octlets zero
E.EXTRACT.I.U. 8.C	Ensemble extract immediate unsigned bytes ceiling
E.EXTRACT.I.U. 8.F	Ensemble extract immediate unsigned bytes floor
E.EXTRACT.I.U. 8.N	Ensemble extract immediate unsigned bytes nearest
E.EXTRACT.I.U.16.C	Ensemble extract immediate unsigned doublets ceiling
E.EXTRACT.I.U.16.F	Ensemble extract immediate unsigned doublets floor
E.EXTRACT.I.U.16.N	Ensemble extract immediate unsigned doublets nearest
E.EXTRACT.I.U.32.C	Ensemble extract immediate unsigned quadlets ceiling
E.EXTRACT.I.U.32.F	Ensemble extract immediate unsigned quadlets floor
E.EXTRACT.I.U.32.N	Ensemble extract immediate unsigned quadlets nearest
E.EXTRACT.I.U.64.C	Ensemble extract immediate unsigned octlets ceiling
E.EXTRACT.I.U.64.F	Ensemble extract immediate unsigned octlets floor
E.EXTRACT.I.U.64.N	Ensemble extract immediate unsigned octlets nearest
E.MUL.X.I. 8.C	Ensemble multiply extract immediate signed bytes ceiling
E.MUL.X.I. 8.F	Ensemble multiply extract immediate signed bytes floor
E.MUL.X.I. 8.N	Ensemble multiply extract immediate signed bytes nearest
E.MUL.X.I. 8.Z	Ensemble multiply extract immediate signed bytes zero
E.MUL.X.I.16.C	Ensemble multiply extract immediate signed doublets ceiling
E.MUL.X.I.16.F	Ensemble multiply extract immediate signed doublets floor
E.MUL.X.I.16.N	Ensemble multiply extract immediate signed doublets nearest
E.MUL.X.I.16.Z	Ensemble multiply extract immediate signed doublets zero

FIG. 97A

E.MUL.X.I.32.C	Ensemble multiply extract immediate signed quadlets ceiling
E.MUL.X.I.32.F	Ensemble multiply extract immediate signed quadlets floor
E.MUL.X.I.32.N	Ensemble multiply extract immediate signed quadlets nearest
E.MUL.X.I.32.Z	Ensemble multiply extract immediate signed quadlets zero
E.MUL.X.I.64.C	Ensemble multiply extract immediate signed octlets ceiling
E.MUL.X.I.64.F	Ensemble multiply extract immediate signed octlets floor
E.MUL.X.I.64.N	Ensemble multiply extract immediate signed octlets nearest
E.MUL.X.I.64.Z	Ensemble multiply extract immediate signed octlets zero
E.MUL.X.I.C. 8.C	Ensemble multiply extract immediate complex bytes ceiling
E.MUL.X.I.C. 8.F	Ensemble multiply extract immediate complex bytes floor
E.MUL.X.I.C. 8.N	Ensemble multiply extract immediate complex bytes nearest
E.MUL.X.I.C. 8.Z	Ensemble multiply extract immediate complex bytes zero
E.MUL.X.I.C.16.C	Ensemble multiply extract immediate complex doublets ceiling
E.MUL.X.I.C.16.F	Ensemble multiply extract immediate complex doublets floor
E.MUL.X.I.C.16.N	Ensemble multiply extract immediate complex doublets nearest
E.MUL.X.I.C.16.Z	Ensemble multiply extract immediate complex doublets zero
E.MUL.X.I.C.32.C	Ensemble multiply extract immediate complex quadlets ceiling
E.MUL.X.I.C.32.F	Ensemble multiply extract immediate complex quadlets floor
E.MUL.X.I.C.32.N	Ensemble multiply extract immediate complex quadlets nearest
E.MUL.X.I.C.32.Z	Ensemble multiply extract immediate complex quadlets zero
E.MUL.X.I.C.64.C	Ensemble multiply extract immediate complex octlets ceiling
E.MUL.X.I.C.64.F	Ensemble multiply extract immediate complex octlets floor
E.MUL.X.I.C.64.N	Ensemble multiply extract immediate complex octlets nearest
E.MUL.X.I.C.64.Z	Ensemble multiply extract immediate complex octlets zero
E.MUL.X.I.M. 8.C	Ensemble multiply extract immediate mixed-signed bytes ceiling
E.MUL.X.I.M. 8.F	Ensemble multiply extract immediate mixed-signed bytes floor
E.MUL.X.I.M. 8.N	Ensemble multiply extract immediate mixed-signed bytes nearest
E.MUL.X.I.M. 8.Z	Ensemble multiply extract immediate mixed-signed bytes zero
E.MUL.X.I.M.16.C	Ensemble multiply extract immediate mixed-signed doublets ceiling

FIG. 97A *continued*

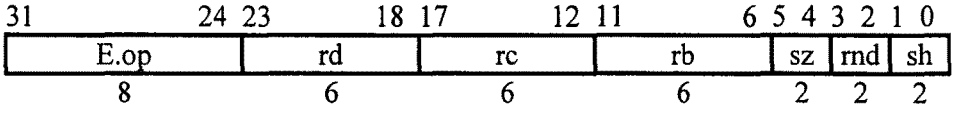
E.MUL.X.I.M.16.F	Ensemble multiply extract immediate mixed-signed doublets floor
E.MUL.X.I.M.16.N	Ensemble multiply extract immediate mixed-signed doublets nearest
E.MUL.X.I.M.16.Z	Ensemble multiply extract immediate mixed-signed doublets zero
E.MUL.X.I.M.32.C	Ensemble multiply extract immediate mixed-signed quadlets ceiling
E.MUL.X.I.M.32.F	Ensemble multiply extract immediate mixed-signed quadlets floor
E.MUL.X.I.M.32.N	Ensemble multiply extract immediate mixed-signed quadlets nearest
E.MUL.X.I.M.32.Z	Ensemble multiply extract immediate mixed-signed quadlets zero
E.MUL.X.I.M.64.C	Ensemble multiply extract immediate mixed-signed octlets ceiling
E.MUL.X.I.M.64.F	Ensemble multiply extract immediate mixed-signed octlets floor
E.MUL.X.I.M.64.N	Ensemble multiply extract immediate mixed-signed octlets nearest
E.MUL.X.I.M.64.Z	Ensemble multiply extract immediate mixed-signed octlets zero
E.MUL.X.I.U. 8.C	Ensemble multiply extract immediate unsigned bytes ceiling
E.MUL.X.I.U. 8.F	Ensemble multiply extract immediate unsigned bytes floor
E.MUL.X.I.U. 8.N	Ensemble multiply extract immediate unsigned bytes nearest
E.MUL.X.I.U.16.C	Ensemble multiply extract immediate unsigned doublets ceiling
E.MUL.X.I.U.16.F	Ensemble multiply extract immediate unsigned doublets floor
E.MUL.X.I.U.16.N	Ensemble multiply extract immediate unsigned doublets nearest
E.MUL.X.I.U.32.C	Ensemble multiply extract immediate unsigned quadlets ceiling
E.MUL.X.I.U.32.F	Ensemble multiply extract immediate unsigned quadlets floor
E.MUL.X.I.U.32.N	Ensemble multiply extract immediate unsigned quadlets nearest
E.MUL.X.I.U.64.C	Ensemble multiply extract immediate unsigned octlets ceiling
E.MUL.X.I.U.64.F	Ensemble multiply extract immediate unsigned octlets floor
E.MUL.X.I.U.64.N	Ensemble multiply extract immediate unsigned octlets nearest

FIG. 97A *continued*

Format

E.op.size.rnd rd=rc,rb,i

rd=eopsizernd(rc,rb,i)




```
sz ← log(size) - 3
case op of
        E.EXTRACT.I, E.EXTRACT.I.U, E.MUL.X.I, E.MUL.X.I.U, E.MUL.X.I.M:
                assert size ≥ i ≥ size-3
                sh ← size - i
        E.MUL.X.I.C:
                assert size+1 ≥ i ≥ size-2
                sh ← size + 1 - i
endcase
```

FIG. 97B

Definition

```
def mul(size,h,vs,v,i,ws,w,j) as
    mul ← ((vs&v_{size-1+i}..i)^{h-size} || v_{size-1+i..i}) * ((ws&w_{size-1+j})^{h-size} || w_{size-1+j..j})
enddef

def EnsembleExtractImmediate(op,rnd,size,ra,rb,rc,sh)
    c ← RegRead(rc, 128)
    b ← RegRead(rb, 128)
    case op of
        E.EXTRACT.I, E.MUL.X.I, E.MUL.X.I.C:
            as ← 1
            cs ← 1
            bs ← 1
        E.MUL.X.I.M:
            as ← 1
            cs ← 0
            bs ← 1
        E.EXTRACT.I.U, E.MUL.X.I.U:
            as ← 1
            cs ← 0
            bs ← 0
            if rnd = Z then
                raise ReservedInstruction
            endif
    endcase
    case op of
        E.EXTRACT.I, E.EXTRACT.I.U, E.MUL.X.I, E.MUL.X.I.U, E.MUL.X.I.M:
            h ← 2*size
        E.MUL.X.I.C:
            h ← (2*size) + 1
    endcase
    r ← h - size - sh
    for i ← 0 to 128-size by size
        case op of
            E.EXTRACT.I, E.EXTRACT.I.U:
                p ← (c || b)_{2*(size+i)-1..2*I}
            E.MUL.X.I, E.MUL.X.I.M, E.MUL.X.I.U:
                p ← mul(size,h,cs,c,i,bs,b,i)
            E.MUL.X.I.C:
                if i & size = 0 then
                    p ← mul(size,h,cs,c,i,bs,b,i) - mul(size,h,cs,c,i+size,bs,b,i+size)
                else
```

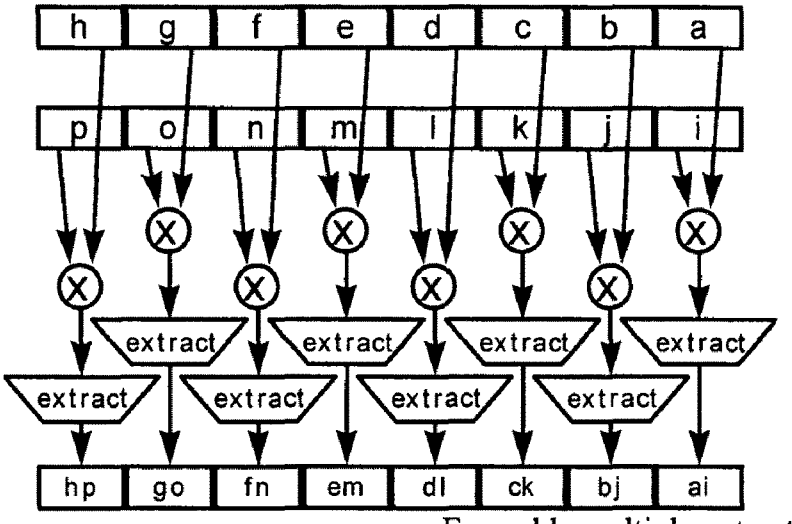
FIG. 97C

```
                    p ← mul(size,h,cs,c,i,bs,b,i+size) + mul(size,h,cs,c,i,bs,b,i+size)
                endif
            endcase
            case rnd of
                none, N:
                    s ← 0^(h-r) || ~p_r || p_r^(r-1)
                Z:
                    s ← 0^(h-r) || p_(h-1)^r
                F:
                    s ← 0^h
                C:
                    s ← 0^(h-r) || 1^r
            endcase
            v ← ((as & p_(h-1))||p) + (0||s)
            if (v_(h..r+size) = (as & v_(r+size-1))^(h+1-r-size) then
                a_(size-1+i..i) ← v_(size-1+r..r)
            else
                a_(size-1+i..i) ← as ? (v_h || ~v_h^(size-1)) : 1^size
            endif
        endfor
        RegWrite(rd, 128, a)
enddef
```

Exceptions

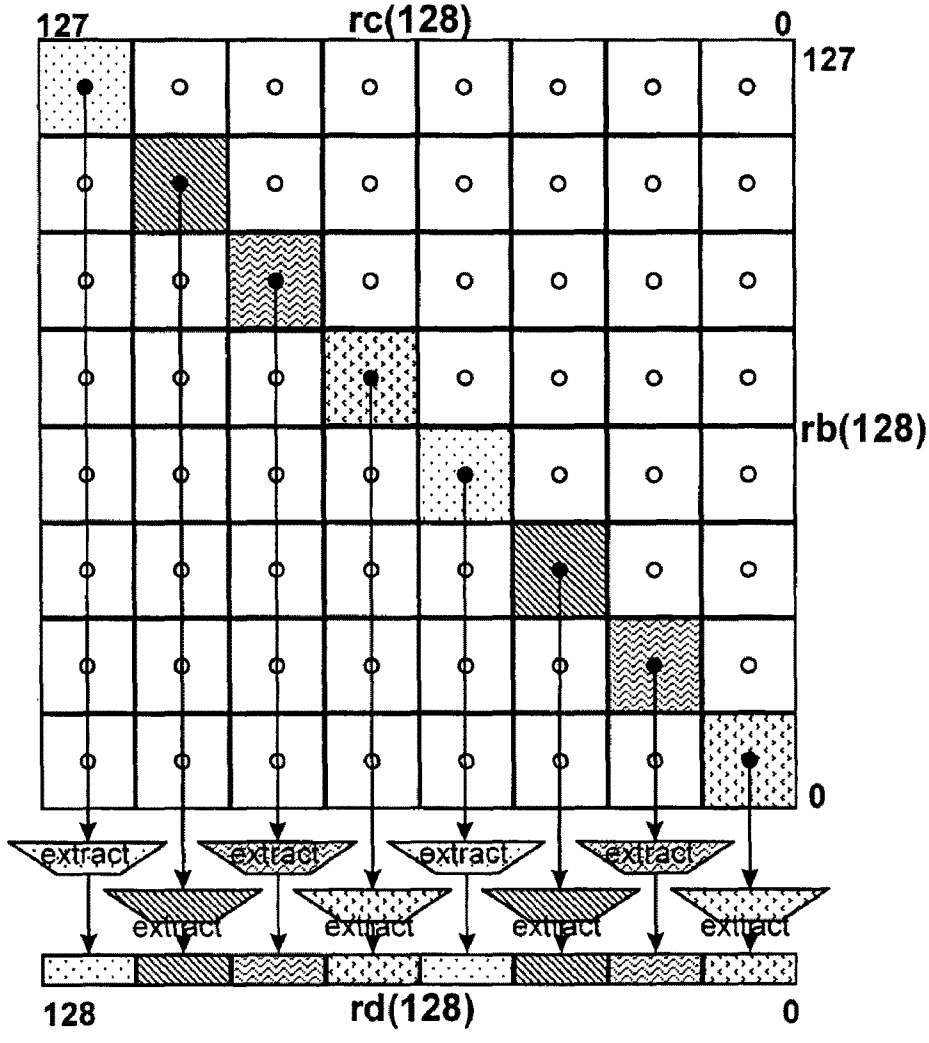
ReservedInstruction

FIG. 97C *continued*



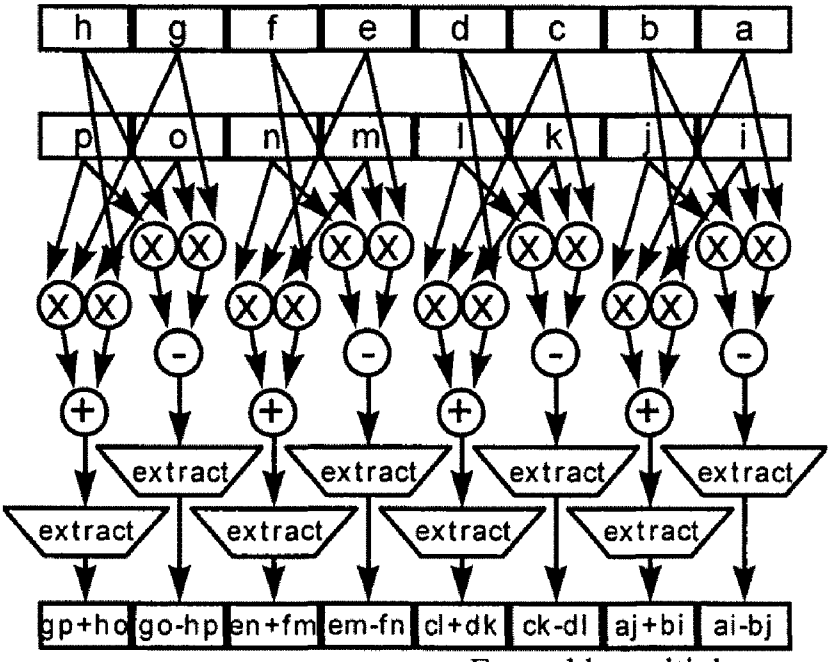
Ensemble multiply extract immediate doublets

FIG. 97D



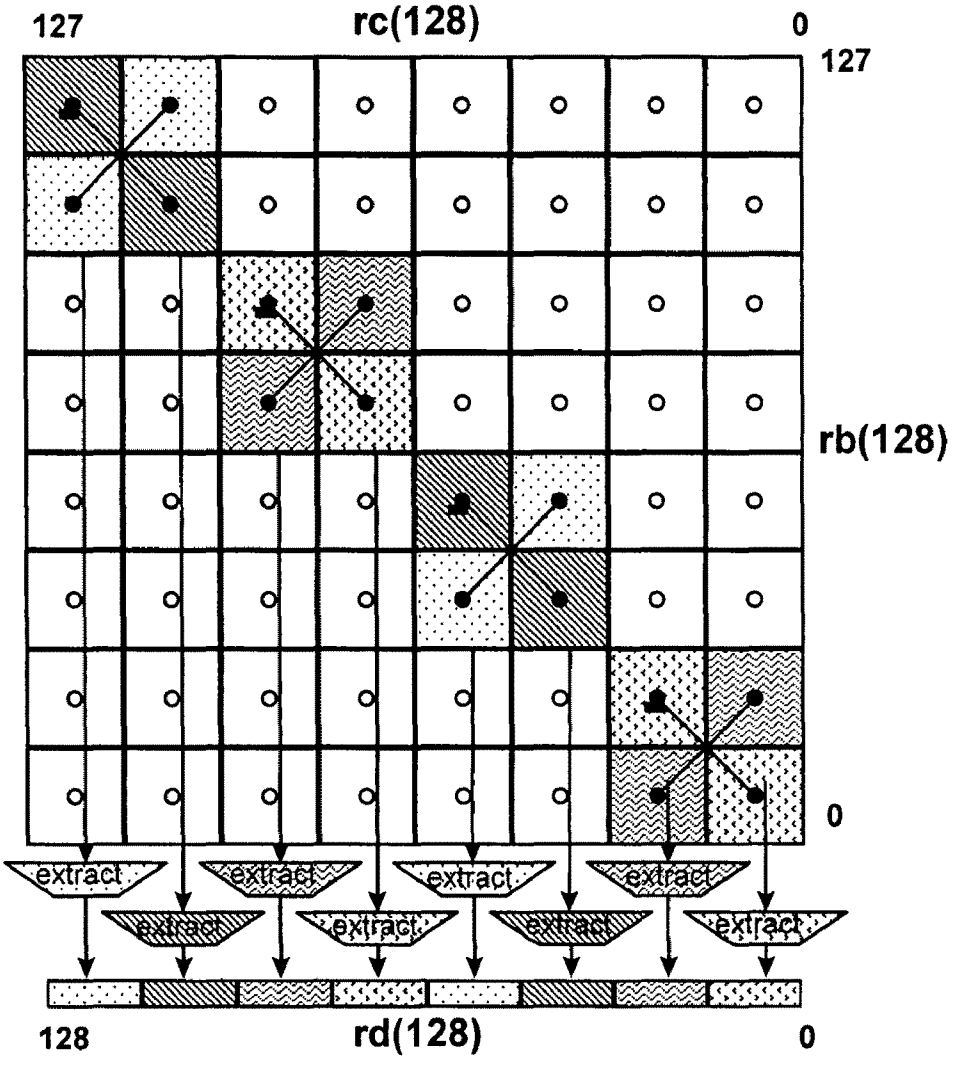
Ensemble multiply extract immediate doublets

FIG. 97E



Ensemble multiply extract immediate complex doublets

FIG. 97F



Ensemble multiply extract immediate complex doublets

FIG. 97G

Operation codes

E.MUL.ADD.X.I.C. 8.C	Ensemble multiply add extract immediate signed complex bytes ceiling
E.MUL.ADD.X.I.C. 8.F	Ensemble multiply add extract immediate signed complex bytes floor
E.MUL.ADD.X.I.C. 8.N	Ensemble multiply add extract immediate signed complex bytes nearest
E.MUL.ADD.X.I.C. 8.Z	Ensemble multiply add extract immediate signed complex bytes zero
E.MUL.ADD.X.I.C.16.C	Ensemble multiply add extract immediate signed complex doublets ceiling
E.MUL.ADD.X.I.C.16.F	Ensemble multiply add extract immediate signed complex doublets floor
E.MUL.ADD.X.I.C.16.N	Ensemble multiply add extract immediate signed complex doublets nearest
E.MUL.ADD.X.I.C.16.Z	Ensemble multiply add extract immediate signed complex doublets zero
E.MUL.ADD.X.I.C.32.C	Ensemble multiply add extract immediate signed complex quadlets ceiling
E.MUL.ADD.X.I.C.32.F	Ensemble multiply add extract immediate signed complex quadlets floor
E.MUL.ADD.X.I.C.32.N	Ensemble multiply add extract immediate signed complex quadlets nearest
E.MUL.ADD.X.I.C.32.Z	Ensemble multiply add extract immediate signed complex quadlets zero
E.MUL.ADD.X.I.C.64.C	Ensemble multiply add extract immediate signed complex octlets ceiling
E.MUL.ADD.X.I.C.64.F	Ensemble multiply add extract immediate signed complex octlets floor
E.MUL.ADD.X.I.C.64.N	Ensemble multiply add extract immediate signed complex octlets nearest
E.MUL.ADD.X.I.C.64.Z	Ensemble multiply add extract immediate signed complex octlets zero
E.MUL.ADD.X.I.M. 8.C	Ensemble multiply add extract immediate mixed-signed bytes ceiling
E.MUL.ADD.X.I.M. 8.F	Ensemble multiply add extract immediate mixed-signed bytes floor
E.MUL.ADD.X.I.M. 8.N	Ensemble multiply add extract immediate mixed-signed bytes nearest
E.MUL.ADD.X.I.M. 8.Z	Ensemble multiply add extract immediate mixed-signed bytes zero

FIG. 98A

E.MUL.ADD.X.I.M.16.C	Ensemble multiply add extract immediate mixed-signed doublets ceiling
E.MUL.ADD.X.I.M.16.F	Ensemble multiply add extract immediate mixed-signed doublets floor
E.MUL.ADD.X.I.M.16.N	Ensemble multiply add extract immediate mixed-signed doublets nearest
E.MUL.ADD.X.I.M.16.Z	Ensemble multiply add extract immediate mixed-signed doublets zero
E.MUL.ADD.X.I.M.32.C	Ensemble multiply add extract immediate mixed-signed quadlets ceiling
E.MUL.ADD.X.I.M.32.F	Ensemble multiply add extract immediate mixed-signed quadlets floor
E.MUL.ADD.X.I.M.32.N	Ensemble multiply add extract immediate mixed-signed quadlets nearest
E.MUL.ADD.X.I.M.32.Z	Ensemble multiply add extract immediate mixed-signed quadlets zero
E.MUL.ADD.X.I.M.64.C	Ensemble multiply add extract immediate mixed-signed octlets ceiling
E.MUL.ADD.X.I.M.64.F	Ensemble multiply add extract immediate mixed-signed octlets floor
E.MUL.ADD.X.I.M.64.N	Ensemble multiply add extract immediate mixed-signed octlets nearest
E.MUL.ADD.X.I.M.64.Z	Ensemble multiply add extract immediate mixed-signed octlets zero
E.MUL.ADD.X.I. 8.C	Ensemble multiply add extract immediate signed bytes ceiling
E.MUL.ADD.X.I. 8.F	Ensemble multiply add extract immediate signed bytes floor
E.MUL.ADD.X.I. 8.N	Ensemble multiply add extract immediate signed bytes nearest
E.MUL.ADD.X.I. 8.Z	Ensemble multiply add extract immediate signed bytes zero
E.MUL.ADD.X.I.16.C	Ensemble multiply add extract immediate signed doublets ceiling
E.MUL.ADD.X.I.16.F	Ensemble multiply add extract immediate signed doublets floor
E.MUL.ADD.X.I.16.N	Ensemble multiply add extract immediate signed doublets nearest
E.MUL.ADD.X.I.16.Z	Ensemble multiply add extract immediate signed doublets zero
E.MUL.ADD.X.I.32.C	Ensemble multiply add extract immediate signed quadlets ceiling
E.MUL.ADD.X.I.32.F	Ensemble multiply add extract immediate signed quadlets floor

FIG. 98A *continued*

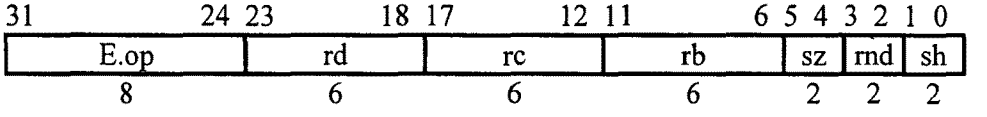
E.MUL.ADD.X.I.32.N	Ensemble multiply add extract immediate signed quadlets nearest
E.MUL.ADD.X.I.32.Z	Ensemble multiply add extract immediate signed quadlets zero
E.MUL.ADD.X.I.64.C	Ensemble multiply add extract immediate signed octlets ceiling
E.MUL.ADD.X.I.64.F	Ensemble multiply add extract immediate signed octlets floor
E.MUL.ADD.X.I.64.N	Ensemble multiply add extract immediate signed octlets nearest
E.MUL.ADD.X.I.64.Z	Ensemble multiply add extract immediate signed octlets zero
E.MUL.ADD.X.I.U. 8.C	Ensemble multiply add extract immediate unsigned bytes ceiling
E.MUL.ADD.X.I.U. 8.F	Ensemble multiply add extract immediate unsigned bytes floor
E.MUL.ADD.X.I.U. 8.N	Ensemble multiply add extract immediate unsigned bytes nearest
E.MUL.ADD.X.I.U.16.C	Ensemble multiply add extract immediate unsigned doublets ceiling
E.MUL.ADD.X.I.U.16.F	Ensemble multiply add extract immediate unsigned doublets floor
E.MUL.ADD.X.I.U.16.N	Ensemble multiply add extract immediate unsigned doublets nearest
E.MUL.ADD.X.I.U.32.C	Ensemble multiply add extract immediate unsigned quadlets ceiling
E.MUL.ADD.X.I.U.32.F	Ensemble multiply add extract immediate unsigned quadlets floor
E.MUL.ADD.X.I.U.32.N	Ensemble multiply add extract immediate unsigned quadlets nearest
E.MUL.ADD.X.I.U.64.C	Ensemble multiply add extract immediate unsigned octlets ceiling
E.MUL.ADD.X.I.U.64.F	Ensemble multiply add extract immediate unsigned octlets floor
E.MUL.ADD.X.I.U.64.N	Ensemble multiply add extract immediate unsigned octlets nearest

FIG. 98A *continued*

Format

E.op.size.rnd rd@rc,rb,i

rd=eopsizernd(rd,rc,rb,i)




```
sz ← log(size) - 3
case op of
        E.MUL.ADD.X.I:
                sh ← size - i - 1
        E.MUL.ADD.X.I.U, E.MUL.ADD.X.I.M, E.MUL.ADD.X.I.C:
                sh ← size - i
endcase
```

FIG. 98B

Definition

```
def mul(size,h,vs,v,i,ws,w,j) as
    mul ← ((vs&vsize-1+i)h-size || vsize-1+i..i) * ((ws&wsize-1+j)h-size || wsize-1+j..j)
enddef

def EnsembleExtractImmediateInplace(op,rnd,size,rd,rc,rb,sh)
    d ← RegRead(rd, 128)
    c ← RegRead(rc, 128)
    b ← RegRead(rb, 128)
    case op of
        E.MUL.ADD.X.I, E.MUL.ADD.X.I.C:
            ds ← 1
            cs ← 1
            bs ← 1
        E.MUL.ADD.X.I.M:
            ds ← 1
            cs ← 0
            bs ← 1
        E.MUL.ADD.X.I.U:
            ds ← 0
            cs ← 0
            bs ← 0
            if rnd = Z then
                raise ReservedInstruction
            endif
    endcase
    case op of
        E.MUL.ADD.X.I, E.MUL.ADD.X.I.U, E.MUL.ADD.X.I.M:
            h ← 2*size + 1
        E.MUL.ADD.X.I.C:
            h ← (2*size) + 2
    endcase
    r ← h - size - sh - 1 - (cs and bs)
    for i ← 0 to 128-size by size
        di ← ((ds and di+size-1)h-size-r||(di+size-1..i)||0r)
        case op of
            E.MUL.ADD.X.I, E.MUL.ADD.X.I.M, E.MUL.ADD.X.I.U:
                p ← mul(size,h,cs,c,i,bs,b,i) + di
```

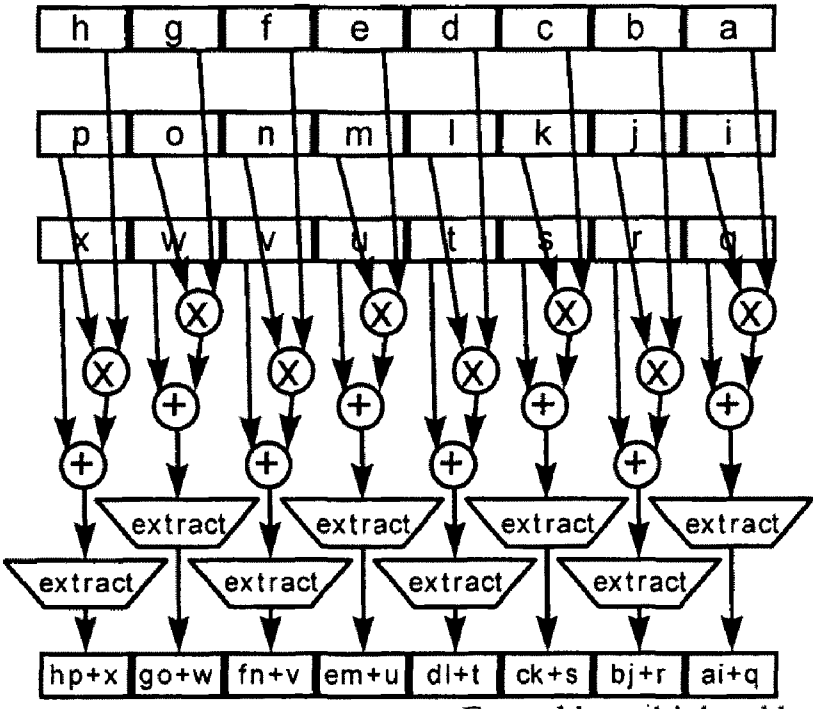
FIG. 98C

```
            E.MUL.ADD.X.I.C:
                if i & size = 0 then
                    p ← mul(size,h,cs,c,i,bs,b,i) - mul(size,h,cs,c,i+size,bs,b,i+size) + di
                else
                    p ← mul(size,h,cs,c,i,bs,b,i+size) + mul(size,h,cs,c,i,bs,b,i+size) + di
                endif
        endcase
        case rnd of
            none, N:
                s ← 0h-r || ~pr || pr-1
            Z:
                s ← 0h-r || ph-1
            F:
                s ← 0h
            C:
                s ← 0h-r || 1r
        endcase
        v ← ((ds & ph-1)||p) + (0||s)
        if (vh..r+size = (ds & vr+size-1)h+1-r-size then
                asize-1+i..i ← vsize-1+r..r
        else
                asize-1+i..i ← ds ? (vh || ~ vhsize-1) : 1size
        endif
    endfor
    RegWrite(rd, 128, a)
enddef
```

Exceptions

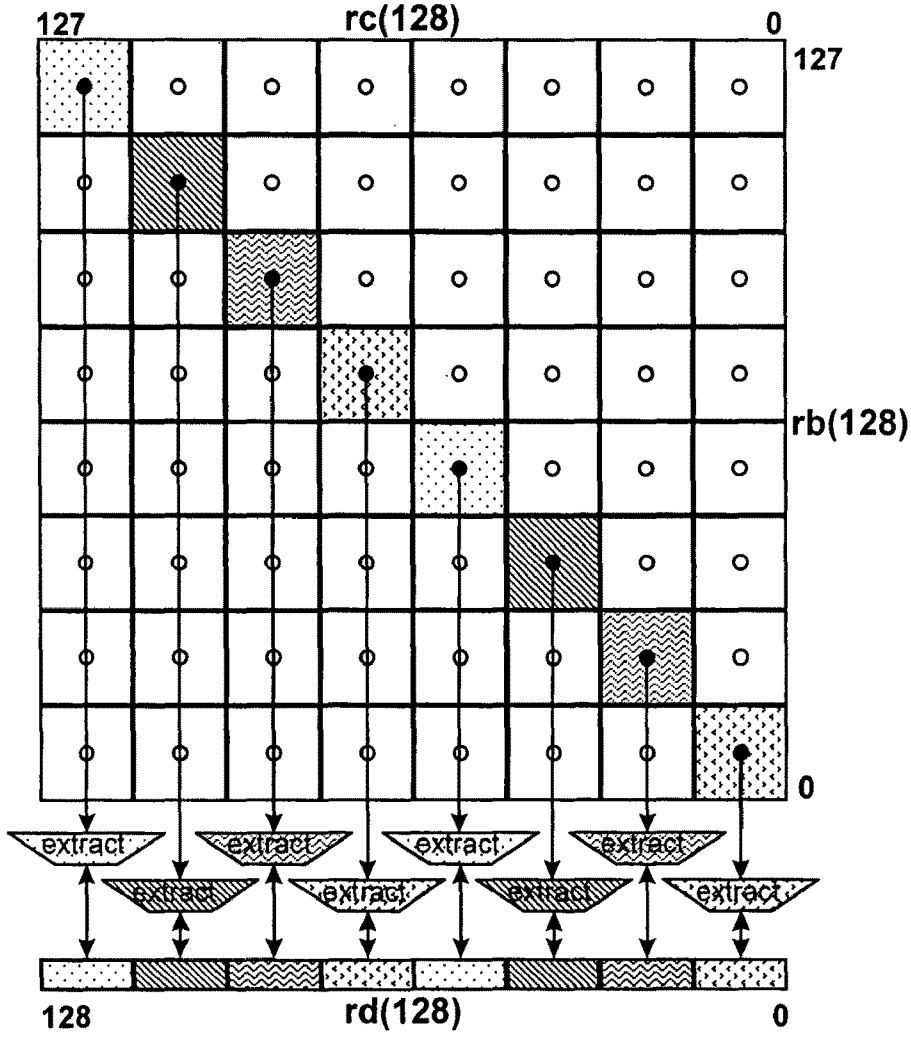
ReservedInstruction

FIG. 98C *continued*



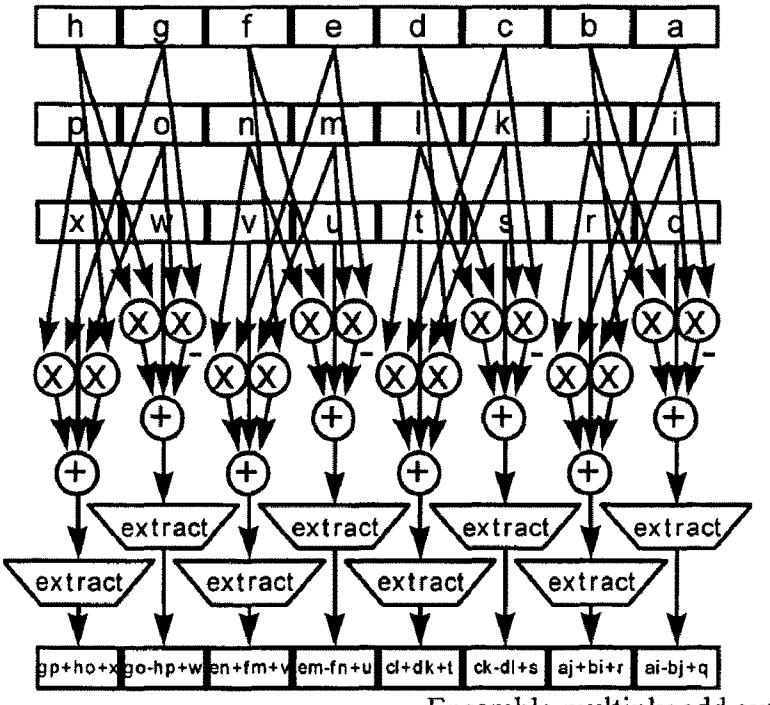
Ensemble multiply add extract immediate doublets

FIG. 98D



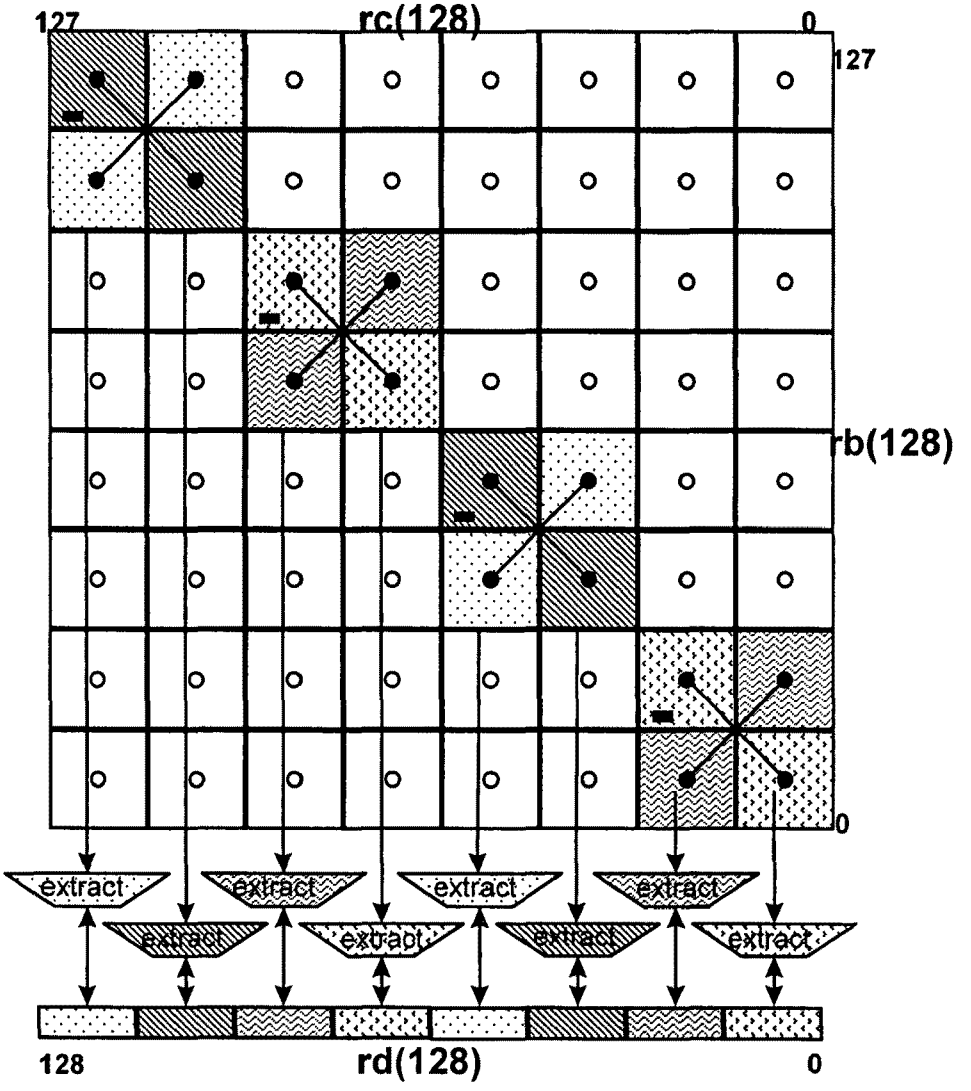
Ensemble multiply add extract immediate doublets

FIG. 98E



Ensemble multiply add extract immediate complex doublets

FIG. 98F



Ensemble multiply add extract immediate complex doublets

FIG. 98G

Operation codes

E.MUL.ADD. 8	Ensemble multiply signed bytes add doublets
E.MUL.ADD.16	Ensemble multiply signed doublets add quadlets
E.MUL.ADD.32	Ensemble multiply signed quadlets add octlets
E.MUL.ADD.64	Ensemble multiply signed octlets add hexlet
E.MUL.ADD.C. 8	Ensemble multiply complex bytes add doublets
E.MUL.ADD.C.16	Ensemble multiply complex doublets add quadlets
E.MUL.ADD.C.32	Ensemble multiply complex quadlets add octlets
E.MUL.ADD.M. 8	Ensemble multiply mixed-signed bytes add doublets
E.MUL.ADD.M.16	Ensemble multiply mixed-signed doublets add quadlets
E.MUL.ADD.M.32	Ensemble multiply mixed-signed quadlets add octlets
E.MUL.ADD.M.64	Ensemble multiply mixed-signed octlets add hexlet
E.MUL.ADD.U. 8	Ensemble multiply unsigned bytes add doublets
E.MUL.ADD.U.16	Ensemble multiply unsigned doublets add quadlets
E.MUL.ADD.U.32	Ensemble multiply unsigned quadlets add octlets
E.MUL.ADD.U.64	Ensemble multiply unsigned octlets add hexlet
E.MUL.SUB. 8	Ensemble multiply signed bytes subtract doublets
E.MUL.SUB.16	Ensemble multiply signed doublets subtract quadlets
E.MUL.SUB.32	Ensemble multiply signed quadlets subtract octlets
E.MUL.SUB.64	Ensemble multiply signed octlets subtract hexlet
E.MUL.SUB.C. 8	Ensemble multiply complex bytes subtract doublets
E.MUL.SUB.C.16	Ensemble multiply complex doublets subtract quadlets
E.MUL.SUB.C.32	Ensemble multiply complex quadlets subtract octlets
E.MUL.SUB.M. 8	Ensemble multiply mixed-signed bytes subtract doublets
E.MUL.SUB.M.16	Ensemble multiply mixed-signed doublets subtract quadlets
E.MUL.SUB.M.32	Ensemble multiply mixed-signed quadlets subtract octlets
E.MUL.SUB.M.64	Ensemble multiply mixed-signed octlets subtract hexlet
E.MUL.SUB.U. 8	Ensemble multiply unsigned bytes subtract doublets
E.MUL.SUB.U.16	Ensemble multiply unsigned doublets subtract quadlets
E.MUL.SUB.U.32	Ensemble multiply unsigned quadlets subtract octlets
E.MUL.SUB.U.64	Ensemble multiply unsigned octlets subtract hexlet

FIG. 99A

Selection

class	op	type	prec
multiply	E.MUL.ADD	NONE M U	8 16 32 64
complex multiply	E.MUL.SUB	C	8 16 32

Format

E.op.size rd=rc,rb

rd=gopsize(rd,rc,rb)

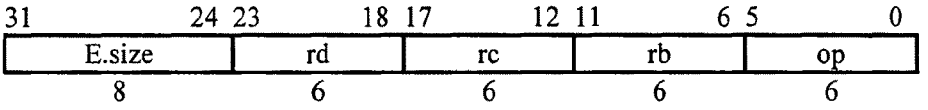


FIG. 99B

Definition

```
def mul(size,h,vs,v,i,ws,w,j) as
    mul ← ((vs&vsize-1+i)h-size || vsize-1+i..i) * ((ws&wsize-1+j)h-size || wsize-1+j..j)
enddef

def EnsembleInplace(op,size,rd,rc,rb) as
    if size=1 then
        raise ReservedInstruction
    endif
    d ← RegRead(rd, 128)
    c ← RegRead(rc, 128)
    b ← RegRead(rb, 128)
    case op of
        E.MUL.ADD, E.MULSUB, E.MUL.ADDC, E.MULSUBC:
            cs ← 1
            bs ← 1
        E.MUL.ADDM, E.MULSUBM:
            cs ← 0
            bs ← 1
        E.MUL.ADDU, E.MULSUBU:
            cs ← 0
            bs ← 0
    endcase
    h ← 2*size
    for i ← 0 to 64-size by size
        di ← d2*(i+size)-1..2*i
        case op of
            E.MUL.ADD, E.MUL.ADDU, E.MUL.ADDM:
                p ← mul(size,h,cs,c,i,bs,b,i) + di
            E.MUL.ADDC:
                if i & size = 0 then
                    p ← mul(size,h,cs,c,i,bs,b,i) - mul(size,h,cs,c,i+size,bs,b,i+size) + di
                else
                    p ← mul(size,h,cs,c,i,bs,b,i+size) + mul(size,h,cs,c,i,bs,b,i+size) + di
                endif
            E.MULSUB, E.MULSUB.U, E.MULSUB.M:
                p ← mul(size,h,cs,c,i,bs,b,i) - di
```

FIG. 99C

```
            E.MULSUBC:
                if i & size = 0 then
                    p ← mul(size,h,cs,c,i,bs,b,i) - mul(size,h,cs,c,i+size,bs,b,i+size) - di
                else
                    p ← mul(size,h,cs,c,i,bs,b,i+size) + mul(size,h,cs,c,i,bs,b,i+size) - di
                endif
        endcase
        a2*(i+size)-1..2*i ← p
    endfor
    RegWrite(rd, 128, a)
enddef
```

Exceptions

none

FIG. 99C *continued*

Operation codes

W.MUL.MAT. 8.B	Wide multiply matrix signed byte big-endian
W.MUL.MAT. 8.L	Wide multiply matrix signed byte little-endian
W.MUL.MAT.16.B	Wide multiply matrix signed doublet big-endian
W.MUL.MAT.16.L	Wide multiply matrix signed doublet little-endian
W.MUL.MAT.32.B	Wide multiply matrix signed quadlet big-endian
W.MUL.MAT.32.L	Wide multiply matrix signed quadlet little-endian
W.MUL.MAT.C. 8.B	Wide multiply matrix signed complex byte big-endian
W.MUL.MAT.C. 8.L	Wide multiply matrix signed complex byte little-endian
W.MUL.MAT.C.16.B	Wide multiply matrix signed complex doublet big-endian
W.MUL.MAT.C.16.L	Wide multiply matrix signed complex doublet little-endian
W.MUL.MAT.M. 8.B	Wide multiply matrix mixed-signed byte big-endian
W.MUL.MAT.M. 8.L	Wide multiply matrix mixed-signed byte little-endian
W.MUL.MAT.M.16.B	Wide multiply matrix mixed-signed doublet big-endian
W.MUL.MAT.M.16.L	Wide multiply matrix mixed-signed doublet little-endian
W.MUL.MAT.M.32.B	Wide multiply matrix mixed-signed quadlet big-endian
W.MUL.MAT.M.32.L	Wide multiply matrix mixed-signed quadlet little-endian
W.MUL.MAT.P. 8.B	Wide multiply matrix polynomial byte big-endian
W.MUL.MAT.P. 8.L	Wide multiply matrix polynomial byte little-endian
W.MUL.MAT.P.16.B	Wide multiply matrix polynomial doublet big-endian
W.MUL.MAT.P.16.L	Wide multiply matrix polynomial doublet little-endian
W.MUL.MAT.P.32.B	Wide multiply matrix polynomial quadlet big-endian
W.MUL.MAT.P.32.L	Wide multiply matrix polynomial quadlet little-endian
W.MUL.MAT.U. 8.B	Wide multiply matrix unsigned byte big-endian
W.MUL.MAT.U. 8.L	Wide multiply matrix unsigned byte little-endian
W.MUL.MAT.U.16.B	Wide multiply matrix unsigned doublet big-endian
W.MUL.MAT.U.16.L	Wide multiply matrix unsigned doublet little-endian
W.MUL.MAT.U.32.B	Wide multiply matrix unsigned quadlet big-endian
W.MUL.MAT.U.32.L	Wide multiply matrix unsigned quadlet little-endian

FIG. 100A

Selection

class	op	type	size	order
multiply	W.MUL.MAT.	NONE M U P	8 16 32	B L
		C	8 16	B L

Format

W.op.size.order rd=rc,rb

rd=wopsizeorder(rc,rb)

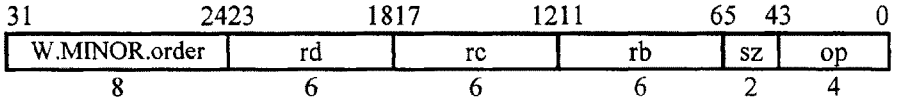


FIG. 100B

Definition

```
def mul(size,h,vs,v,i,ws,w,j) as
    mul ← ((vs&v_size-1+i)^h-size || v_size-1+i..i) * ((ws&w_size-1+j)^h-size || w_size-1+j..j)
enddef

def c ← PolyMultiply(size,a,b) as
    p[0] ← 0^2*size
    for k ← 0 to size-1
        p[k+1] ← p[k] ^ a_k ? (0^size-k || b || 0^k) : 0^2*size
    endfor
    c ← p[size]
enddef

def MemoryMultiply(major,op,gsize,rd,rc,rb)
    d ← RegRead(rd, 128)
    c ← RegRead(rc, 64)
    b ← RegRead(rb, 128)
    lgsize ← log(gsize)
    if c_lgsize-4..0 ≠ 0 then
        raise AccessDisallowedByVirtualAddress
    endif
    if c_2..lgsize-3 ≠ 0 then
        wsize ← (c and (0-c)) || 0^4
        t ← c and (c-1)
    else
        wsize ← 64
        t ← a
    endif
    lwsize ← log(wsize)
    if t_lwsize+6-lgsize..lwsize-3 ≠ 0 then
        msize ← (t and (0-t)) || 0^4
        VirtAddr ← t and (t-1)
    else
        msize ← 128*wsize/gsize
        VirtAddr ← t
    endif
    case major of
        W.MINOR.B:
            order ← B
        W.MINOR.L:
            order ← L
```

FIG. 100C

```
    endcase
    case op of
        W.MUL.MAT.U.8, W.MUL.MAT.U.16, W.MUL.MAT.U.32, W.MUL.MAT.U.64:
            ms ← bs ← 0
        W.MUL.MAT.M.8, W.MUL.MAT.M.16, W.MUL.MAT.M.32, W.MUL.MAT.M.64:
            ms ← 0
            bs ← 1
        W.MUL.MAT.8, W.MUL.MAT.16, W.MUL.MAT.32, W.MUL.MAT.64,
        W.MUL.MAT.C.8, W.MUL.MAT.C.16, W.MUL.MAT.C.32, W.MUL.MAT.C.64:
            ms ← bs ← 1
        W.MUL.MAT.P.8, W.MUL.MAT.P.16, W.MUL.MAT.P.32, W.MUL.MAT.P.64:
    endcase
    m ← LoadMemory(c,VirtAddr,msize,order)
    h ← 2*gsize
    for i ← 0 to wsize-gsize by gsize
        q[0] ← 0^(2*gsize)
        for j ← 0 to vsize-gsize by gsize
            case op of
                W.MUL.MAT.P.8, W.MUL.MAT.P.16, W.MUL.MAT.P.32,
W.MUL.MAT.P.64:
                    k ← i+wsize*j8..lgsize
                    q[j+gsize] ← q[j] ^ PolyMultiply(gsize,mk+gsize-1..k,bj+gsize-1..j)
                W.MUL.MAT.C.8, W.MUL.MAT.C.16, W.MUL.MAT.C.32,
W.MUL.MAT.C.64:
                    if (~i) & j & gsize = 0 then
                        k ← i-(j&gsize)+wsize*j8..lgsize+1
                        q[j+gsize] ← q[j] + mul(gsize,h,ms,m,k,bs,b,j)
                    else
                        k ← i+gsize+wsize*j8..lgsize+1
                        q[j+gsize] ← q[j] - mul(gsize,h,ms,m,k,bs,b,j)
                    endif
                W.MUL.MAT.8, W.MUL.MAT.16, W.MUL.MAT.32, W.MUL.MAT.64,
                W.MUL.MAT.M.8, W.MUL.MAT.M.16, W.MUL.MAT.M.32,
W.MUL.MAT.M.64,
                W.MUL.MAT.U.8, W.MUL.MAT.U.16, W.MUL.MAT.U.32,
W.MUL.MAT.U.64:
                    q[j+gsize] ← q[j] + mul(gsize,h,ms,m,i+wsize*j8..lgsize,bs,b,j)
        endfor
        a2*gsize-1+2*i..2*i ← q[vsize]
    endfor
    a127..2*wsize ← 0
```

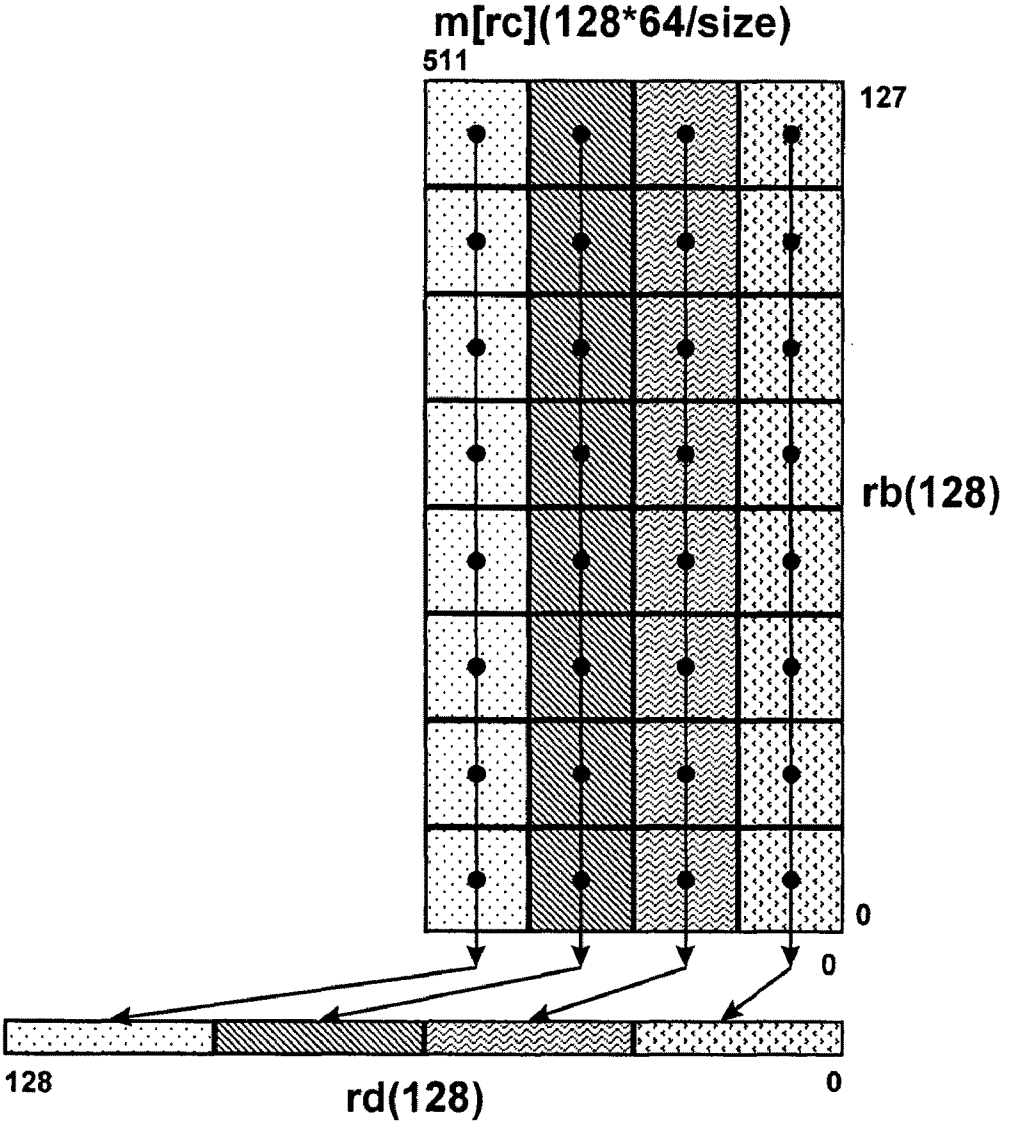
FIG. 100C *continued*

```
    RegWrite(rd, 128, a)
enddef
```

Exceptions

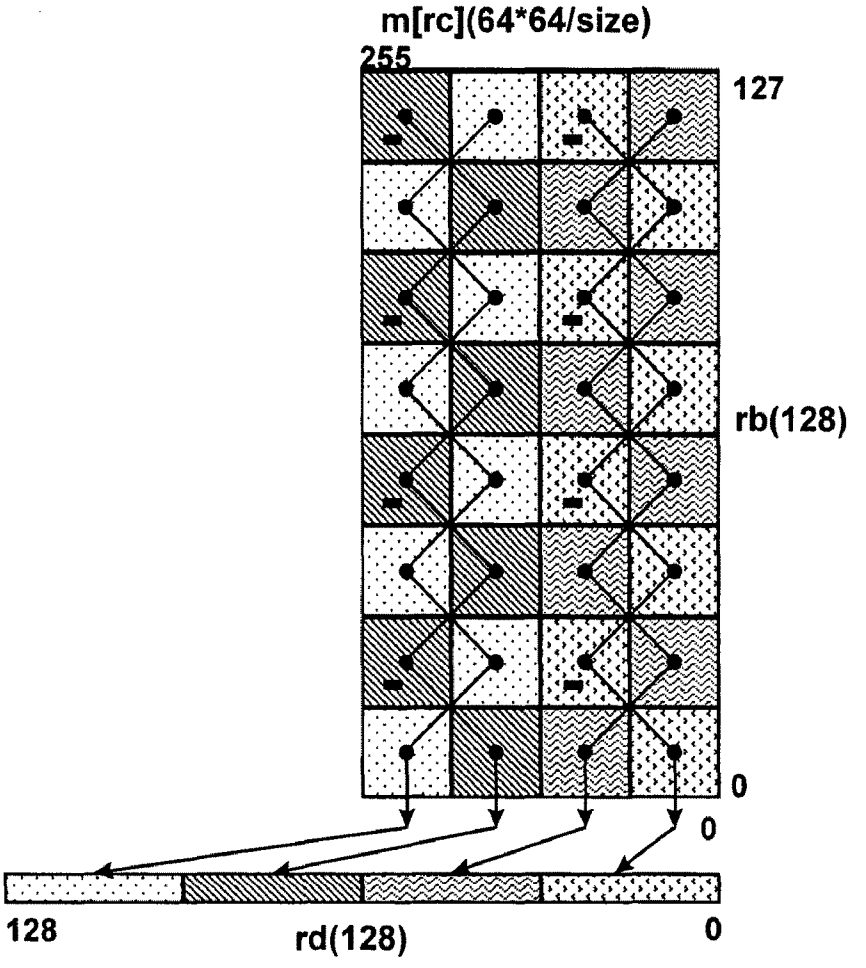
Access disallowed by virtual address
Access disallowed by tag
Access disallowed by global TB
Access disallowed by local TB
Access detail required by tag
Access detail required by local TB
Access detail required by global TB
Local TB miss
Global TB miss

FIG. 100C *continued*



Wide multiply matrix

FIG. 100D



Wide multiply matrix complex

FIG. 100E

Operation codes

W.MUL.MAT.X.B	Wide multiply matrix extract big-endian
W.MUL.MAT.X.L	Wide multiply matrix extract little-endian

FIG. 101A

Format

op ra=rc,rd,rb

ra=op(rc,rd,rb)

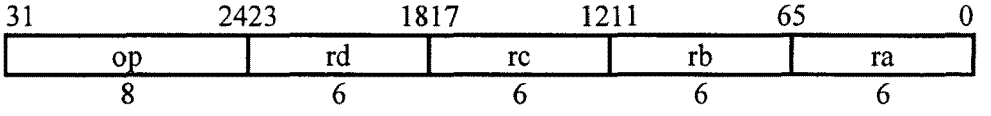


FIG. 101B

Definition

```
def mul(size,h,vs,v,i,ws,w,j) as
    mul ← ((vs&v_size-1+i)^h-size || v_size-1+i..i) * ((ws&w_size-1+j)^h-size || w_size-1+j..j)
enddef

def WideMultiplyExtractMatrix(op,ra,rb,rc,rd)
    d ← RegRead(rd, 128)
    c ← RegRead(rc, 64)
    b ← RegRead(rb, 128)
    case b_8..0 of
        0..255:
            sgsize ← 128
        256..383:
            sgsize ← 64
        384..447:
            sgsize ← 32
        448..479:
            sgsize ← 16
        480..495:
            sgsize ← 8
        496..503:
            sgsize ← 4
        504..507:
            sgsize ← 2
        508..511:
            sgsize ← 1
    endcase
    l ← b_11
    m ← b_12
    n ← b_13
    signed ← b_14
    if c_3..0 ≠ 0 then
        wsize ← (c and (0-c)) || 0^4
        t ← c and (c-1)
    else
        wsize ← 128
        t ← c
    endif
    if sgsize < 8 then
        gsize ← 8
    elseif sgsize > wsize/2 then
```

FIG. 101C

```
        gsize ← wsize/2
    else
        gsize ← sgsize
    endif
    lgsize ← log(gsize)
    lwsize ← log(wsize)
    if t[lwsize+6-n-lgsize..lwsize-3 ≠ 0 then
        msize ← (t and (0-t)) || 0^4
        VirtAddr ← t and (t-1)
    else
        msize ← 64*(2-n)*wsize/gsize
        VirtAddr ← t
    endif
    vsize ← (1+n)*msize*gsize/wsize
    mm ← LoadMemory(c,VirtAddr,msize,order)
    h ← (2*gsize) + 7 - lgsize
    lmsize ← log(msize)
    if (VirtAddr[lmsize-4..0 ≠ 0 then
        raise AccessDisallowedByVirtualAddress
    endif
    case op of
        W.MUL.MAT.X.B:
            order ← B
        W.MUL.MAT.X.L:
            order ← L
    endcase
    ms ← signed
    ds ← signed ^ m
    as ← signed or m
    spos ← (b8..0) and (2*gsize-1)
    dpos ← (0 || b23..16) and (gsize-1)
    r ← spos
    sfsize ← (0 || b31..24) and (gsize-1)
    tfsize ← (sfsize = 0) or ((sfsize+dpos) > gsize) ? gsize-dpos : sfsize
    fsize ← (tfsize + spos > h) ? h - spos : tfsize
    if (b10..9 = Z) & ~signed then
        rnd ← F
    else
        rnd ← b10..9
    endif
    for i ← 0 to wsize-gsize by gsize
```

FIG. 101C *continued*

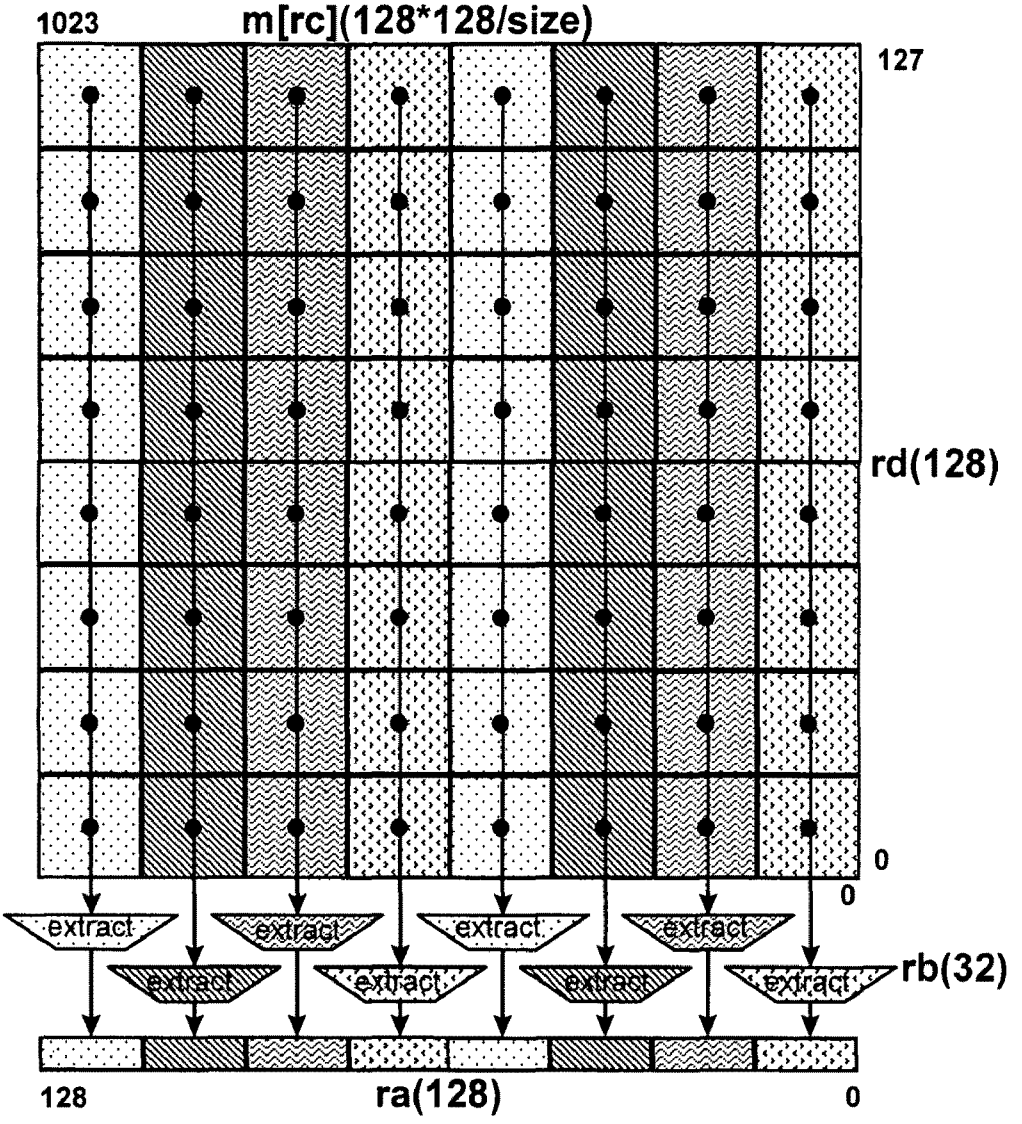
```
        q[0] ← 0^(2*gsize+7-lgsize)
        for j ← 0 to vsize-gsize by gsize
            if n then
                if (~i) & j & gsize = 0 then
                    k ← i-(j&gsize)+wsize*j8..lgsize+1
                    q[j+gsize] ← q[j] + mul(gsize,h,ms,mm,k,ds,d,j)
                else
                    k ← i+gsize+wsize*j8..lgsize+1
                    q[j+gsize] ← q[j] - mul(gsize,h,ms,mm,k,ds,d,j)
                endif
            else
                q[j+gsize] ← q[j] + mul(gsize,h,ms,mm,i+j*wsize/gsize,ds,d,j)
            endif
        endfor
        p ← q[128]
        case rnd of
            none, N:
                s ← 0^(h-r) || ~p_r || p_r^(r-1)
            Z:
                s ← 0^(h-r) || p_h-1^r
            F:
                s ← 0^h
            C:
                s ← 0^(h-r) || 1^r
        endcase
        v ← ((ds & p_h-1)||p) + (0||s)
        if (v_h..r+fsize = (as & v_r+fsize-1)^(h+1-r-fsize)) or not l then
            w ← (as & v_r+fsize-1)^(gsize-fsize-dpos) || v_fsize-1+r..r || 0^dpos
        else
            w ← (s ? (v_h || ~v_h^(gsize-dpos-1)) : 1^(gsize-dpos)) || 0^dpos
        endif
        a_size-1+i..i ← w
    endfor
    a_127..wsize ← 0
    RegWrite(ra, 128, a)
enddef
```

FIG. 101C *continued*

Exceptions

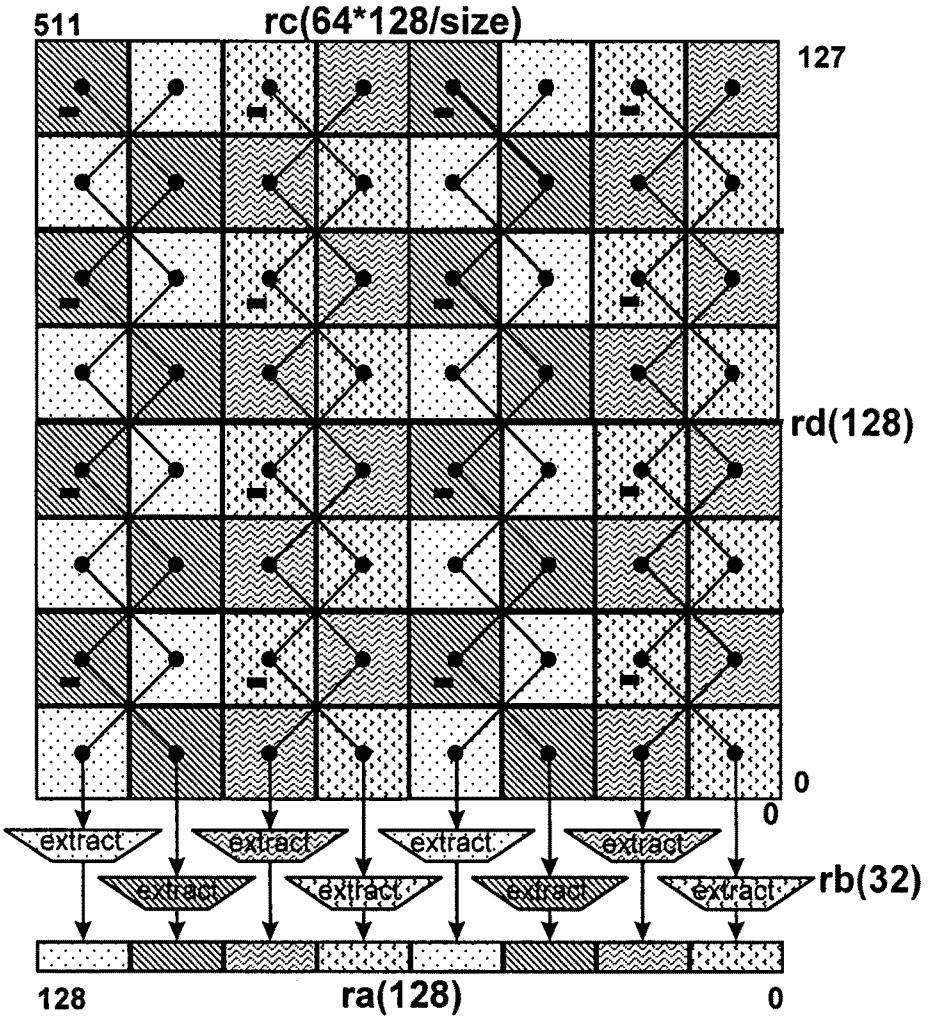
Access disallowed by virtual address
Access disallowed by tag
Access disallowed by global TB
Access disallowed by local TB
Access detail required by tag
Access detail required by local TB
Access detail required by global TB
Local TB miss
Global TB miss

FIG. 101C *continued*



Wide multiply extract matrix doublets

FIG. 101D



Wide multiply extract matrix complex doublets

FIG. 101E

Operation codes

W.MUL.MAT.X.I. 8.C.B	Wide multiply matrix extract immediate signed bytes big-endian ceiling
W.MUL.MAT.X.I. 8.C.L	Wide multiply matrix extract immediate signed bytes little-endian ceiling
W.MUL.MAT.X.I. 8.F.B	Wide multiply matrix extract immediate signed bytes big-endian floor
W.MUL.MAT.X.I. 8.F.L	Wide multiply matrix extract immediate signed bytes little-endian floor
W.MUL.MAT.X.I. 8.N.B	Wide multiply matrix extract immediate signed bytes big-endian nearest
W.MUL.MAT.X.I. 8.N.L	Wide multiply matrix extract immediate signed bytes little-endian nearest
W.MUL.MAT.X.I. 8.Z.B	Wide multiply matrix extract immediate signed bytes big-endian zero
W.MUL.MAT.X.I. 8.Z.L	Wide multiply matrix extract immediate signed bytes little-endian zero
W.MUL.MAT.X.I.16.C.B	Wide multiply matrix extract immediate signed doublets big-endian ceiling
W.MUL.MAT.X.I.16.C.L	Wide multiply matrix extract immediate signed doublets little-endian ceiling
W.MUL.MAT.X.I.16.F.B	Wide multiply matrix extract immediate signed doublets big-endian floor
W.MUL.MAT.X.I.16.F.L	Wide multiply matrix extract immediate signed doublets little-endian floor
W.MUL.MAT.X.I.16.N.B	Wide multiply matrix extract immediate signed doublets big-endian nearest
W.MUL.MAT.X.I.16.N.L	Wide multiply matrix extract immediate signed doublets little-endian nearest
W.MUL.MAT.X.I.16.Z.B	Wide multiply matrix extract immediate signed doublets big-endian zero
W.MUL.MAT.X.I.16.Z.L	Wide multiply matrix extract immediate signed doublets little-endian zero
W.MUL.MAT.X.I.32.C.B	Wide multiply matrix extract immediate signed quadlets big-endian ceiling
W.MUL.MAT.X.I.32.C.L	Wide multiply matrix extract immediate signed quadlets little-endian ceiling
W.MUL.MAT.X.I.32.F.B	Wide multiply matrix extract immediate signed quadlets big-endian floor
W.MUL.MAT.X.I.32.F.L	Wide multiply matrix extract immediate signed quadlets little-endian floor

FIG. 102A

W.MUL.MAT.X.I.32.N.B	Wide multiply matrix extract immediate signed quadlets big-endian nearest
W.MUL.MAT.X.I.32.N.L	Wide multiply matrix extract immediate signed quadlets little-endian nearest
W.MUL.MAT.X.I.32.Z.B	Wide multiply matrix extract immediate signed quadlets big-endian zero
W.MUL.MAT.X.I.32.Z.L	Wide multiply matrix extract immediate signed quadlets little-endian zero
W.MUL.MAT.X.I.64.C.B	Wide multiply matrix extract immediate signed octlets big-endian ceiling
W.MUL.MAT.X.I.64.C.L	Wide multiply matrix extract immediate signed octlets little-endian ceiling
W.MUL.MAT.X.I.64.F.B	Wide multiply matrix extract immediate signed octlets big-endian floor
W.MUL.MAT.X.I.64.F.L	Wide multiply matrix extract immediate signed octlets little-endian floor
W.MUL.MAT.X.I.64.N.B	Wide multiply matrix extract immediate signed octlets big-endian nearest
W.MUL.MAT.X.I.64.N.L	Wide multiply matrix extract immediate signed octlets little-endian nearest
W.MUL.MAT.X.I.64.Z.B	Wide multiply matrix extract immediate signed octlets big-endian zero
W.MUL.MAT.X.I.64.Z.L	Wide multiply matrix extract immediate signed octlets little-endian zero
W.MUL.MAT.X.I.C. 8.C.B	Wide multiply matrix extract immediate complex bytes big-endian ceiling
W.MUL.MAT.X.I.C. 8.C.L	Wide multiply matrix extract immediate complex bytes little-endian ceiling
W.MUL.MAT.X.I.C. 8.F.B	Wide multiply matrix extract immediate complex bytes big-endian floor
W.MUL.MAT.X.I.C. 8.F.L	Wide multiply matrix extract immediate complex bytes little-endian floor
W.MUL.MAT.X.I.C. 8.N.B	Wide multiply matrix extract immediate complex bytes big-endian nearest
W.MUL.MAT.X.I.C. 8.N.L	Wide multiply matrix extract immediate complex bytes little-endian nearest
W.MUL.MAT.X.I.C. 8.Z.B	Wide multiply matrix extract immediate complex bytes big-endian zero
W.MUL.MAT.X.I.C. 8.Z.L	Wide multiply matrix extract immediate complex bytes little-endian zero
W.MUL.MAT.X.I.C.16.C.B	Wide multiply matrix extract immediate complex doublets big-endian ceiling

FIG. 102A *continued*

W.MUL.MAT.X.I.C.16.C.L	Wide multiply matrix extract immediate complex doublets little-endian ceiling
W.MUL.MAT.X.I.C.16.F.B	Wide multiply matrix extract immediate complex doublets big-endian floor
W.MUL.MAT.X.I.C.16.F.L	Wide multiply matrix extract immediate complex doublets little-endian floor
W.MUL.MAT.X.I.C.16.N.B	Wide multiply matrix extract immediate complex doublets big-endian nearest
W.MUL.MAT.X.I.C.16.N.L	Wide multiply matrix extract immediate complex doublets little-endian nearest
W.MUL.MAT.X.I.C.16.Z.B	Wide multiply matrix extract immediate complex doublets big-endian zero
W.MUL.MAT.X.I.C.16.Z.L	Wide multiply matrix extract immediate complex doublets little-endian zero
W.MUL.MAT.X.I.C.32.C.B	Wide multiply matrix extract immediate complex quadlets big-endian ceiling
W.MUL.MAT.X.I.C.32.C.L	Wide multiply matrix extract immediate complex quadlets little-endian ceiling
W.MUL.MAT.X.I.C.32.F.B	Wide multiply matrix extract immediate complex quadlets big-endian floor
W.MUL.MAT.X.I.C.32.F.L	Wide multiply matrix extract immediate complex quadlets little-endian floor
W.MUL.MAT.X.I.C.32.N.B	Wide multiply matrix extract immediate complex quadlets big-endian nearest
W.MUL.MAT.X.I.C.32.N.L	Wide multiply matrix extract immediate complex quadlets little-endian nearest
W.MUL.MAT.X.I.C.32.Z.B	Wide multiply matrix extract immediate complex quadlets big-endian zero
W.MUL.MAT.X.I.C.32.Z.L	Wide multiply matrix extract immediate complex quadlets little-endian zero
W.MUL.MAT.X.I.C.64.C.B	Wide multiply matrix extract immediate complex octlets big-endian ceiling
W.MUL.MAT.X.I.C.64.C.L	Wide multiply matrix extract immediate complex octlets little-endian ceiling
W.MUL.MAT.X.I.C.64.F.B	Wide multiply matrix extract immediate complex octlets big-endian floor
W.MUL.MAT.X.I.C.64.F.L	Wide multiply matrix extract immediate complex octlets little-endian floor
W.MUL.MAT.X.I.C.64.N.B	Wide multiply matrix extract immediate complex octlets big-endian nearest
W.MUL.MAT.X.I.C.64.N.L	Wide multiply matrix extract immediate complex octlets little-endian nearest

FIG. 102A *continued*

W.MUL.MAT.X.I.C.64.Z.B	Wide multiply matrix extract immediate complex octlets big-endian zero
W.MUL.MAT.X.I.C.64.Z.L	Wide multiply matrix extract immediate complex octlets little-endian zero
W.MUL.MAT.X.I.M.8.C.B	Wide multiply matrix extract immediate mixed-signed bytes big-endian ceiling
W.MUL.MAT.X.I.M.8.C.L	Wide multiply matrix extract immediate mixed-signed bytes little-endian ceiling
W.MUL.MAT.X.I.M.8.F.B	Wide multiply matrix extract immediate mixed-signed bytes big-endian floor
W.MUL.MAT.X.I.M. 8.F.L	Wide multiply matrix extract immediate mixed-signed bytes little-endian floor
W.MUL.MAT.X.I.M.8.N.B	Wide multiply matrix extract immediate mixed-signed bytes big-endian nearest
W.MUL.MAT.X.I.M.8.N.L	Wide multiply matrix extract immediate mixed-signed bytes little-endian nearest
W.MUL.MAT.X.I.M.8.Z.B	Wide multiply matrix extract immediate mixed-signed bytes big-endian zero
W.MUL.MAT.X.I.M.8.Z.L	Wide multiply matrix extract immediate mixed-signed bytes little-endian zero
W.MUL.MAT.X.I.M.16.C.B	Wide multiply matrix extract immediate mixed-signed doublets big-endian ceiling
W.MUL.MAT.X.I.M.16.C.L	Wide multiply matrix extract immediate mixed-signed doublets little-endian ceiling
W.MUL.MAT.X.I.M.16.F.B	Wide multiply matrix extract immediate mixed-signed doublets big-endian floor
W.MUL.MAT.X.I.M.16.F.L	Wide multiply matrix extract immediate mixed-signed doublets little-endian floor
W.MUL.MAT.X.I.M.16.N.B	Wide multiply matrix extract immediate mixed-signed doublets big-endian nearest
W.MUL.MAT.X.I.M.16.N.L	Wide multiply matrix extract immediate mixed-signed doublets little-endian nearest
W.MUL.MAT.X.I.M.16.Z.B	Wide multiply matrix extract immediate mixed-signed doublets big-endian zero
W.MUL.MAT.X.I.M.16.Z.L	Wide multiply matrix extract immediate mixed-signed doublets little-endian zero
W.MUL.MAT.X.I.M.32.C.B	Wide multiply matrix extract immediate mixed-signed quadlets big-endian ceiling
W.MUL.MAT.X.I.M.32.C.L	Wide multiply matrix extract immediate mixed-signed quadlets little-endian ceiling
W.MUL.MAT.X.I.M.32.F.B	Wide multiply matrix extract immediate mixed-signed quadlets big-endian floor

FIG. 102A *continued*

W.MUL.MAT.X.I.M.32.F.L	Wide multiply matrix extract immediate mixed-signed quadlets little-endian floor
W.MUL.MAT.X.I.M.32.N.B	Wide multiply matrix extract immediate mixed-signed quadlets big-endian nearest
W.MUL.MAT.X.I.M.32.N.L	Wide multiply matrix extract immediate mixed-signed quadlets little-endian nearest
W.MUL.MAT.X.I.M.32.Z.B	Wide multiply matrix extract immediate mixed-signed quadlets big-endian zero
W.MUL.MAT.X.I.M.32.Z.L	Wide multiply matrix extract immediate mixed-signed quadlets little-endian zero
W.MUL.MAT.X.I.M.64.C.B	Wide multiply matrix extract immediate mixed-signed octlets big-endian ceiling
W.MUL.MAT.X.I.M.64.C.L	Wide multiply matrix extract immediate mixed-signed octlets little-endian ceiling
W.MUL.MAT.X.I.M.64.F.B	Wide multiply matrix extract immediate mixed-signed octlets big-endian floor
W.MUL.MAT.X.I.M.64.F.L	Wide multiply matrix extract immediate mixed-signed octlets little-endian floor
W.MUL.MAT.X.I.M.64.N.B	Wide multiply matrix extract immediate mixed-signed octlets big-endian nearest
W.MUL.MAT.X.I.M.64.N.L	Wide multiply matrix extract immediate mixed-signed octlets little-endian nearest
W.MUL.MAT.X.I.M.64.Z.B	Wide multiply matrix extract immediate mixed-signed octlets big-endian zero
W.MUL.MAT.X.I.M.64.Z.L	Wide multiply matrix extract immediate mixed-signed octlets little-endian zero
W.MUL.MAT.X.I.U. 8.C.B	Wide multiply matrix extract immediate unsigned bytes big-endian ceiling
W.MUL.MAT.X.I.U. 8.C.L	Wide multiply matrix extract immediate unsigned bytes little-endian ceiling
W.MUL.MAT.X.I.U. 8.F.B	Wide multiply matrix extract immediate unsigned bytes big-endian floor
W.MUL.MAT.X.I.U. 8.F.L	Wide multiply matrix extract immediate unsigned bytes little-endian floor
W.MUL.MAT.X.I.U. 8.N.B	Wide multiply matrix extract immediate unsigned bytes big-endian nearest
W.MUL.MAT.X.I.U. 8.N.L	Wide multiply matrix extract immediate unsigned bytes little-endian nearest
W.MUL.MAT.X.I.U.16.C.B	Wide multiply matrix extract immediate unsigned doublets big-endian ceiling
W.MUL.MAT.X.I.U.16.C.L	Wide multiply matrix extract immediate unsigned doublets little-endian ceiling

FIG. 102A *continued*

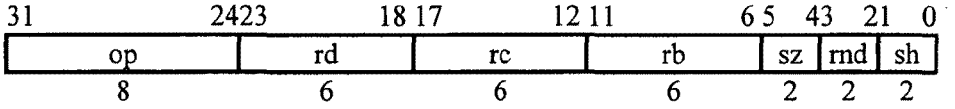
W.MUL.MAT.X.I.U.16.F.B	Wide multiply matrix extract immediate unsigned doublets big-endian floor
W.MUL.MAT.X.I.U.16.F.L	Wide multiply matrix extract immediate unsigned doublets little-endian floor
W.MUL.MAT.X.I.U.16.N.B	Wide multiply matrix extract immediate unsigned doublets big-endian nearest
W.MUL.MAT.X.I.U.16.N.L	Wide multiply matrix extract immediate unsigned doublets little-endian nearest
W.MUL.MAT.X.I.U.32.C.B	Wide multiply matrix extract immediate unsigned quadlets big-endian ceiling
W.MUL.MAT.X.I.U.32.C.L	Wide multiply matrix extract immediate unsigned quadlets little-endian ceiling
W.MUL.MAT.X.I.U.32.F.B	Wide multiply matrix extract immediate unsigned quadlets big-endian floor
W.MUL.MAT.X.I.U.32.F.L	Wide multiply matrix extract immediate unsigned quadlets little-endian floor
W.MUL.MAT.X.I.U.32.N.B	Wide multiply matrix extract immediate unsigned quadlets big-endian nearest
W.MUL.MAT.X.I.U.32.N.L	Wide multiply matrix extract immediate unsigned quadlets little-endian nearest
W.MUL.MAT.X.I.U.64.C.B	Wide multiply matrix extract immediate unsigned octlets big-endian ceiling
W.MUL.MAT.X.I.U.64.C.L	Wide multiply matrix extract immediate unsigned octlets little-endian ceiling
W.MUL.MAT.X.I.U.64.F.B	Wide multiply matrix extract immediate unsigned octlets big-endian floor
W.MUL.MAT.X.I.U.64.F.L	Wide multiply matrix extract immediate unsigned octlets little-endian floor
W.MUL.MAT.X.I.U.64.N.B	Wide multiply matrix extract immediate unsigned octlets big-endian nearest
W.MUL.MAT.X.I.U.64.N.L	Wide multiply matrix extract immediate unsigned octlets little-endian nearest

FIG. 102A *continued*

Format

W.op.size.rnd rd=rc,rb,i

rd=wopsizernd(rc,rb,i)



sz ← log(size) - 3
case op of
    W.MUL.MAT.X.I, W.MUL.MAT.X.I.C:
        assert size + 6 - log(size) ≥ i ≥ size + 6 - log(size) - 3
        sh ← size + 6 - log(size) - i
    W.MUL.MAT.X.I.M, W.MUL.MAT.X.I.U:
        assert size + 7 - log(size) ≥ i ≥ size + 7 - log(size) - 3
        sh ← size + 7 - log(size) - i
endcase

FIG. 102B

Definition

```
def mul(size,h,vs,v,i,ws,w,j) as
    mul ← ((vs&v_{size-1+i})^{h-size} || v_{size-1+i..i}) * ((ws&w_{size-1+j})^{h-size} || w_{size-1+j..j})
enddef

def WideMultiplyExtractImmediateMatrix(op,rnd,gsize,rd,rc,rb,sh)
    c ← RegRead(rc, 64)
    b ← RegRead(rb, 128)
    lgsize ← log(gsize)
    case op of
        W.MUL.MAT.X.I.B, WMUL.MAT.X.I.L, WMUL.MAT.X.I.U.B,
WMUL.MAT.X.I.U.L,
        WMUL.MAT.X.I.M.B, WMUL.MAT.X.I.M.L :
            if c_{lgsize-4..0} ≠ 0 then
                raise AccessDisallowedByVirtualAddress
            endif
            if c_{3..lgsize-3} ≠ 0 then
                wsize ← (c and (0-c)) || 0^4
                t ← c and (c-1)
            else
                wsize ← 128
                t ← c
            endif
            lwsize ← log(wsize)
            if t_{lwsize+6-lgsize..lwsize-3} ≠ 0 then
                msize ← (t and (0-t)) || 0^4
                VirtAddr ← t and (t-1)
            else
                msize ← 128*wsize/gsize
                VirtAddr ← t
            endif
            vsize ← msize*gsize/wsize
        W.MUL.MAT.X.I.C.B, W.MUL.MAT.X.I.C.L:
            if c_{lgsize-4..0} ≠ 0 then
                raise AccessDisallowedByVirtualAddress
            endif
            if c_{3..lgsize-3} ≠ 0 then
                wsize ← (c and (0-c)) || 0^4
                t ← c and (c-1)
            else
                wsize ← 128
```

FIG. 102C

```
                t ← c
            endif
            lwsize ← log(wsize)
            if t[lwsize+5-lgsize..lwsize-3 ≠ 0 then
                msize ← (t and (0-t)) || 0^4
                VirtAddr ← t and (t-1)
            else
                msize ← 64*wsize/gsize
                VirtAddr ← t
            endif
            vsize ← 2*msize*gsize/wsize
    endcase
    case op of
        W.MUL.MAT.X.I.B, W.MUL.MAT.X.I.U.B, W.MUL.MAT.X.I.M.B,
W.MUL.MAT.X.I.C.B:
            order ← B
        W.MUL.MAT.X.I.L, W.MUL.MAT.X.I.U.L, W.MUL.MAT.X.I.M.L,
W.MUL.MAT.X.I.C.L:
            order ← L
    endcase
    case op of
        W.MUL.MAT.X.I.U.B, W.MUL.MAT.X.I.U.L:
            as ← ms ← bs ← 0
        W.MUL.MAT.X.I.M.B, W.MUL.MAT.X.I.M.L,:
            bs ← 0
            as ← ms ← 1
        W.MUL.MAT.X.I.B, W.MUL.MAT.X.I.L, W.MUL.MAT.X.I.C.B,
W.MUL.MAT.X.I.C.L:
            as ← ms ← bs ← 1
    endcase
    m ← LoadMemory(c,VirtAddr,msize,order)
    h ← (2*gsize) + 7 - lgsize - (ms and bs)
    r ← h - size - sh
    for i ← 0 to wsize-gsize by gsize
        q[0] ← 0^(2*gsize+7-lgsize)
        for j ← 0 to vsize-gsize by gsize
            case op of
                W.MUL.MAT.X.I.B, W.MUL.MAT.X.I.L,
                W.MUL.MAT.X.I.U.B, W.MUL.MAT.X.I.U.L,
                W.MUL.MAT.X.I.M.B, W.MUL.MAT.X.I.M.L:
                    q[j+gsize] ← q[j] + mul(gsize,h,ms,m,i+wsize*j8..lgsize,bs,b,j)
                W.MUL.MAT.X.I.C.B, W.MUL.MAT.X.I.C.L:
```

FIG. 102C *continued*

```
                    if (~i) & j & gsize = 0 then
                        k ← i-(j&gsize)+wsize*j8..lgsize+1
                        q[j+gsize] ← q[j] + mul(gsize,h,ms,m,k,bs,b,j)
                    else
                        k ← i+gsize+wsize*j8..lgsize+1
                        q[j+gsize] ← q[j] - mul(gsize,h,ms,m,k,bs,b,j)
                    endif
                endcase
            endfor
            p ← q[vsize]
            case rnd of
                none, N:
                    s ← 0^(h-r) || ~p_r || p_r^(r-1)
                Z:
                    s ← 0^(h-r) || p_r^(h-1)
                F:
                    s ← 0^h
                C:
                    s ← 0^(h-r) || 1^r
            endcase
            v ← ((as & p_(h-1))||p) + (0||s)
            if (v_(h..r+gsize) = (as & v_(r+gsize-1))^(h+1-r-gsize) then
                a_(gsize-1+i..i) ← v_(gsize-1+r..r)
            else
                a_(gsize-1+i..i) ← as ? (v_h || ~v_h^(gsize-1)) : 1^gsize
            endif
        endfor
        a127..wsize ← 0
        RegWrite(rd, 128, a)
    enddef
```

Exceptions

Access disallowed by virtual address
Access disallowed by tag
Access disallowed by global TB
Access disallowed by local TB
Access detail required by tag
Access detail required by local TB
Access detail required by global TB
Local TB miss
Global TB miss

FIG. 102C *continued*