```
$PAGE "INCINSTR -- Special MPE/XL HP3000 Instructions"
<<*****************************************************>>
<<                                                     >>
<<                  INCINSTR - M2                      >>
<<                                                     >>
<< SPECIAL stack HP3000 instructions for MPE/XL machines >>
<<                                                     >>
<<   MFVA: Move from virtual address                   >>
<<         S-5     target db relative byte address     >>
<<         S-4     source hi virtual space             >>
<<         S-3     source lo virtual space             >>
<<         S-2     source hi virtual space offset      >>
<<         S-1     source lo virtual space offset      >>
<<         S-0     positive byte count                 >>
<<                                                     >>
<<   MTVA: Move to virtual address                     >>
<<         S-5     target hi virtual space             >>
<<         S-4     target lo virtual space             >>
<<         S-3     target hi virtual space offset      >>
<<         S-2     target lo virtual space offset      >>
<<         S-1     source db relative byte address     >>
<<         S-0     positive byte count                 >>
<<                                                     >>
<<   DPID: Disable PID protection                      >>
<<   EPID: Enable  PID protection                      >>
<<                                                     >>
<<   SWT:  Switch to NM                                >>
<<                                                     >>
<<   BRKP: Breakpoint instruction, last eight bits     >>
<<         are interpreted as an immediate field       >>
<<                                                     >>
<<*****************************************************>>

DEFINE
        MFVA'inst   =  CON  %20440#,
        MTVA'inst   =  CON  %20442#,

        DPID'inst   =  CON  %20444#,
        EPID'inst   =  CON  %20445#,

        SWT'inst    =  CON  %30064#,
        SWT2'inst   =  CON  %30066#,
        SWT3'inst   =  CON  %30067#,

        BRKP'inst   =  CON  %36000#,
        BRKP1'inst  =  CON  %36001#,
        BRKP2'inst  =  CON  %36002#;

DEFINE
        MFVA   =  assemble( MFVA'inst  )#,
        MTVA   =  assemble( MTVA'inst  )#,

        DPID   =  assemble( DPID'inst  )#,
        EPID   =  assemble( EPID'inst  )#,

        SWT    =  assemble( SWT'inst   )#,
        SWT2   =  assemble( SWT2'inst  )#,
        SWT3   =  assemble( SWT3'inst  )#,

        BRKP   =  assemble( BRKP'inst  )#,

        BRKP1  =  assemble( BRKP1'inst )#,
        BRKP2  =  assemble( BRKP2'inst )#;
```

MFVA Move From Virtual Address.

This instruction moves a byte string from a Precision Architecture Virtual Address to an HP3000 DB area. It expects to find a positive byte count in A which represents the size of the block of bytes to be transferred, an offset into a source virtual space in C and D to be used with the source Virtual space ID in E and D to determine the address of the first source data byte, and a DB relative address in F to be used in calculating the target address for the first byte transfer. The stack before execution is as follows:

```
                +----------------------------------+
TOS-5 (F)       |   Target DB-rel byte address     |
                +----------------------------------+
TOS-4 (E)       |   Source Hi virtual space ID     |
                +----------------------------------+
TOS-3 (D)       |   Source Lo virtual space ID     |
                +----------------------------------+
TOS-2 (C)       | Source Hi virtual space offset   |
                +----------------------------------+
TOS-1 (B)       | Source Lo virtual space offset   |
                +----------------------------------+
TOS   (A)       |           Byte count             |
                +----------------------------------+
```

The source address is used directly as given, and the DB relative target byte address is formed by adding the address of DB and the DB relative address in F. Byte wrap-around is implemented, i.e. if not in split stack mode and the target address is above S, then 2**16 is subtracted from the target address. This is the normal byte mechanism to do DB negative addressing. The target byte address thus pointed to receives a byte from the source byte address, and the byte count is decremented. Bytes from the source area continue to be moved to the target area until the byte count reaches zero. The stack then is popped of all parameters. Data is protected by whatever means Native Mode has employed on the virtual address.

```
Opcode: 020440
Indicators: unaffected
Traps: MODE, STUN, BNDV
New for HP3000 Compatibility Mode on
Precision Architecture.
```

```
Begin
 SBA:=(S-4,S-3,S-2,S-1); <<Source Byte Addr. in virtual space>>
 TBA:=(S-5) + DB; <<Target Byte Address>>
 IF (TBA>S AND Not Split Stack)
  THEN TBA:=TBA-2**16;<<DB negative>>
 While (S) > 0
  do
   Begin
    (TBA) := (SBA);
    SBA := SBA + 1;
    TBA := TBA + 1;
    (S) := (S) - 1;
   End;
 S := S - 6;
End;
```

MTVA Move To Virtual Address.

This instruction moves a string of bytes from an HP3000 DB area to a Precision Architecture virtual address. It expects to find a positive byte count in A which represents the size of the block of bytes to be transferred, a DB relative address in B to be used in calculating the source address of the first byte to be transferred, and an offset into a target virtual space in D and C to be used with the target virtual space ID in F and E to determine the address of the first target byte location. The stack before execution is as follows:

```
                +---------------------------------+
TOS-5 (F)       |   Target Hi virtual space ID    |
                +---------------------------------+
TOS-4 (E)       |   Target Lo virtual space ID    |
                +---------------------------------+
TOS-3 (D)       | Target Hi virtual space offset  |
                +---------------------------------+
TOS-2 (C)       | Target Lo virtual space offset  |
                +---------------------------------+
TOS-1 (B)       |  Source DB-rel byte address     |
                +---------------------------------+
TOS   (A)       |           Byte count            |
                +---------------------------------+
```

The DB relative source byte address is formed by adding the address of DB and the DB relative address in B. Byte wrap-around is implemented, i.e., if not in split stack mode and the source address is above S, the 2**16 is subtracted from the source address. This is the normal byte mechanism for achieving DB negative addressing. The target address is used directly as given in F, E, D and C. The source byte pointed to is is moved to the target location, and the byte count is decremented. Bytes from the source area continue to be moved to the target area until the byte count reaches zero. The stack then is popped of all parameters. Data is protected by whatever means Native Mode has employed on the virtual address.

```
        Opcode: 020442
        Indicators: unaffected
        Traps: MODE, STUN, BNDV
        New for HP3000 Compatibility Mode on
        Precision Architecture.

Begin
 TBA := (S-5,S-4,S-3,S-2); <<Target Byte Addr. in virtual space>>
 SBA := (S-1) + DB; <<Source Byte Address>>
 IF (SBA>S AND Not Split Stack)
  THEN SBA:=SBA-2**16;<<DB negative>>
 While (S) > 0
  do
   Begin
    (TBA) := (SBA);
    TBA := TBA + 1;
    SBA := SBA + 1;
    (S) := (S) - 1;
   End;
 S := S - 6;
End;
```

MQFV Move Q relative From Virtual.

This instruction moves a byte string from a Precision Architecture Virtual Address to an HP3000 stack. It expects to find a positive byte count in A which represents the size of the block of bytes to be transferred, an offset into a source virtual space in C and D to be used with the source Virtual space ID in E and D to determine the address of the first source data byte, and a Q relative address in F to be used in calculating the target address for the first byte transfer. The stack before execution is as follows:

```
                   +--------------------------------+
TOS-5 (F)          |    Target Q rel byte address   |
                   +--------------------------------+
TOS-4 (E)          |   Source Hi virtual space ID   |
                   +--------------------------------+
TOS-3 (D)          |   Source Lo virtual space ID   |
                   +--------------------------------+
TOS-2 (C)          | Source Hi virtual space offset |
                   +--------------------------------+
TOS-1 (B)          | Source Lo virtual space offset |
                   +--------------------------------+
TOS   (A)          |           Byte count           |
                   +--------------------------------+
```

The source address is used directly as given, and the Q relative target byte address is formed by adding the byte address of Q and the Q relative address in F. Byte wrap-around is implemented, i.e. if the target address is above S, then 2**16 is subtracted from the target address. This is the mechanism to do Q negative byte addressing. The target byte address thus pointed to receives a byte from the source byte address, and the byte count is decremented. Bytes from the source area continue to be moved to the target area until the byte count reaches zero. The stack then is popped of all parameters. Data is protected by whatever means Native Mode has employed on the virtual address.

```
Opcode: 020441
Indicators: unaffected
Traps: MODE, STUN, BNDV
New for HP3000 Compatibility Mode on
Precision Architecture.
```

```
Begin
 SBA:=(S-4,S-3,S-2,S-1); <<Source Byte Addr. in virtual space>>
 TBA:=(S-5) + Q; <<Target Byte Address>>
 IF TBA>S
  THEN TBA:=TBA-2**16;<<Q negative>>
 While (S) > 0
  do
   Begin
    (TBA) := (SBA);
    SBA := SBA + 1;
    TBA := TBA + 1;
    (S) := (S) - 1;
   End;
 S := S - 6;
End;
```

MQTV Move Q relative To Virtual.

This instruction moves a string of bytes from an HP3000 stack to a Precision Architecture virtual address. It expects to find a positive byte count in A which represents the size of the block of bytes to be transferred, a Q relative address in B to be used in calculating the source address of the first byte to be transferred, and an offset into a target virtual space in D and C to be used with the target virtual space ID in F and E to determine the address of the first target byte location. The stack before execution is as follows:

```
                    +----------------------------------+
TOS-5 (F)           |    Target Hi virtual space ID    |
                    +----------------------------------+
TOS-4 (E)           |    Target Lo virtual space ID    |
                    +----------------------------------+
TOS-3 (D)           | Target Hi virtual space offset   |
                    +----------------------------------+
TOS-2 (C)           | Target Lo virtual space offset   |
                    +----------------------------------+
TOS-1 (B)           |    Source Q rel byte address     |
                    +----------------------------------+
TOS   (A)           |            Byte count            |
                    +----------------------------------+
```

The Q relative source byte address is formed by adding the byte address of Q and the Q relative address in B. Byte wrap-around is implemented, i.e., if the source address is above S, the 2**16 is subtracted from the source address. This is the mechanism for achieving Q negative byte addressing. The target address is used directly as given in F, E, D and C. The source byte pointed to is is moved to the target location, and the byte count is decremented. Bytes from the source area continue to be moved to the target area until the byte count reaches zero. The stack then is popped of all parameters. Data is protected by whatever means Native Mode has employed on the virtual address.

```
Opcode: 020443
Indicators: unaffected
Traps: MODE, STUN, BNDV
New for HP3000 Compatibility Mode on
Precision Architecture.
```

```
Begin
 TBA := (S-5,S-4,S-3,S-2); <<Target Byte Addr. in virtual space>
 SBA := (S-1) + Q; <<Source Byte Address>>
 IF SBA>S
  THEN SBA:=SBA-2**16;<<Q negative>>
 While (S) > 0
  do
   Begin
    (TBA) := (SBA);
    TBA := TBA + 1;
    SBA := SBA + 1;
    (S) := (S) - 1;
   End;
 S := S - 6;
End;
```