```
ENTITY modo0 IS
   PORT(dentro, fuera: INOUT bit_vector (7 DOWNTO 0);
        dir, rst, rd, wr: IN bit);
END modo0;

ARCHITECTURE modo0arch OF modo0 IS

   SIGNAL estado: bit_vector (7 DOWNTO 0);

BEGIN
   dentro <= fuera WHEN rd = '0' ELSE
             (OTHERS => 'Z');
   fuera <= estado WHEN dir = '0' ELSE
             (OTHERS => 'Z');
   PROCESS(wr, rst)
   BEGIN
     IF rst = '0' THEN
        estado <= (OTHERS => '0');
     ELSIF wr = '0' THEN
        estado <= dentro;
     ENDIF;
   END PROCESS;

END modo0arch;

---

ENTITY modo1 IS
   PORT(dentro, fuera: INOUT bit_vector (7 DOWNTO 0);
        dir, rst, rd, wr, stb, ack, inte: IN bit;
        intr, obf, ibf: OUT bit);
END modo1;

ARCHITECTURE modo1arch OF modo1 IS

   SIGNAL estado: bit_vector (7 DOWNTO 0);
   SIGNAL sintr, sobf, sibf;

BEGIN
   dentro <= estado WHEN rd = '0' ELSE
             (OTHERS => 'Z');
   fuera <= estado WHEN dir = '0' ELSE
             (OTHERS => 'Z');
   obf <= sobf;
   ibf <= sibf;
   intr <= sintr WHEN inte = '1' ELSE
           '0';
   PROCESS(wr, rd, rst)
   BEGIN
     IF rst = '0' THEN
        estado <= (OTHERS => '0');
     ELSIF wr = '0' and dir = '0' THEN
        estado <= dentro;
     ELSIF stb = '0' and dir = '1' THEN
        estado <= fuera;
     ENDIF;
   END PROCESS;

   PROCESS(stb, rd, rst)
```

```
   BEGIN
     IF rst = '0' THEN
        sibf <= '0';
     ELSIF stb'event AND stb = '0' THEN
        sibf <= '1';
     ELSIF rd'event AND rd = '1' THEN
        sibf <= '0';
     ENDIF;
   END PROCESS;

   PROCESS(ack, wr, rst)
   BEGIN
     IF rst = '0' THEN
        sobf <= '1';
     ELSIF ack'event AND ack = '0' THEN
        sobf <= '1';
     ELSIF wr'event AND wr = '1' THEN
        sobf <= '0';
     ENDIF;
   END PROCESS;

   PROCESS(stb, rd, ack, wr, rst)
   BEGIN
     IF rst = '0' THEN
        sintr <= '0';
     ELSIF dir = '0' THEN
       IF ack'event AND ack = '1' THEN
          sintr <= '1';
       ELSIF wr'event AND wr = '0' THEN
          sintr <= '0';
       ENDIF;
     ELSIF dir = '1' THEN
       IF stb'event AND stb = '1' THEN
          sintr <= '1';
       ELSIF rd'event AND rd = '0' THEN
          sintr <= '0';
       ENDIF;
     ENDIF;
   END PROCESS;
END modo1arch;

---

ENTITY modo2 IS
   PORT(dentro, fuera: INOUT bit_vector (7 DOWNTO 0);
        rst, rd, wr, stb, ack, intei, inteo: IN bit;
        intr, obf, ibf: OUT bit);
END modo2;

ARCHITECTURE modo2arch OF modo2 IS

   SIGNAL estado: bit_vector (7 DOWNTO 0);
   SIGNAL sintr, sobf, sibf;

BEGIN
   dentro <= estado WHEN rd = '0' ELSE
             (OTHERS => 'Z');
   fuera <= estado WHEN ack = '0' ELSE
             (OTHERS => 'Z');
   obf <= sobf;
```

```
   ibf <= sibf;
   intr <= sintr;

   PROCESS(wr, rd, rst)
   BEGIN
     IF rst = '0' THEN
        estado <= (OTHERS => '0');
     ELSIF wr = '0' THEN
        estado <= dentro;
     ELSIF stb = '0' THEN
        estado <= fuera;
     ENDIF;
   END PROCESS;

   PROCESS(stb, rd, rst)
   BEGIN
     IF rst = '0' THEN
        sibf <= '0';
     ELSIF stb'event AND stb = '0' THEN
        sibf <= '1';
     ELSIF rd'event AND rd = '1' THEN
        sibf <= '0';
     ENDIF;
   END PROCESS;

   PROCESS(ack, wr, rst)
   BEGIN
     IF rst = '0' THEN
        sobf <= '1';
     ELSIF ack'event AND ack = '0' THEN
        sobf <= '1';
     ELSIF wr'event AND wr = '1' THEN
        sobf <= '0';
     ENDIF;
   END PROCESS;

   PROCESS(stb, rd, ack, wr, rst)
   BEGIN
     IF rst = '0' THEN
        sintr <= '0';
     ELSIF ack'event AND ack = '1' THEN
          sintr <= inteo;
     ELSIF wr'event AND wr = '0' THEN
          sintr <= '0';
     ELSIF stb'event AND stb = '1' THEN
          sintr <= intei;
     ELSIF rd'event AND rd = '0' THEN
          sintr <= '0';
     ENDIF;
   END PROCESS;
END modo2arch;

---

ENTITY modos IS
   PORT(dentro, fuera: INOUT bit_vector (7 DOWNTO 0);
        modo: IN bit_vector (1 DOWNTO 0);
        dir, rst, rd, wr, stb, ack, intei, inteo: IN bit;
        intr, obf, ibf: OUT bit);
END modos;
```

```
ARCHITECTURE modosarch OF modos IS

   SIGNAL estado: bit_vector (7 DOWNTO 0);
   SIGNAL sintr, sobf, sibf;

BEGIN
   dentro <= fuera WHEN rd = '0' AND modo = "01" ELSE
             estado WHEN rd = '0' ELSE
             (OTHERS => 'Z');
   fuera <= estado WHEN ack = '0' AND modo = "10" ELSE
            estado WHEN dir = '0' AND modo /= "10" ELSE
            (OTHERS => 'Z');
   obf <= sobf WHEN modo /= "01" ELSE
          UNAFFECTED;
   ibf <= sibf WHEN modo /= "01" ELSE
          UNAFFECTED;
   intr <= sintr WHEN modo /= "01" ELSE
          UNAFFECTED;

   PROCESS(wr, rd, rst)
   BEGIN
     IF rst = '0' THEN
        estado <= (OTHERS => '0');
     ELSIF wr = '0' THEN
        estado <= dentro;
     ELSIF stb = '0' THEN
        estado <= fuera;
     ENDIF;
   END PROCESS;

   PROCESS(stb, rd, rst)
   BEGIN
     IF rst = '0' THEN
        sibf <= '0';
     ELSIF stb'event AND stb = '0' THEN
        sibf <= '1';
     ELSIF rd'event AND rd = '1' THEN
        sibf <= '0';
     ENDIF;
   END PROCESS;

   PROCESS(ack, wr, rst)
   BEGIN
     IF rst = '0' THEN
        sobf <= '1';
     ELSIF ack'event AND ack = '0' THEN
        sobf <= '1';
     ELSIF wr'event AND wr = '1' THEN
        sobf <= '0';
     ENDIF;
   END PROCESS;

   PROCESS(stb, rd, ack, wr, rst)
   BEGIN
     IF rst = '0' THEN
        sintr <= '0';
     ELSIF modo = "10" THEN
       IF ack'event AND ack = '1' THEN
          sintr <= inteo;
```

```
       ELSIF wr'event AND wr = '0' THEN
          sintr <= '0';
       ELSIF stb'event AND stb = '1' THEN
          sintr <= intei;
       ELSIF rd'event AND rd = '0' THEN
          sintr <= '0';
       ENDIF;
     ELSIF modo = "01" THEN
       IF dir = '0' THEN
         IF ack'event AND ack = '1' THEN
            sintr <= inteo;
         ELSIF wr'event AND wr = '0' THEN
            sintr <= '0';
         ENDIF;
       ELSIF dir = '1' THEN
         IF stb'event AND stb = '1' THEN
            sintr <= inteo;
         ELSIF rd'event AND rd = '0' THEN
            sintr <= '0';
         ENDIF;
       ENDIF;
     ENDIF;
   END PROCESS;
END modosarch;
```