```
1     --  Copyright 2001 Michael Kellett
2     --  mk@mkesc.co.uk
3     --   Behavioural model of Burr Brown ADS 8320 A-D converter
4     --   If CS goes low when the clock is high the model behaves like the chip
5     --   If CS goes low when the clock is low the chip would be OK so long as 60nS elapses before the clock goes high
6     --  The model does not need the setup time and will start sampling on the first clock edge after CS falls
7
8
9     library IEEE;
10    use IEEE.std_logic_1164.all;
11
12    entity bbads8320_bhv is
13     port (
14         DCLOCK: in STD_LOGIC;
15         CS: in STD_LOGIC;
16         DATA: out STD_LOGIC;
17         AN_DATA: in STD_LOGIC_VECTOR (15 downto 0)
18     );
19    end bbads8320_bhv;
20
21
22    architecture BHV of bbads8320_bhv is
23
24    --   declare state machine type with the state names
25    type state_type is (POWER_DOWN, SAMPLING, CONVERTING, REPEATING, TRISTATED);
26    --   declare signals for internal variables
27    --   declare analogue data
28
29    begin
30
31        STATE_MACHINE:      -- optional label
32        process (CS, DCLOCK)
33        -- declarations
34        variable STATE : state_type;
35     variable FALLING_EDGES : integer range 0 to 32;
36     variable RISING_EDGES : integer range 0 to 32;
37     variable CURRENT_BIT : integer range -1 to 16 := 15;
38     begin
39         if CS /= '0' then           -- if the CS is not low the state machine resets and output tri-states
40             STATE := POWER_DOWN;
41             DATA <= 'U';
42             FALLING_EDGES := 0;
43             RISING_EDGES := 0;
44             CURRENT_BIT := 15;
45         else
46             if CS'event then
47                 --and DCLOCK = '1' then -- change to sampling only if DCLOCK high (not accurate for chip but safe )
48                 STATE := SAMPLING;              -- if CS falling edge recognised can go straight to sampling state
49             else
50                 if DCLOCK'event then    -- only fire the state machine on clock edges
51                     case STATE is
52                         when POWER_DOWN =>
53                         if DCLOCK  = '1' then                   -- only move out of power down on a rising edge
54                             STATE := SAMPLING;
55                         end if;
```

```
56                      when SAMPLING =>
57                      if DCLOCK = '1' and RISING_EDGES < 5 then
58                          RISING_EDGES := RISING_EDGES + 1;   -- count up to 5 rising edges in sampling state
59                          if RISING_EDGES = 5 then
60                              STATE := CONVERTING;            -- and jump states when we get them
61                          end if;
62                      end if;
63                      when CONVERTING =>
64                      if DCLOCK = '0' then                    -- look for falling clock in converting mode
65                          DATA <= AN_DATA(CURRENT_BIT);       -- put the data out on the data pin
66                          CURRENT_BIT := CURRENT_BIT - 1;     -- count data bits
67                          if CURRENT_BIT = -1 then
68                              STATE := REPEATING;             -- change state when they are all done
69                              CURRENT_BIT := 0;               -- point to the LS bit for next state
70                          end if;
71                      end if;
72                      when REPEATING =>
73                      if DCLOCK = '0' then                    -- look for falling clock in repeating mode
74                          DATA <= AN_DATA(CURRENT_BIT);       -- put the data out on the data pin in reverse order
75                          CURRENT_BIT := CURRENT_BIT + 1;     -- count data bits
76                          if CURRENT_BIT = 16 then
77                              STATE := TRISTATED;             -- change state when they are all done
78                          end if;
79                      end if;
80                      when TRISTATED =>
81                      if DCLOCK = '0' then                    -- tri-state on each falling clock when finished
82                          DATA <= 'U';
83                      end if;
84                  end case;
85              end if; -- only fire the state machine on clock edges
86          end if;     -- if the CS has just gone low wait 20nS before looking for next rising clock
87      end if;         -- if the CS is not low the state machine resets and output tri-states
88
89      end process;
90
91  end BHV;
92
```