Chapter 5 – Sub module testing and verification

Due to the lack of readily available programmer's kit for the purpose of flashing the FPGA, out of all the programmable modules available only module annotated as **5(a)** - Downward movement FM head up to pres-stop position (which was a sub condition of the interlock dealing with the Support structure assembly). Before proceeding on to the further sections, it would be worthwhile to note why this module was selected for testing amongst all the available modules.

- Due to a large number of inputs available in the module leading to an exponential number of input test cases.
- The module involved both active high and active low inputs.
- The module used predefined constants stored in non-volatile memory.
- The hardware implementation proved to be complex enough to be tested on a test jig so as to utilize considerable number of I/Os available on the kit.

5.1 SOFTWARE CODE

Following is the VHDL code for sub module 5(a)

```
--------------------------------------------------------------------------------
-- Organisation: BARC
--
-- File: code_5_a.vhd
-- File history:
--      1: 17/06/2019: first draft
--
--
-- Description:
--
-- AHWR interlock module 5(a)
--
-- Targeted device: Family::ProASIC3L Die::A3P600L Package::484 FBGA
-- Author: Pronoy Mandal
--
--------------------------------------------------------------------------------


library IEEE;
```

```
use IEEE.std_logic_1164.all;

entity code_5_a is
port (
    --<port_name> : <direction> <type>;

    --inputs bifurcated by conditions

    --sub condition 1
      DI13,DI14,DI15,DI16: IN  std_logic;
    --sub condition 2
    -- DI15 defined earlier
     DI1,DI19 : IN  std_logic;
    --sub condition 3
     --omitted as of now
    --sub condition 4
     DI86,DI87,DI88,DI89,DI90,DI91,DI92,DI93 : IN  std_logic;
    --sub condition 5
     DI94,DI98 : IN std_logic;
    --sub condition 6
     --DI94 and DI98 defined earlier
    DI95,DI99 : IN std_logic;
    --sub condition 7
    AI57 : IN  std_logic;
    --sub condition 8
    --omitted as of now
    --sub condition 9
sense_finger_error : IN std_logic;
--digital output
    Y5_a : OUT std_logic);
end code_5_a;
```

```
architecture architecture_code_5_a of code_5_a is


-- signal, component etc. declarations


-- all intermediate outputs

signal
intermed1,intermed2,intermed4,intermed5,intermed6,intermed7,intermed9,AO5,AO6 :
std_logic;


--fixed constants to be stored in non volatile memory

constant PVNULL : std_logic:='0';

begin


   --internal logic circuitry

   intermed1 <= (DI13 nand DI14) and (DI15 and DI16);

   intermed2 <= DI1 or DI15 or DI19;

   intermed4 <= DI86 and DI87 and DI88 and DI89 and DI90 and DI91 and DI92 and
DI93;

   intermed5 <= DI94 and DI98;

   intermed6 <= DI94 and DI95 and DI98 and DI99;

   intermed7 <= AI57;

   intermed9 <= sense_finger_error;

   AO5 <= intermed1 and intermed2 and intermed4 and intermed5 and intermed6 and
intermed7 and not intermed9;

   AO6 <= AO5;


   --assign final output

   Y5_a <= (AO5 and not PVNULL) and (AO6 and not PVNULL);


end architecture_code_5_a;
```

5.2 RESULTS

The reader may refer to **section 3.1.5.a** for the full set of conditions encountered while developing the interlock and **Figure 3.1.5.a** for the hardwired implementation of the above code.

The module was completely tested on the M&SL test jig and all outputs were verified. While doing so a typical feature of Libero® SoC software suite was revealed – auto removal of logical redundancies (discussed more in **Chapter 6**). While editing various pin attributes associated with module 5(a), the editing table automatically managed to redact the useless input pads.

The waveforms can be referred back to **section 3.2.5.a**.