P.K. Gaikwad, International Journal of Computer Science and Mobile Applications,
Vol.1 Issue. 4, October- 2013, pg. 1-6 **ISSN: 2321-8363**

PROTOTYPE DEVELOPMENT OF FPGA BASED PS/2 MOUSE CONTROLLED PCB DRILL MACHINE

P. K. Gaikwad[1]

[1] *Department of Electronics,*
Willingdon College, Sangli, (M.S.), INDIA,
pawangaikwad2003@yahoo.co.in

Abstract

The research work presents a prototype development of an electromechanical setup to perform the mouse controlled Printed Circuit Board (PCB) drilling. The Soft Intellectual Proprietary (IP) Core was developed to drive two stepper motors; simultaneously, to reach the drill point. The Personal System, PS/2 mouse was used to generate serial data and clock signals. The serial data packets emerging from PS/2 mouse indicates the exact location of the mouse device. The mouse was moving on a white paper; with PCB pads printed thereon. The horizontal and vertical location data was sent to the Field Programmable Gate Array (FPGA); driving two stepper motors. The shafts of motors were coupled with separate conveyer belts. The drill-assembly consisting of a 12 V Direct Current (DC) motor was fixed on a belt. This entire arrangement was placed on one more conveyer belt; coupled to the shaft of an additional stepper motor. The motors were driven by FPGA logic core in such a way that, the drill machine was pulled towards the drill point (pad) on the actual PCB.

***Keywords**:* Conveyer Belt; FPGA; PCB Drilling; PS/2 Mouse; Stepper Motor

1. Introduction

As given by (Ngoc Quy Le, Jae Wook Jeon, 2007), in motion control systems, stepper motors are widely used and the stepper motor driver is mainly implemented using a DSP (Digital Signal Processor). The DSP based driver's exhibit good performance; they are reliable, precise, and responsive. However, when mass production is needed, FPGA (Field-programmable Gate Array) based drivers are preferred since the FPGA based design is compatible with conversation to an ASIC (Application Specific Integrated Circuit).

The research work reported by (Daniel Piši, Ilona Kalová, 2011) deals with implementation of linear speed controller for stepper motors within an FPGA. It shows how to deal with computational complexity and balance performance against resources requirements using Xilinx MicroBlaze soft microcontroller. The designed peripheral is able to generate linear speed profiles including micro-stepping with no microcontroller load involved, while performing computationally complex initialization the software way.

The research paper by (Arvind Kumar, M. Valarmathi, 2013) describes a stepper motor controller designed using Very-High-Speed Integrated Circuits Hardware Description Language (VHDL) and implemented in FPGA. The system is capable of controlling the stepper motor in terms of step-angle at 0.915˚. In addition to this it makes it rotatable at different speeds, displacement clockwise and anticlockwise direction wirelessly.

The CNC PCB drilling machine reported by (Zulkifli Tahir, Nur Azman Abu, Shahrin Sahib, Nanna Suryana Herman, 2010) was equipped with three dimensional movements and considered to produce good precision accuracy for a competitive development cost. The design approach gives promising fast near optimal solution in total tour travelling time and total distance on the PCB with the error average less than 10% compare to best

© 2013, IJCSMA All Rights Reserved, www.ijcsma.com

known solution. However, it requires refinements and improvements need to be done in the machine concept design to get better solution and functionality.

The present paper deals with position controlling and monitoring of a drilling tool; moving in horizontal and vertical axes. The moving drill-object was assembled with a 12V Direct Current (DC) motor and its shaft was coupled with a drill-bit to drill the pads located on the Printed Circuit Board (PCB). The PCB tools like Protel/Altium, Eagle, KICAD, gEDA and ZenitPCB etc. provides the output print of various layers of a PCB. For example, Protel/Altium generates layout prints of Top Overlay, Top and Bottom Tracks, Pads etc. In this research work the PCB Pads-printed-page was placed beneath the Personal System, PS/2 mouse. The mouse had an attached optical sensor; detecting the PCB Pads on the page. The dark shaded pad absorbs the transmitted light from the photo-transmitter placed in the same assembly. In this way by moving the mouse device, the location of PCB pad was detected. The Soft Intellectual Proprietary (IP) Core was hardwired in Xilinx FPGA to detect the location of mouse on the paper; where PCB pads were printed. The FPGA keeps monitoring the status of switches (left-click, right-click, and scroll-switch) and take necessary actions to move the drill-object to the drill-pads on actual PCB. The movement of both stepper motors get stopped as soon as the pad location is found, and the DC motor switches-on to drill the pad addressed.

2. Source Code for FPGA based PS/2 Mouse Controlled Two Axes Drill Machine

The Mouse reference component given in (Support Document, 2008) has been deployed to design soft IP core necessary in this research work. It has three internal modules, 'p2interface', resolution_mouse_informer', and 'mouse_controller.' It detects the horizontal and vertical position of the mouse device. This module has two wire connections required for serial communication: 'ps2_clk' and 'ps2_data'. Importantly, it has a provision of detecting the position of mouse and provides the 10-bit vector signals; pertaining to its horizontal and vertical location. It also indicates the mouse buttons pressed by three different signals, i.e. the Right and Left click as well as the Scroll (middle) switch.

The mouse reference component given in (Support Document, 2008) has two output vectors; indicating the location of the mouse in a given frame. A VHDL module designed in this work has two buffers to store the horizontal and vertical position-coordinates. The buffer-variables associated with a location are down-counted every time when the stepper motors move the drill-tool with one step ahead, i.e. until the drill position is achieved. The zero values in both these variables ensure the exact location obtained by the drill-tool and the DC motor relay was energized to start drilling the pad on PCB. The drilling process execute when the middle (scroll) button on the mouse was pressed. The mouse reference component has another output signal; which indicates if any new movement of the mouse occurs. The VHDL entity, port mapped with the mouse reference module, gets new movement alert signal and enters into the wait state to obtain new and exact coordinates of the mouse-position.

The following lines of VHDL code show partial part of the architecture. The port map operations between the mouse reference component given in (Support Document, 2008) and another module designed in this work is given below.

```
Mouse : MouseRefComp
        port map(     CLK           => CLK_T,
                      RESOLUTION     => RESOLUTION_T,
                      RST           => RST_T,
                      SWITCH        => SWITCH_T ,
                      LEFT          => LEFT_T ,
                      MIDDLE        => MIDDLE_T ,
                      NEW_EVENT     => NEW_EVENT_T ,
                      RIGHT         => RIGHT_T ,
                      XPOS           => XPOS_int ,
                      YPOS           => YPOS_int ,
                      PS2_CLK       => PS2_CLK_T ,
                      PS2_DATA      => PS2_DATA_T );
```

```
XY_Plot: drill
port map(
            CLK             => CLK_T,
            RESET           => RST_T,
            XPOS_Count      => XPOS_int ,
            YPOS_Count      => YPOS_int,
            OUTM1           => MOTOR_X,
            OUTM2           => MOTOR_Y,
            DRILL           => MIDDLE_T,
            TAKE_POS        => LEFT_T,
            DRILL_OUT       => DRILL_OUT_T
  );
```

The VHDL port mapping shows that, there are two entities named as ‘MouseRefComp’ and ‘drill’, their instance names are ‘Mouse’ and ‘XY_Plot’ respectively. The former entity has clock signal ‘CLK’ connected with the top level module’s clock signal, named as ‘CLK_T’. Similarly, rest of the port map connections given in the code are self explanatory. The output signals named as ‘XPOS’ and ‘YPOS’ are connected with the internal signals called as ‘XPOS_int’ and ‘YPOS_int’. In the port map operation for another module, ‘drill’, these signals are connected with its input signals called as ‘XPOS_Count’ and ‘YPOS_Count’. When input signal ‘TAKE_POS’ asserts to high, the entity ‘drill’ collects the data from internal signals associated with the horizontal and vertical position of the mouse.

As given in (Support Document, 2008), the X and Y sign bit is the most significant bit of the X and Y movement value. Thus, the range of movement is -256 to 255. The delta movement from the mouse’s last position is sent for either axis. The mouse does not keep track of the absolute position of the mouse. The controller uses the delta movements received to keep track of the absolute position. The mouse uses the lower-left corner for axes origin, meaning the mouse reports a positive displacement on X when moved to the right and negative (X sign bit is ‘1’) when moved to the left. When the mouse is moved upward, the Y displacement is positive, and when moved downward it is negative.

Considering the sign of the displacement, the research module designed here uses the sign bit to control the direction of the stepper motors; moving in horizontal and vertical axes. The following lines of VHDL entity designed in this research, called as ‘drill’ show the reset state of the Finite State Machine (FSM) as well as the first state transition.

```
if RESET='1' then                -- active HIGH RESET
      state<=S0;
            XPOS_Count_int<=0;
            OUTM1<= "0101";       -- 5
       elsif TAKE_POS='1' then
            XPOS_Count_int<= conv_integer(XPOS_Count(7 downto 0));
      elsif(CLK'event and CLK='1') then

      case state is
      when S0 => if (XPOS_Count(8)='0') and (XPOS_Count_int>0) then    --Reverse direction
                  state<=s3;
                  OUTM1<= "1001"; -- 9
                  XPOS_Count_int<= XPOS_Count_int-1;
                elsif (XPOS_Count(8)='1')  and (XPOS_Count_int>0) then --Forward direction
                  state<=s1;
                  OUTM1<= "0110";-- 6
                  XPOS_Count_int<= XPOS_Count_int-1;
               end if;
```

The partial code of the entity ‘drill’ shows that, when FSM was reset, it enters into the state ‘S0’, and initializes the counter signal ‘XPOS_Count_int’ to zero. The 4-bit signal, ‘OUTM1’ was given to the motor driver card that enhances the current and provide to the stepper motor’s four coils. This motor signal was initialized with

IJCSMA
P.K. Gaikwad, International Journal of Computer Science and Mobile Applications,
Vol.1 Issue. 4, October- 2013, pg. 1-6 **ISSN: 2321-8363**

value “0101”; upon the reset state. The mouse reference component entity ‘MouseRefComp’ has output signal called as ‘LEFT’, which is further connected with the ‘drill’ entity’s input signal, named as ‘TAKE_POS’. Therefore, signal ‘TAKE_POS’ gets enabled when the left-click button of the mouse is held pressed. In this FSM, this signal (‘TAKE_POS’) plays a vital role; which accepts horizontal position of the mouse using VHDL line: *XPOS_Count_int<= conv_integer(XPOS_Count(7 downto 0));*

Upon releasing the left-click, the FSM starts transition from reset state to the new states at every positive transition of the input clock. At every state transition, new signal to the motor coils was provided to take one step ahead. Simultaneous to this, the counter value of horizontal position get decremented by VHDL line: *XPOS_Count_int<= XPOS_Count_int-1;*

The direction of the motor was also controlled by sensing the sing bit received from mouse reference component. This sign bit is available in a slice of signal, called as ‘XPOS_Count(8)’ and every time the count value is also verified for being its non-zero and then take one step ahead by the stepper motor. The sign bit decides to take forward or reverse direction of the motor. To perform this, an *if(...)* conditional statement in VHDL was constructed as:

if (XPOS_Count(8)='0') and (XPOS_Count_int>0) then

There was another process running concurrently; to perform the same task of accepting the location coordinates. But in this process, the position of vertical location was stored in another buffer-variable. The stepper motor used for vertical movement was moved.

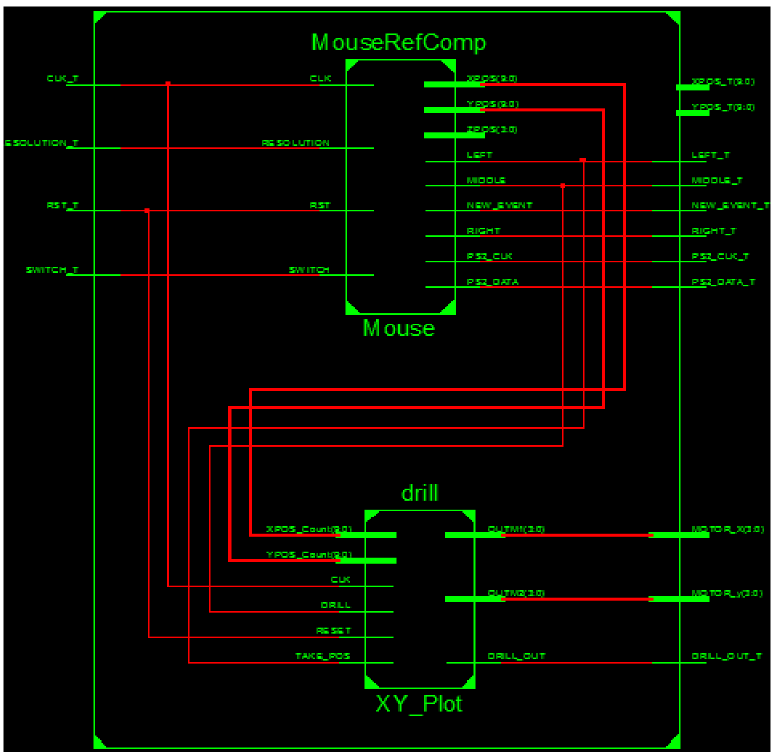

Figure 1: Internal View of the RTL Synthesis showing Two Entities: ‘MouseRefComp’ and ‘drill’ connected together to form Top Level Entity

3. Synthesis Results of the Top Level VHDL Entity

The Figure 1 illustrates the synthesis results obtained by executing the process of Synthesis; as a design flow being used in Xilinx Integrated System Environment (ISE), version 14.6. The explanation of input and output lines of the top level entity is given in the port map operations in the partial VHDL code of the entity. The Figure 1 illustrates the detailed view of the interconnections of two VHDL modules; developed by using

© 2013, IJCSMA All Rights Reserved, www.ijcsma.com

structural modelling style of the VHDL architecture. It shows that, there are two entities named as 'MouseRefComp' and 'drill' and are connected together to form a top level module: 'MouseDriverTOP'. Following to the process of Synthesis, the project went through the 'Implementation' process as well. Before which, there was a user constraint format file (.ucf) written to interconnect the top level I/O lines with actual I/O pads of the Xilinx FPGA Spartan 3E device. The FPGA was embedded on the development board Nexys2 designed and manufactured by Digilent Inc.

4. Mechanical Setup of the Mouse and FPGA Controlled Drill Machine

The PS/2 mouse was attached with an FPGA board Nexys2 embedded with Spartan 3E device. The FPGA was hardwired with the Soft Intellectual Proprietary (IP) Core; which was able to detect the position of mouse. The Figure 2 shows two stepper motors M1 and M2 coupled with separate conveyer belts. The angular movements of the stepper motors shaft were converted into the linear motion by means of conveyer belts for horizontal and vertical displacements. The vertical movable assembly consists of a DC motor; coupled with drill-bit. The motor M2 shaft was to pull the conveyer belt and make the DC motor assembly movable in vertical axis. The entire and vertically moving assembly was fixed on another conveyer belt; which was stirring horizontally by the stepper motor M1. When Personal System, PS/2 mouse moves and the photo detector sensor detects the position of the drilling pad on the printed paper, it requires a left-click on the mouse. Thus, the mouse position was communicated to the FPGA core, and provides the X, Y position coordinates to it. The stepper motors M1 and M2 gets driven by FPGA-configured logic core and further the assembly of DC motor moves towards the actual PCB pad location. The FPGA gets acknowledgement of reaching such a location and it turns on the DC motor to drill the selected PCB pad.

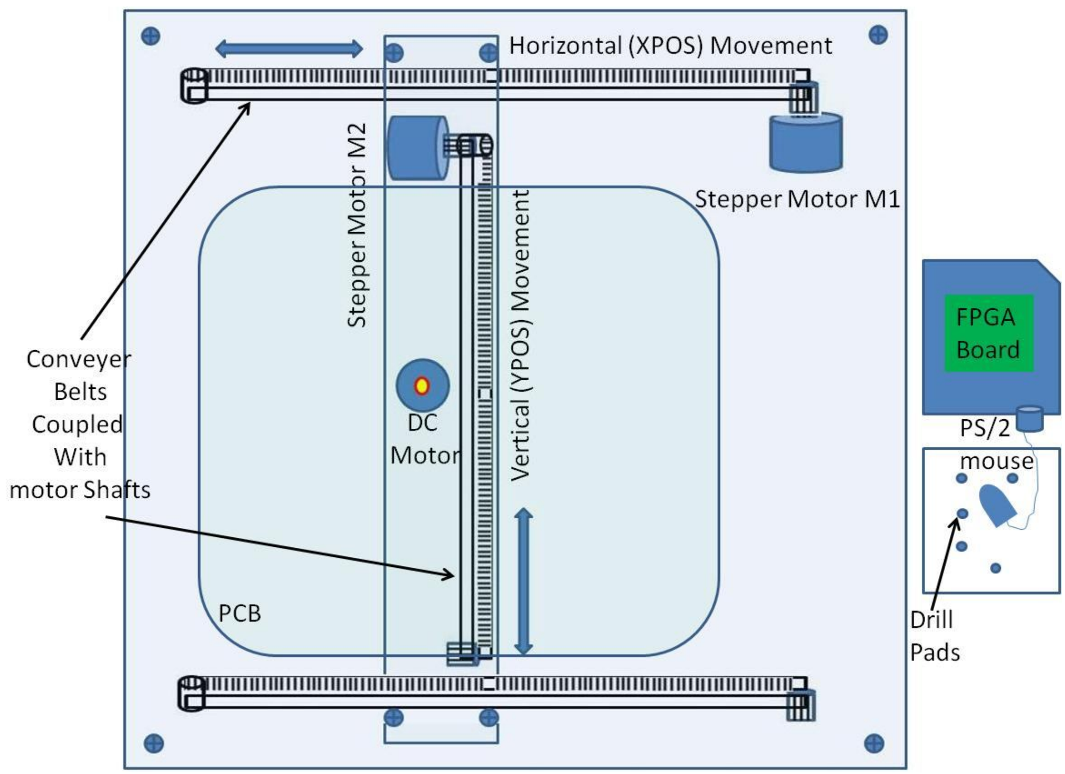

Figure 2: Prototype of Mechanical Setup of Mouse Controlled and FPGA based PCB Drill Machine

4. Conclusion

The research work technologically advances the drilling of a PCB pads. The potential of Hardware Description Language (HDL) i.e. performing multitasking and executing number of processes concurrently has been used in this work by running two stepper motors simultaneously. The motors are moved in two the axes to reach the proper drill position; upon receiving the necessary commands from PS/2 mouse and FPGA.

References:

[1] Arvind Kumar, M. Valarmathi, 2013, "High Precision Stepper Motor Controller Implementation on FPGA with GUI on LabVIEW", *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, Vol. 2, Issue 4, pp. 1197-1202,

[2] Daniel Piši, Ilona Kalová, 2011, "GENERATING STEPPER MOTOR SPEED PROFILES USING FPGA", retrieved from http://www.feec.vutbr.cz/EEICT/2011/sbornik/03-Doktorske%20projekty/03-Kybernetika%20a%20automatizace/10-xpisid00.pdf

[3] Ngoc Quy Le, Jae Wook Jeon, 2007 "An Open-loop Stepper Motor Driver Based on FPGA", *International Conference on Control, Automation and Systems 2007 Oct. 17-20, in COEX, Seoul, Korea*

[4] Support Document, 2008, "Mouse Reference Component", *Digilent Romania, Digilent Inc*., pp. 1-7, Doc: DSD-0000234

[5] Zulkifli Tahir, Nur Azman Abu, Shahrin Sahib, Nanna Suryana Herman, 2010, "CNC PCB Drilling Machine using Novel Natural Approach to Euclidean TSP", retrieved from http://www.google.co.in/url?sa=t&rct=j&q=&esrc=s&frm=1&source=web&cd=1&cad=rja&ved=0CCkQFjAA&url=http%3A%2F%2Fwww.researchgate.net%2Fpublication%2F224172606_CNC_PCB_drilling_machine_using_Novel_Natural_Approach_to_Euclidean_TSP%2Ffile%2F9fcfd50bc2b5c506d8.pdf&ei=xWJOUsSDE8q3kAW7nYCgBA&usg=AFQjCNF857gbGRUx9IhOQaqVXI1O1i9bkA&bvm=bv.53537100,d.dGI

[1]**P.K. Gaikwad –** Dr. Pawan K. Gaikwad, born in INDIA, on August 29th, 1976, is M.Sc.-Electronic Science, from Department of Electronic Science, University of Pune, Pune(1999), Ph.D. in Electronics(2010), Diploma in VLSI Design, Pune(2000). He is a Research Guide in Electronics for the faculty of Science in Shivaji University, Kolhapur (Maharashtra), INDIA.