# Project Name: new\_pid

## Software Used:

Xilinx ISE Version 13.2

## Board:

Nexys 2

## Procedure:

1. Open the Xilinx ISE 13.4 application
2. Go to:  
   File -> New Project
3. Fill in the following details to create a project for Nexys 2 board compatibility:
   * Family – Spartan3E
   * Device – XC3S500E
   * Package – FG320
   * Speed - -4
4. Add the following files to this project by right clicking the  icon and selecting “Add a copy of the source” option-
   * Pid\_top.vhd
   * Quad\_decoder.vhd
   * Display.vhd
   * Prescalar.vhd
   * Pid\_control.vhd
   * Pid\_bounder.vhd
   * Pwm\_gen.vhd
   * Pres\_clk.vhd
   * Brk\_add.vhd
   * Add\_3.vhd
   * Bintobcd.vhd
   * Accum\_err\_sum\_16bits.xco
   * Bin\_cntr\_19.xco
   * Mult\_16.xco
   * Add\_32.xco
   * Subtractor\_15bit\_error.xco
   * Adder\_15\_bit.xco
   * Counter\_14bit\_up.xco
   * Counter\_16\_up\_sclr.xco
5. Here the last few \*.xco files are Xilinx generated IP core files. Following are the steps to form a new IP core file:
   * 1. Go to: Tools->Core Generator
     2. In the “Xilinx Core Generator” window, select: File->New Project
     3. Save the IP Core project in the main project folder for easy reference.
     4. You can either select the required IP core by browsing through the “View by functions” section or searching for it in the “View by name” section. For example, DCM IP can be found under “FPGA Features and Design -> Clocking -> Spartan3E -> Single\_DCM”
     5. Double click on the required IP and enter the requirements. Click on “Generate” button to add this IP core to your IP project in order to be used in your Xilinx ISE VHDL/Verilog Project.

# Code Description

This VHDL code implements the PID generation and control part via the mathematical procedure described in the paper “FPGA Implementation of Closed-Loop Control System for Small-Scale Robot”.



The above equations show the required arithmetic operations which find out the error of the current speed from required speed, e(n), and hence compute the appropriate feedback signal, u(n).

The given VHDL code takes as input the required speed, in the form of number of required ‘ticks’ of the quadrature encoder, and using the signals from encoders attached to the robot-wheel base, computes the current speed and displays it on the 7 segment for the user.

The VHDL file “pid\_control.vhd” takes care of this computation part and provides the new ‘control signal’ for generation of PWM that controls the omni-wheels.