

PIF EASY PROGRAMMING

INSTRUCTIONS FOR USE

(For programming iBot – “Introduction to Engineering” Course)

1. Introduction

PIF Easy Programming is created by the members of the FEEE Student Research Club (Pay-It-Forward Club) and developed based on open source softwares (Eclipse RCP, IBM GEF and Texas Instruments CCS toolchain).

PIF Easy Programming is intended to be used in the “Introduction to Engineering” Course for 1st year student, specifically to program the iBot robot.

Since 1st year student hasn't taken any specialized course, and hasn't been introduced to programming languages (C for instance...), this software is created to present simple, explicit and easy for learning subjects on Electrical Engineering for 1st year student but still addressing fundamental concepts and overall principles.

Software overview:

- + The PIF Easy Programming software is used to write programs for Texas Instruments's MSP430G2553 MCU (Microcontroller).
- + Friendly drag-and-drop UI (User Interface).
- + Students “program” by connecting blocks with each other (similar to Matlab/Simulink, Labview).
- + Program is constructed by connecting blocks → students can have an overall look at the system, understand system's working procedure without diving deep into MCU's architecture.
- + *Easy to compile and download program to KIT:* Simply press the RUN button on the tool bar, the program (consisted of blocks) will be compiled into C code and automatically download to Robot KIT without linking to any other software (i.e. CCS – compiler for the MCU in iBot).
- + The software will create a C source file (main.c) for students to check the code when needed to.

2. Installation

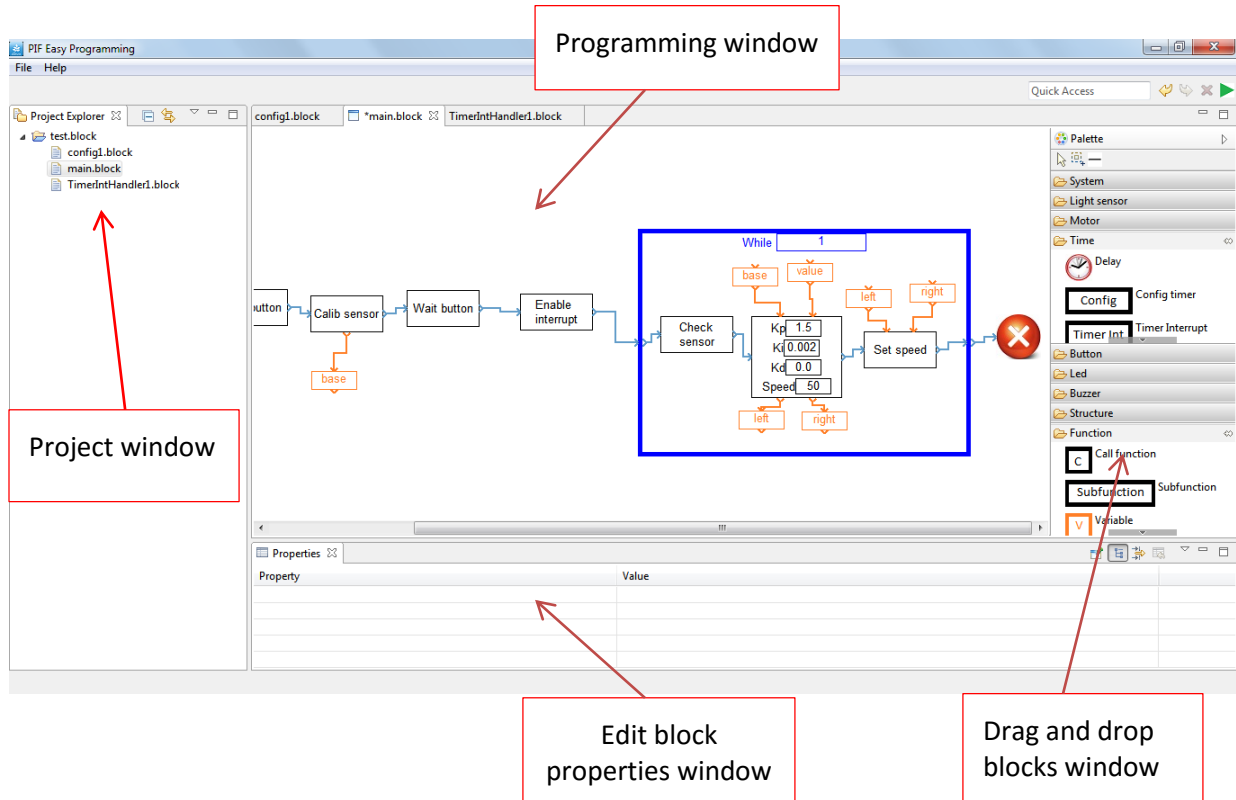
2.1. System Requirements

- Win XP, Win 7, Win 8 32/64 bit.

2.2. Installation Guide

- No installation step is required.

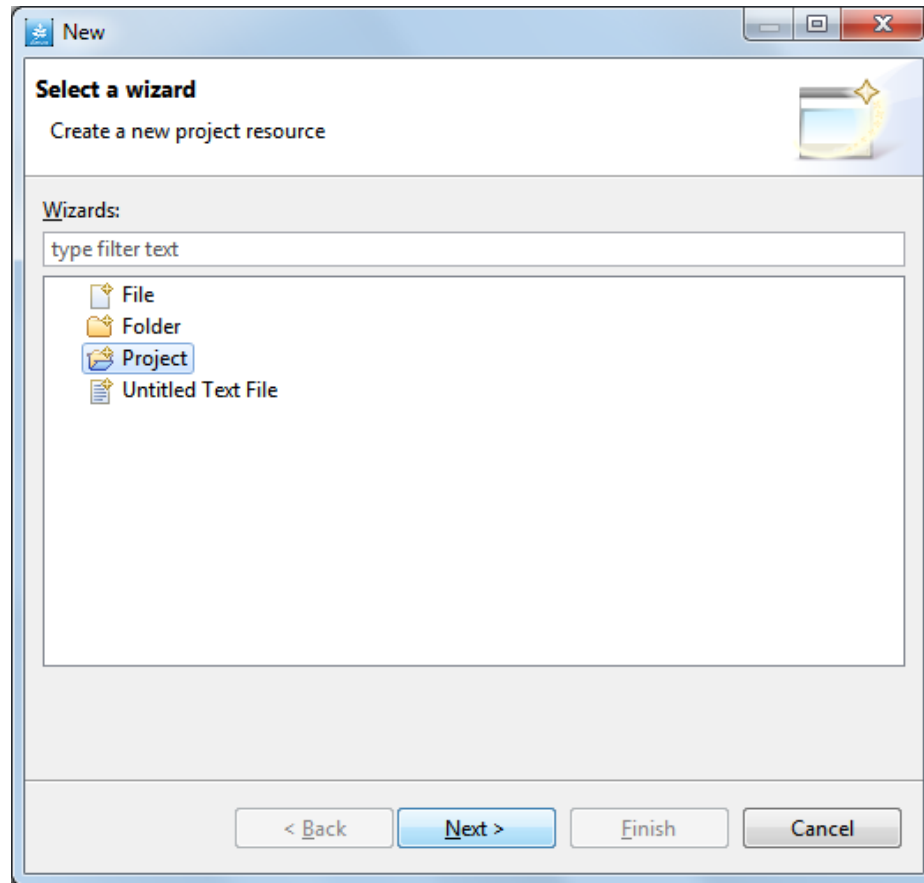
3. Software's Main Window



4. Basic Working Steps

4.1. Create new projects

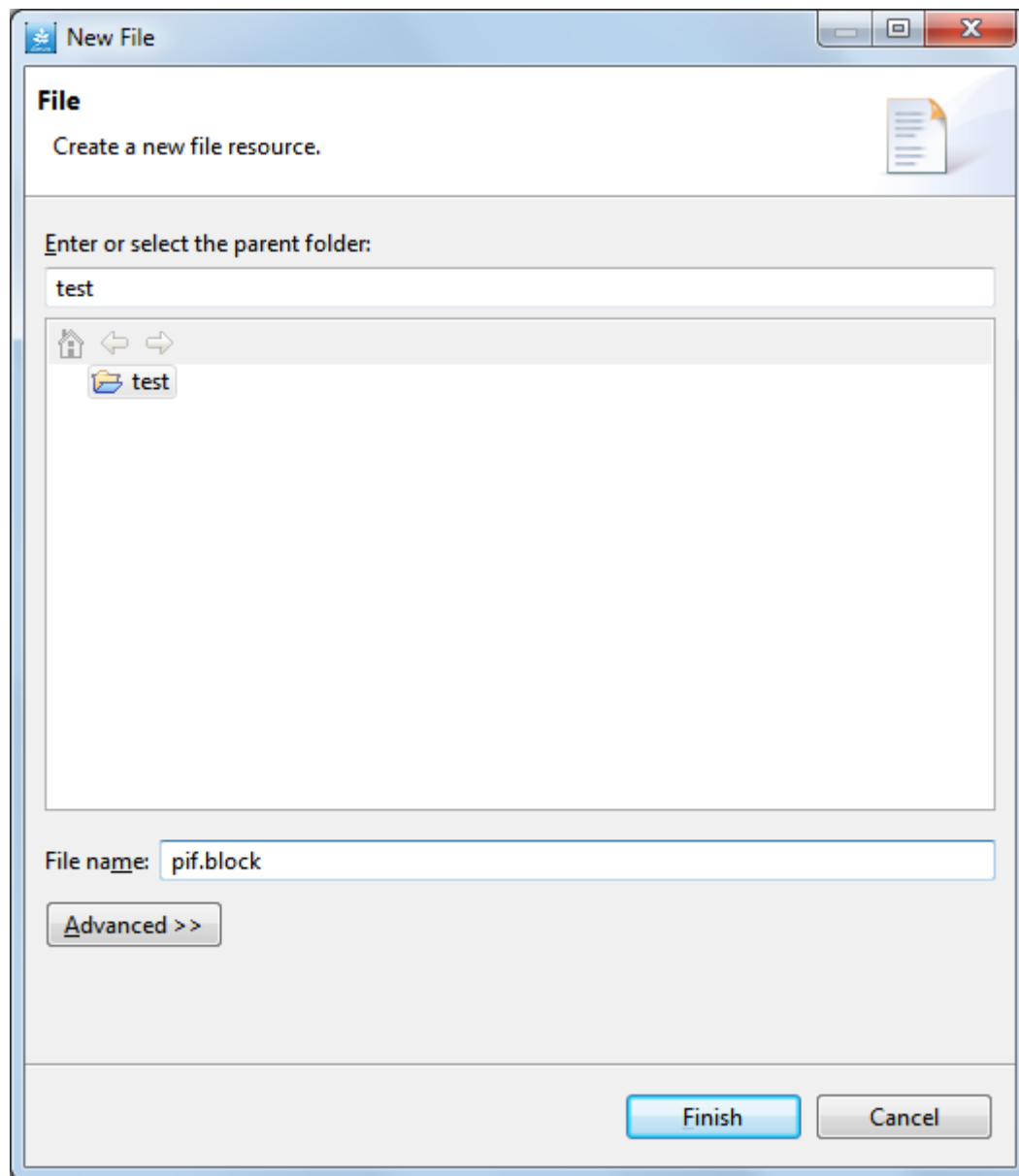
- Select menu File-> New, a new window will pop up.



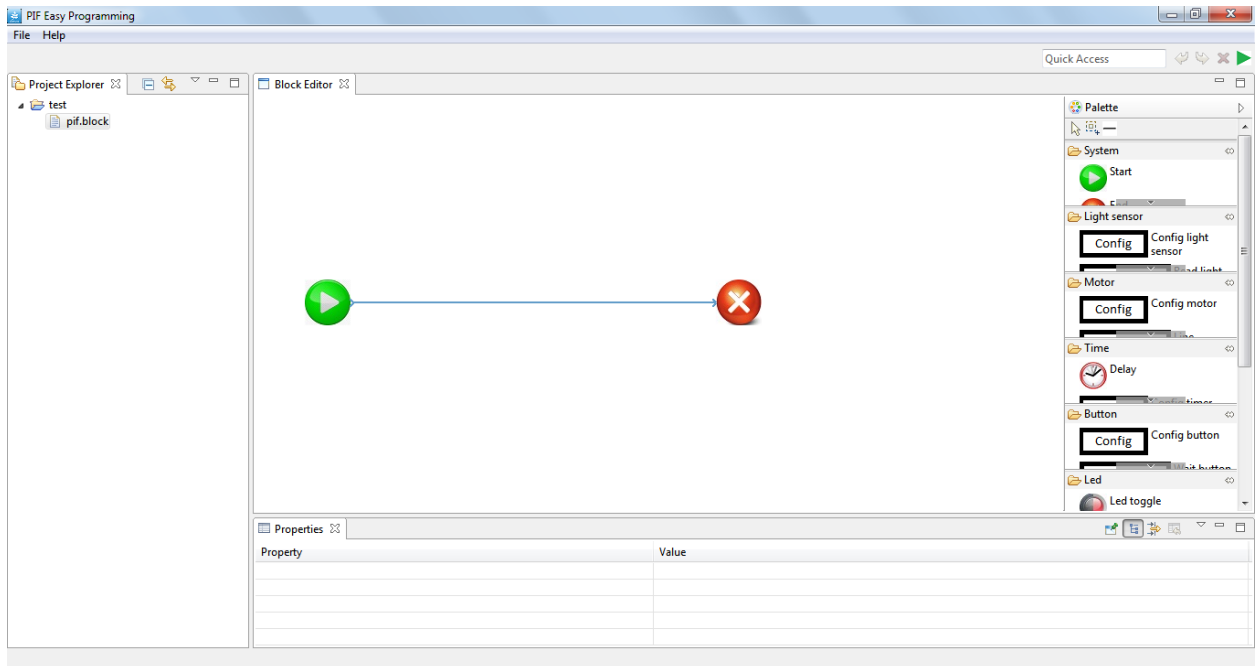
- Select Project -> Next.
- Type in project's name and click Finish.

4.2. Create new files

- Right-click on the project's name in Project Explorer window, select New->File, a new window will pop up.




- Select project to store files in the list of projects, type in file name with extension **.block**.
- ❖ **Notice:** with each project, user only creates one `.block` file at the beginning, other `.block` files will be created automatically by the software when you drag and drop blocks in programming window.
- When a new project is successfully created, a new programming window will appear with two blocks – Start and End – like the picture below:

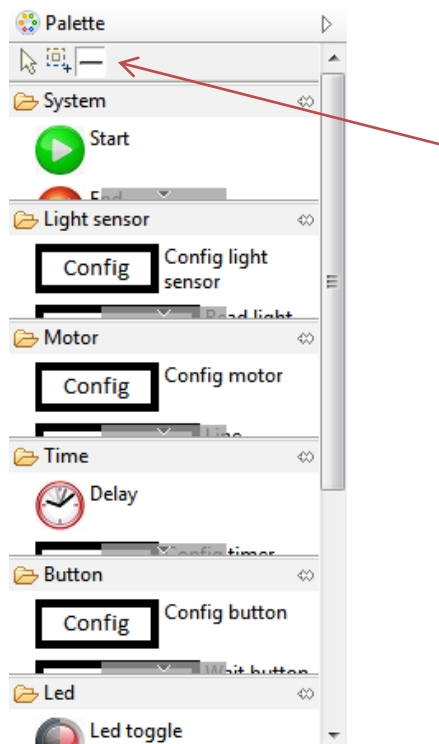


4.3. Adding blocks to the programming window

- Drag blocks from the block window into the programming window.

4.4. Connecting blocks

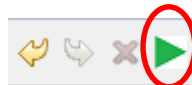
- Click on the  symbol on the block window and connect this block's output to the other block's input.



- To reconnect the blocks, click and move the start or end of a connection to other locations.

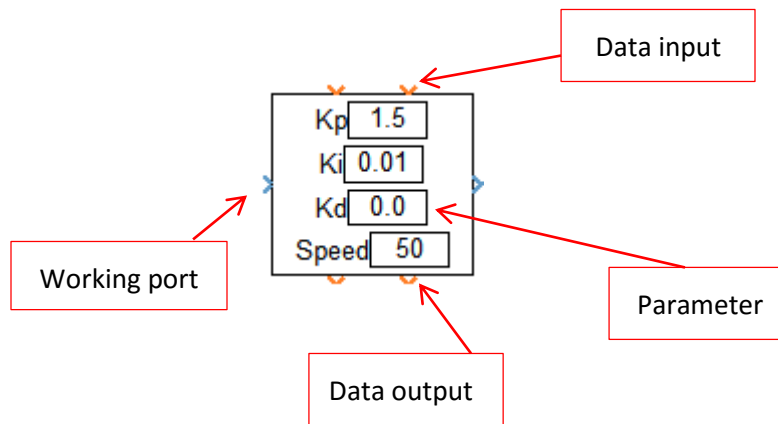
4.5. Download program to the robot

- Click on Run symbol (the green arrow on the toolbar) or press the shortcut Ctrl+R.



5. Function blocks

- In general, a function block consists of: working port (blue color, decides program working order), data port (orange, input/output data) and parameter blocks (decides how the block works).



- Data ports can only connect to Variable blocks in Function group.
- Parameters can be changed directly by clicking on the parameters' blocks or using the Properties window at the bottom of programming window.

List of function groups in a program:

5.1. System Group

Start Block: Start the main program, you can only place one Start block in your program.

Stop Block: Stop the main program, you can only place one Stop block in your program.

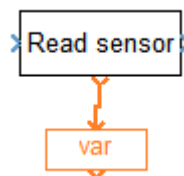
Config Block: configuring robot's modules.

Enable Interrupt Block: allows interrupt in the program.

5.2. Light sensor Group

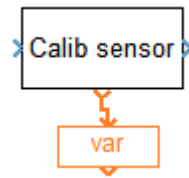
Config Block: configuring light sensor module.

Read Sensor Block: read light sensor's module data, sensor's data is stored in the Variable block which connects to data output port.



Check sensor Block: Check whether sensor reading is finished, usually after ReadSensor Block.

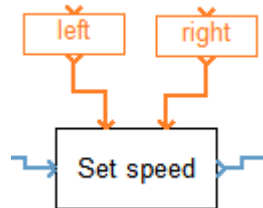
Calib sensor Block: get sensor's first data for calibration.



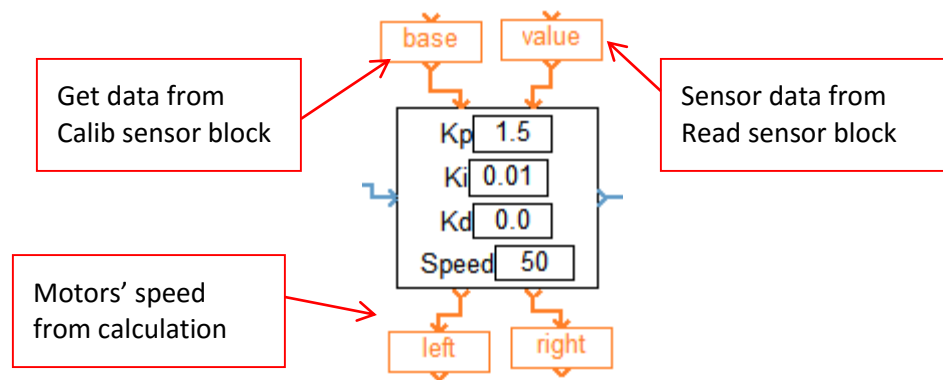
5.3. Motor Group

Config Block: configuring motor controller module.

Set Speed Block: set the desired speed to two motors.



Follow Line Block: process the line-following algorithm.



5.4. Time Group

Config Block: configuring timer module, the value in parameter block is the time interval between interrupt routine calls.

Delay Block: delay robot system for an amount of time (*ms*).

Timer Interrupt Block: double click to write timer's interrupt function.

5.5. Button Group

Config Block: configuring buttons' module

Wait Button Block: wait for user to press left or right button.

5.6. Led Group

Config Block: configuring LED module.

Led Toggle Block: toggle between LED's state.

Led On Block: turn on left or right LED.

Led Off Block: turn off left or right LED.

5.7. Buzzer Group

Config Block: configuring buzzer module.

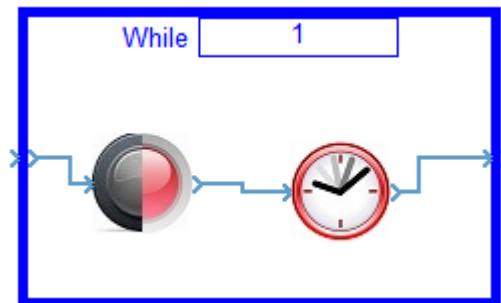
Buzzer On Block: turn on buzzer.

Buzzer Off Block: turn off buzzer.

5.8. Structure Group

While Block: blocks placed in While block will be processed in an infinite loop.

Example: the following program will change the LED's state after a period of time determined by the delay block.



5.9. Function Group

- *Subfunction Block*: double click to create a subfunction, usually used with Call function block.
- *Call function Block*: to call subfunction, used with Subfunction block.

Example:

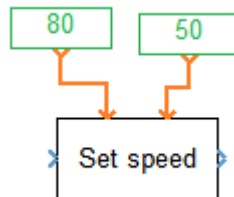
The following program creates a subfunction named buzzer and call this function, delay an amount of time and call it again.



Buzzer function will turn on the buzzer for a short time then turn it off.

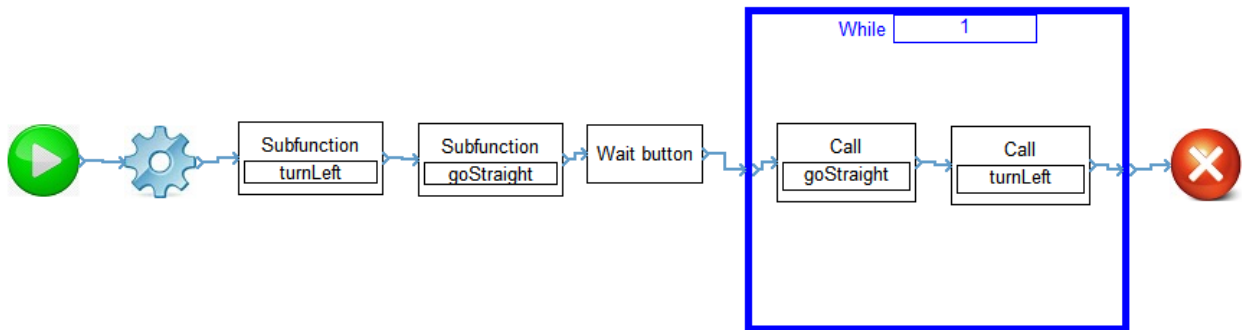


- *Variable Block*: stores the input/output data of other blocks, connects to other block's data port.
- *Constant Block*: stores constant number data, used to connect data input port of Set Speed Block.

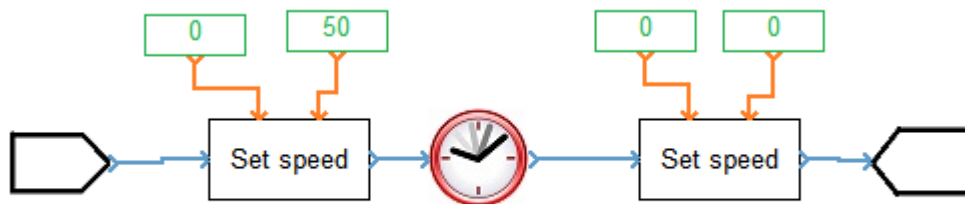


6. Program Examples

Exp 1: Program to get iBot follow a square path.

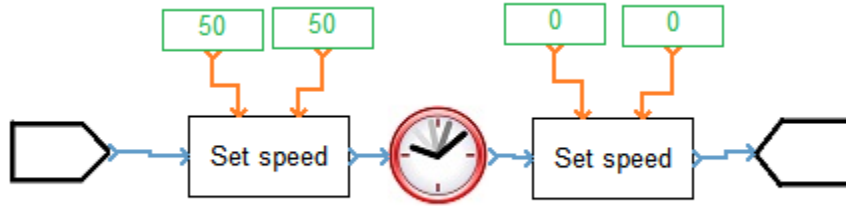


Function turnLeft

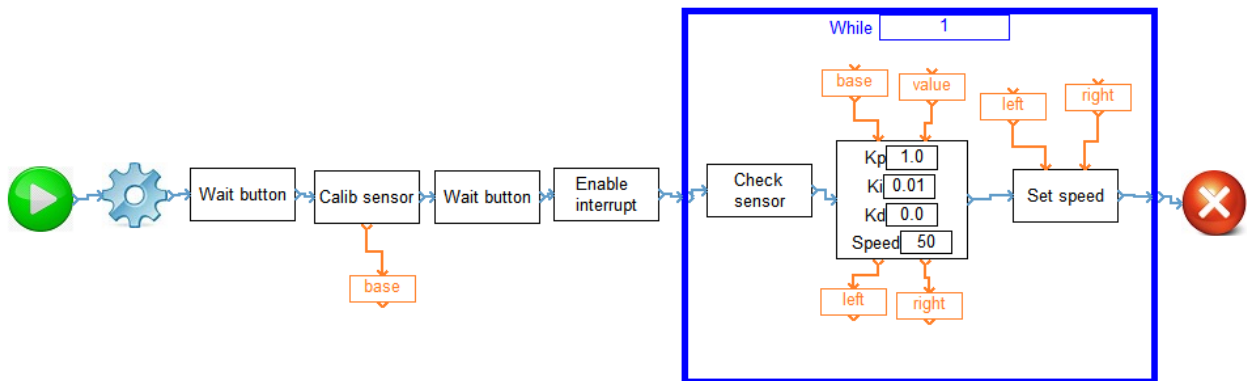


Choose the appropriate delay time to turn robot 90 degrees.

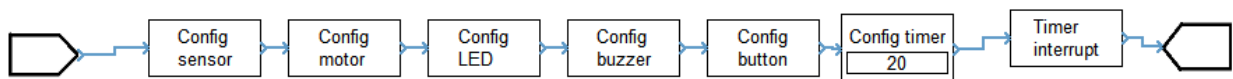
Function goStraight



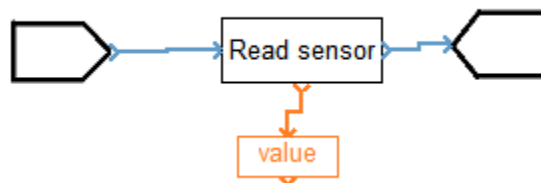
Exp 2: Line follower



In Config block, add two blocks *Timer Config* and *Timer Interrupt*.



Timer Interrupt Block



7. Possible errors and causes

7.1. *There are at least two Start blocks.*

You can only have one Start block in your program.

7.2. *There is no Start block.*

You need to have one Start and Stop block in your program.

7.3. Control flow is not continuous.

The work port of all blocks must be connected.

7.4. Data port is not connected yet.

The block's data port needs to be connected.

7.5. Variable has inconsistent type, Constant and data port have inconsistent data type.

Incorrect data type for at least one block's parameter. Input and output data port of PID block has different types and cannot be stored in a same Var block, you can try changing the Var block's name.

7.6. Function ... is called but is not declared yet.

The function that you want to call is not yet created. You can use the subfunction block to create that function.

7.7. There are at least two interrupt blocks from one module.

You can not have more than one interrupt block in your program, so make sure you leave only one interrupt block in your program and delete the others.

8. Download program and author information:

8.0 Java Runtime Environment (JRE)

Install JRE to run PIF Easy Programming.

www.java.com/en/download/manual.jsp

8.1 PIF Easy Programming bản 32-bit v1.0

https://github.com/PIFClub/PIFEasyProgramming/tree/master/PIFEasyProgramming_x86/PIFEasyProgramming_v1

8.2 PIF Easy Programming bản 64-bit v1.0

https://github.com/PIFClub/PIFEasyProgramming/tree/master/PIFEasyProgramming_x64/PIFEasyProgramming_v1