## Speech script for Stepper Interfacing with Firebird V (LPC2148) video tutorial

|  |  |
| --- | --- |
| Title Page | Hello everyone! Welcome to the video tutorial on Firebird V robotics research platform. This platform is based on the LPC2148 microcontroller from the ARM7 family. In this tutorial we will learn about stepper motors, ways to control them and how to interface a stepper motor with the Firebird V robot. |
| Agenda for Discussion | Let's see the agenda for discussion in this tutorial.  First we will have an introduction where we will discuss what is a stepper motor and its types.  Then we will move on to learn how to control a stepper motor in different stepping sequences like wave, full and half stepping modes followed by a comparison of the three stepping modes.  Then we will have a short demonstration of how to identify the wires of a usually unlabeled stepper motor.  This will be followed by a discussion of the circuitry required to drive a stepper motor and then finally we will jump on to actually programming the robot to control the stepper motor. |
| What is a stepper motor? | So the first question that arises is, What is a stepper motor? A stepper motor is a special kind of motor,  **(Next)**  whose rotation is divided into discrete steps which allow precise control over its angle.  **(Next)**  It can be commanded to hold a step or move to the next step. |
| Types of stepper motor | Types  Steppers are mainly classified on the basis of their internal wiring. These classes are as follows.  **(Next)**  Bipolar Steppers usually have 4 wires to control them and their drivers are complex  **(Next)**  Unipolar Steppers on the other hand have 5 or 6 wires and are relatively easier to drive  **(Next)**  Since, Unipolar motors are commonly used, we will be using them in this tutorial |
| Stepping sequences | Moving on to how to control a stepper motor, a stepper motor can be controlled by sending specific sequences of signals to its wires. The different stepping sequences are  **(Next)**  Wave Stepping  **(Next)**  Full Stepping  **(Next)**  and Half Stepping  **(Next)**  We will now see and understand how to use each one of them. |
| Wave Stepping | Wave Stepping  It involves exciting each coil turn by turn in a circular fashion.  **(Next)**  The table shows which coils are to be excited for generating the wave stepping sequence. Here a 1 denotes ON whereas a 0 denotes OFF. To rotate the stepper motor in one direction, we send the sequence, 1-2-3-4-1 and so on while for the opposite direction, we send the sequence in reverse, i.e., 4-3-2-1-4 and so on. |
| Wave Stepping (contd.) | Shown here is an illustration that depicts the direction of the rotor inside a stepper motor in the wave stepping sequence. The rotor points to the excited coil at that instant. When coil A is excited, the rotor aligns itself to point to it. Next, when coil B is excited, the rotor turns to align itself again. This process repeats for the other coils. |
| Full Stepping | Full stepping  It involves exciting two adjacent coils turn by turn in a circular fashion.  **(Next)**  The table shows which coils are to be excited for generating the full stepping sequence. Notice that at each step exactly two coils are excited.  Now let's see what happens inside the motor when two coils are excited at once. |
| Full Stepping (contd.) | As you can see, when two coils, say A and B are excited, the rotor settles between them. This results in a higher torque than that found in wave stepping. |
| Half Stepping | Half Stepping  This stepping mode is a combination of wave and full stepping.  **(Next)**  If you look closely, half stepping uses full stepping between the step positions of wave stepping. Let's see what happens inside the motor during this sequence. |
| Half Stepping (contd.) | As you can see, there are 8 positions corresponding to the 8 steps shown in the previous table. It is called half stepping because it effectively halves the step angle and offers more resolution. |
| Comparison of stepping modes | We will now see the differences and similarities between the 3 different stepping modes  **(Next)**  Talking about torque, since only one coil at a time is energised in wave stepping, its torque is the lowest. In full stepping, 2 coils are excited at once creating a larger attractive force resulting in a larger torque. Since half stepping excites 1 or 2 coils per step, the torque output is intermediate.    Vibration in a stepper motor increases with torque and decreases when the step angle decreases. Since wave stepping has low torque, but a step angle larger than half stepping, it has an intermediate level of vibration. Full stepping has higher torque resulting in more vibration and half stepping, due to a smaller step angle exhibits the least vibration.    Next is speed. Speed is equal to the step angle times the step frequency, since the step angle is the same for wave and full stepping, they have the same speed. For half stepping, the stepping angle is halved and thus the speed is also halved.    Resolution is fineness of control over rotational angle and is equal to the step angle. The step angle is equal for wave and full stepping while it is halved for half stepping. |
| Identifying the wires of a stepper motor | We will now learn how to identify the wires of a stepper motor.  **(Video to show how to identify wires of stepper motor – to be inserted using video editing tool)** |
| Stepper Motor Driver Circuit | This is the circuit diagram for the stepper motor driver  The ULN2003 is a transistor array package of 7 transistors that are capable of switching 500 milli Amperes of current per transistor. The pins 1-7 are the inputs to the transistors, while the pins 16-10 are the corresponding open-collector outputs. All transistors have a commmon emitter that is connected to the ground at pin 8. The COM pin at pin 9 is connected externally to the supply voltage. The circuit is basically used to drive the stepper's high current windings by a microcontroller's General Purpose IO pins. |
| Interfacing LPC2148 - GPIO pins | We will now discuss how to interface the ARM7 based Firebird V robot with the stepper motor using this circuit  **(Next)**  These are the Firebird V robot's JTAG connector pins which we are going to use to connect to the driver circuit shown previously.  pin 11 on the JTAG connector i.e., P1.26 is connected to the ULN2003 driver IC's pin 1  pin 13 i.e., P1.27 is connected to the ULN2003 driver IC's pin 2  pin 5 i.e., P1.28 is connected to the ULN2003 driver IC's pin 3  pin 9 i.e., P1.29 is connected to the ULN2003 driver IC's pin 4  pin 4 i.e., Ground is connected to the ULN2003 driver IC's Ground on pin 8  **(Next)**  The pins are numbered on the connector as shown in the figure. We will be using the pins 11, 13, 5, 9 and 4. The numbering is done assuming you are looking into the 20-pin male FRC connector on the Firebird. |
| Interfacing with LPC2148 - Code | Now for an example, we will be rotating the stepper motor in one direction at a fixed speed for a complete revolution and then change its direction and repeat the process in the opposite direction. For this, we can either use delays between steps or we can use one of the timers for the delays. For this video we will be using simple delays for the job.  **(Next)**  This is a list of headers to be included, which are: lpc214x.h and a custom header called stepper.h that contains the functions for controlling the GPIO pins of the stepper in different step modes.  **(Next)**  Moving on to the main function, first we initialize the IO ports for the stepper motor.  **(Next)**  Then in an infinite while loop, we have a call to the function full(underscore)step which takes direction as argument and is defined in stepper.h and will be discussed in the next slide.  **(Next)**  This is followed by a delay of 10 milliseconds so that the resulting step frequency is 100Hz.  **(Next)**  direction here is a global variable that is initially set to +1 and is changed to -1 after every 200 steps by the following code which increments count in each iteration and changes direction from +1 to -1 and vice versa if count exceeds 200. It then resets the count to 0.  The stepper motor we use, completes a revolution in 200 steps. Modify this value if your stepper motor's steps per revolution value is different. |
| Interfacing with LPC2148 – Code (contd.) | Now let's have a look at the port intialisation function, stepper(underscore)port(underscore)init. The function starts by setting the PINSEL2 register's bit 3 on and bits 2, 1 and 0 off. Bits 1 and 0 are reserved bits and bit 2 is set to zero to disable the JTAG functionality of the pins to allow us to use them as GPIO pins. The second line sets the four pins as output pins as specified in the #define constants COILAPIN, COILBPIN, COILCPIN and COILDPIN in the register represented by the constant STEPPERPORTDIR which equals IO1DIR, i.e., the port direction register for IO Port1.  Using #define constants for ports and pins allow greater flexibility and reusability of code should you later decide to use another port or pin. Although it is used only once here, it will be used more often in the step functions that we will discuss later. |
|  | **(Show code in Keil uV4 IDE)**  **(Show working video of stepper)**  So by now we have successfully understood the control of stepper motors and how to interface a stepper motor with the LPC2148 based Firebird V robotics platform. You may modify the code to experiment with different stepping modes or by setting it to specific angles or by using timers or basically anything that your imagination leads you to. |
| Thank you | With this we have reached the end of this video tutorial. Thank you for listening! For any doubts or suggestions feel free to mail them at helpdesk@e(hyphen)yantra(dot)org  This is Joel Pinto signing off! |