## Speech script for Servo 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 of ARM family. In this tutorial we will learn about Servo motors, their working, how to control them and how to interface a Servo motor with the Firebird V robot. |
| Agenda for Discussion | Let's see the agenda for discussion in this tutorial.  The presentation starts with how a servo motor works, its internal circuitry, what makes it different from dc motor and how can we rotate it. We will learn selecting a servo for specific use and how to interface servo with Firebird-V.  Then finally we will jump on the actual programming for the same. |
| Prerequisite knowledge | Before we proceed with the tutorial, make sure you know the input/output interfacing using ports in and know how to work with PWM registers of LPC2148. |
| Introduction | So, let us see what a servo motor is. It is basically a dc motor that can rotate with precise angle, speed and acceleration.  Servos can put up about 42 oz/inch of torque.  They are easily found in toys like model car, airplanes, etc. controlling steering or adjusting wing surfaces.  They are relatively inexpensive. Hence widely used for educational purpose in mechatronics as they can be controlled via controller. |
| Principle of Working (Under section: Servo Motor) | A servo mainly contains a dc motor coupled with gears having proper gear ratio and a sensor for position feedback. Generally, this work is done by coupling it with a potentiometer. And a control circuitry to form a closed loop control system.  **(Next)**  Looking into the control system, we find that the output port is connected with one of the input terminals of the error detector amplifier and electrical signal is given to rotate servo is given at another terminal of error detector and amplifier.  **(Next)**  The difference between the signals to the error detector are amplified and applied to dc motor to drive it.  **(Next)**  This in turn rotates the potentiometer which is coupled to the shaft of dc motor using gears. As the angular position of the potentiometer knob progresses the output or feedback signal increases.  **(Next)**  When the potentiometer reaches the desired position, the error in the signal from output and the applied control signal is negligible. Hence, there would be no input signal to the dc motor to rotate it.  Continuously applying the same control signal makes the motor to stay at that position. This is how a simple conceptual servo motor works. |
| Operating Servo Motor | Now let us see how to operate a servo motor.  Servo motor is operated using the wires provided in it.  It is driven to a particular angle by control signal sent to it via a controller. These signals are basically Pulse Width Modulated (PWM) waves and that angular position is determined by their ‘on-time’.  **(Next)**  This on-time period for various angles depends on the servo model and the manufacturer while not on the frequency of PWM.  **(Next)**  It is interesting to note that this angular position is independent of the duty cycle of the PWM signal.  **(Next)**  Generally this time period is around 1ms for 0o and around 2ms for 180o but depends on model and the manufacturer as said above. Graph of on-time period vs. respective angle is linear. Therefore, the values of time period for angles other than these can be easily calculated by getting the eq. of line from the 2 values given. |
| Selection of Servo motor | The typical specifications of servo motors are torque, speed, weight and its dimensions.  A manufacturer may compromise torque over speed or speed over torque in different models. The weight and dimensions are directly proportional to the torque. The selection of a servo should be made as per the requirement of these specifications. |
| Controlling servo using Firebird V | Now let’s move on to the interfacing part.  This figure shows the servo connector. The ground wire marked as no. 1 in the fig. is generally brown or black in colour. The no. 2 pin, i.e. power pin is generally red in colour. While, the signal pin is typically yellow or orange in colour which is marked as no.3  **(Next)**  This figure shows where and how to connect servo connector with firebird V robot. |
| Using PWM register for PWM generation | Here, we would use PWM register for generating PWM signal. Let us use the most commonly used PWM mode i.e. single edged PWM signal. We would use the internal crystal of the controller for delay purpose in timer and for counting.  The prescaler is very helpful when the crystal frequency changes. The change in the clock rate can easily be compensated by changing the prescaler accordingly without changing other code.  Let’s choose prescaler as 120.  **(Next)**  Now, the frequency of crystal is 12 MHz; Therefore, the counter increments its value every 0.01ms.  The main logic for generating PWM signal is the counter counts from 0 to a count value set in PWMMR0. Now, when the value of the counter is less than the reference value provided in the match register1, the o/p where the servo is connected is set high else low.  Hence, in this way, we can set on-time period as well as frequency of the PWM signal.  **(Next)**  Let the frequency of signal be 50 Hz that gives value of PWMMR0 as 2000.  **(Next)**  Values for register like PWMTCR and PWMSEL can be found using ARM7 LPC2148 datasheet.  **(Next)**  Let us calculate the value for on-time period i.e. for MR1. As the frequency is 120 the resultant frequency of the counter will be 12 MHz/120 which is 100 kHz.  **(Next)**  Seeing other way round it will take 1/100 = 0.01ms for counter to increment by 1 value.  **(Next)**  Therefore, total counts needed to be increment for getting time delay of 0.6 ms will be 60. Similarly for 180o we get 220.  **(Next)**  The relation between on-time period and corresponding degree is linear for every servo motor. Also we know that relation between time delay and counts required is also linear for given prescaler. Hence, we can obtain eq. of line by plotting count vs. respective degree from values of count for 0 and 180o.  **(Next)**  The slope of the line will be 1/1.125 and the intercept on y-axis will give us the value of count for 0o which = 60. Therefore the eq. of MR1 will be slope into degrees plus y intercept value which is as sown here. |
| Code | Now, let’s jump to coding part.  This block shows the header file which is needed to be included in the code for programming the controller.  **(Next)**  This block shows the application of general eq. of line derived earlier.  **(Show code in Atmel Studio)**  Now, we will write a code in Keil and try to burn it on robot. Configure Keil for ARM7 LPC2148. Here, I have already configured Keil and got the program ready.  Looking at the program the first thing that we see is header files which are to be included to use inbuilt functions.  This function delaymsec is used to provide delay for servo to rotate to certain angle. Here comes the initialization of PWM register. This line selects PWM1 o/p for port 0.0 single edged PWM is selected by this line PWMPC-0. Than we have set the prescaler to 120. First we shall disable the counter and set the required on-time period and frequency of the required PWM signal. For tis PWMPC=0 is used. And as we learnt in the earlier section, PWMMR0 is used to set the frequency of the required PWM signal. The value of PWMMR0=2000 is set for 50 Hz PWM signal and is calculated using the formula we discussed in the earlier section. PWMMR1=60 sets the counter value to 60 until which the o/p will be high and ten it would become low till it reaches PWMMR0 register’s value. The rest of the registers PWMMR2 to 6 are set to 0 i.e. disabled. PWMMCR sets the command to reset the o/p when counter reaches the value in PWMMR1. PWMLER is basically used to update match registers. Finally, we start the counter using this function.  This function converts the degree in terms of count values and sets that in PWMMR1 register and is updated by PWMLER. Here the value 02 in hexadecimal sets the command to update the 1st bit of MR i.e. PWMMR1 for servo to rotate to next angle. This function frees the servo so that sufficient power can be used by other servo. This function is used to initialize PWM as discussed ere.  Now, comes the main function.  Here we set all the pins to GPIO pins first then we initialize registers. Now command to move servo are given. Here we just need to specify the angle to which the servo has to be rotated like Servo\_1(90) will rotate the servo to 90o by passing 90 as argument to the Servo\_1 function.  Here the servo will continuously rotate from 90o-180o-90o-0o-90o infinite no. of times as it is in the infinite loop.  Now let’s have a look at the o/p after building the code and burning it using software like flash magic.  **(Show working video of servo)** |
| Thank you | Going through the tutorial and knowing how we can drive servo motor and the how to program a controller for that, you can try different things like velocity control, 1 degree precision, etc. You can also challenge the operating frequency range of PWM.  With this we end this video tutorial here. Thank you for listening! For any doubts or suggestions feel free to mail them at helpdesk@e(hyphen)yantra(dot)org  This is Vishal H. Rajai signing off! |