Title of script: Using ADC on Firebird V

Author: Shailesh Jain

Keywords: ADC, register, channels.

|  |  |
| --- | --- |
| Slides | Narration |
| 1. Title | * Hello Friends. * Welcome to the spoken tutorial on Firebird V Robotics Research Platform. * This platform is based on LPC 2148 microcontroller which belongs to ARM architecture based microcontroller family. * In this tutorial, we will study the **ADC interfacing on Firebird V robot**.   **(press next)** |
| 1. Agenda | * Now let us consider the agenda of discussion in this tutorial. * This includes the mechanism of operation ADC of LPC 2148 with a description of various ADC channels. * We also use ADC Interfacing on Firebird V robot which includes the initialization and configuration of various registers. So what is ADC? It stands for Analog to Digital Conversion. Why is there a need for ADC?   **(press next)** |
| 1. Need for  ADC | * Fire Bird V has eight Analog IR proximity sensors**(press next)** * five Sharp IR range sensor**(press next)** * three white line sensors**(press next)** * battery voltage sensing sensor**(press next)** * All these sensors give analog output. We need to use LPC 2148 microcontroller's ADC to convert these analog values in to digital values.   **(press next)** |
| 1. In built ADC | * Let us now see the features of the in-built ADC in the LPC 2148. **(press next)** * The LPC 2148 features a 10-bit successive approximation ADC. The resolution of the ADC indicates the number of discrete values it can produce over the range of analog values. The values are usually stored electronically in binary form, so the resolution is usually expressed in bits. **(press next)** * It takes more than 2.44us conversion time. **(press next)** * LPC 2148 has two inbuilt ADC modules known as ADC0 and ADC1. * ADC0 has 6 channels **(press next)** * ADC1 has 8 channels **(press next)** * In LPC 2148 ADC can be operated in power down mode(**press next**) * The reference voltage for the ADC (VREF) indicates the conversion range for the ADC its typically 3V **(press next)** * In burst mode the ADC can simultaneously convert the data from multiple inputs or channels(**press next**) * ADC can start conversion depending on the pin status or timer match signal or it can do conversions repeatedly.(**press next**) * There is global start for both the convertors(**press next**) * The ADC operates in two modes i.e. free running and single conversion mode. **(press next)** * Interrupt flag is set when ADC completes the conversion(**press next**) |
| 1. ADC channels | * Now let us see ADC channels of LPC2148. **(press next)** * As you can see in the table we have 7 single-ended voltage inputs from the pins of PORT0 and PORT1.Each of this channels has the device connected to it, so that the values from this device can be read in digital format using the appropriate channel number.   **(press next)** |
| 1. ADC initialization | * The first step of the program is always the initialization so let us begin with the ADC initialization. **(press next)** * To Program ADC, we have to initialize some registers before use. * These registers are: **(press next)** * ADxCR - ADC Control Register and x can be 0 or 1**(press next)** * AD0CR is used for the ADC 0(**press next**) * And AD1CR is used for the ADC1(**press next**) * Both the registers are of 32 bits(**press next**) * As I have mentioned earlier that ADC 0 has only 6 channels. AD0.1,2,3,4,6 and 7 are available so AD0.0 and AD0.5 are not available in ADC0.ADC 1 has 8 channels AD1.0 to AD1.7.(**press next)** |
| 1. ADxCR | * Now let us see the bit description of ADxCR register ADC Control register**(press next) (press next)** * Bits 0 to 7 are known as SEL bits which are used to select the channel to be sampled and converted how they are used is described in next slide(**press next**)Initailly there value will be all zero(**press next**) * Next, bits 8 to 15, CLKDIV bits are used to produce the clock for ADC which should be less than or equal to 4.5MHz.(**press next**)Initially its value is 00001110. (**press next**) * 16th bit is BURST bit, This is to disable the repeated conversions(**press next**) and initially its value is kept at 0 because we want repeated conversions(**press next**) * Then bits 17 to 19 are CLKS bits, which selects the number of clocks per bits This bits are explained further in slide 9(**press next**) Initially its value is kept as 000. (**press next**) * Then bit 20 is researved for the system use(**press next**) so its value is 0(**press next**) * Bit 21 is PDN which is used to select between operational or power down mode. Here we want operational mode(**press next**) so we kept value as 1(**press next**) * Bits 22 and 23 are researved(**press next**) so there values are 00(**press next**) * Bits 24 to 26 are START bit and this bits controls whether and when the AD conversion is started. (**press next**) Initially its value is kept at 000(**press next**). * Bit 27 represent the EDGE bit, This bit is significant only when the start bits are 010 to 111 in our case this bit has no significance.It is used to select occurrence of conversion on falling edge or rising edge (**press next**)we have initialized it with 0(**press next**) * Bits 28 to 31 are researved(**press next**) kept at 0000(**press next**) * To summarize ADxCR register is initialized with the value 0x00200E00. (**press next**) |
| 1. ADxCR cont | * Here is a detailed description of bits 0 to 7 in ADxCR. (**press next**) * These 8 bits are used to select the channels in either ADC 0 or in ADC 1 to be sampled and converted. As an example if bit 7 in AD0CR is set then AD0.7 is sampled and converted similarly if bit 7 in AD1CR is set then AD1.7 is sampled and converted. So in this way these 8 bits controls the channels in both the ADCs.Specific bit is used for individual channel so that at a time more than one conversion can occur(**press next**) |
| 1. ADxCR cont | * Here is a description of bits 17 to 19 in ADxCR. (**press next**) * This field selects the number of clocks used for each conversion its selects between 11 clocks i.e. 10 bits and 4 clocks i.e. 3 bits. More the bits more will be the accuracy.(**press** **next** ) |
| 1. ADxCR cont | * Next is bits 24 to 26, START bits(**press** **next** ) * When the burst bit is 0 these bits control whether and when AD conversion is started * When the bit value is 000 there will be no start of conversion hence initially we kept it as 000. To start the conversion give value as 001. The conversion can be started depending on CAP/MAT signal but we will be not using this signal to start the conversion. (**press** **next** ) |
| 1. ADxGDR | * The next register is ADxGDR register which is A to D global data register. This register contains the ADC done bit and the result of most recent ADC conversion. This is also a 32 bit register. (**press** **next** ) * Bits 0 to 5 are reserved bits(**press** **next** ) * Bits 6 to 15 are RESULT bits, When ADC completes the conversion the result or the converted data is stored in this 10 bits(**press** **next** ) * Bits 16 to 23 are again reserved for the future use(**press** **next** ) * Then bits 24 to 26 are channel bits. As I have mentioned earlier that at a time more than one channel data can be sampled and converted so these bits indicate the channel number whose converted data is available in RESULT bits. (**press** **next** ) * Bits 27 to 29 are reserved bits(**press** **next** ) * 30 th bit is OVERUN bit, it is 1 when result of one or more conversions were lost(**press** **next** ) * The last bit is done bit,This bit is automatically set when ADC completes its conversion(**press** **next** ) |
| 1. Syntax for initialization | * Now let us look at how we write the program.(**press next**) First write a function to initialize ADC pins. **(press next)**. * In this function, P0.4,5 , 6, 12, 13, 28 and 29th pin are set as ADC pins using PINSEL registers. (**press** **next** ) * To set a pin as ADC, write value 11 in corresponding PINSEL bits. * Now to initialize both ADCs make another function named as Init\_ADC. (**press** **next** ) In this AD0CR and AD1CR registers are initialized as explained earlier. (**press** **next** ) (**press** **next** ) |
| 1. Syntax for main program | * Let us now begin with the syntax for C program. **(press next)** * In *main* we need to configure the ports using the function init\_peripherals. We have the infinite *while* loop in which we are using the LCD\_print() function to display the values of ADC on LCD. * Here we are using the channel numbers AD0.1, AD0.2 and AD1.3for reading the values of the White line sensors of the firebird robot **(press next)**. |
| 1. ADC Conversion program | * Here is a function for AD conversion for ADC0. (**press** **next** ) * This function is used for the purpose of conversion of analog to digital values **(press next).** * This function takes the value of char as input which is the channel number where conversion is to be done, and after the conversion it returns the char value obtained from conversion.   In the function initially we are selecting the appropriate channel number in SEL bits of AD0CR.  Then to start the conversion, give value 001 on START bits.  The *while* loop is then used to wait for the conversion to complete, the condition used in the while loop is wait until DONE bit of AD0GDR is set. The value obtained after the conversion is stored in RESULT bits of AD0GDR register which is stored in temporary variable Temp. Now only 8 RESULT bits of AD0GDR is of our use so to obtain that(**point cursor**) this last operation has been performed. Thus using the function we can obtain the digital value from the analog sensors through the use of appropriate ADC channel. (**press** **next** )  **(press next)** |
|  | * Similarly a separate function for AD conversion for ADC1 is also required. (**press** **next** ) |
| 1. Thank you | * So here we have successfully understood the mechanism of operation of ADC of LPC2148 and programmed ADC Interfacing on Firebird V robot. With this we have come to end to this tutorial. Thank you for listening. For any queries or doubts you can visit <http://qa.e-yantra.org/> * This is Bhumika Varshney Signing off!! |