

Lab-3

Friday, October 09, 2020 11:27 AM

Title:- Configuring ADC with LPC2129

Date:- 02/09/2020

Program :-

AIM: Select channel no. 0 for giving analog input 3.3 V and configure ADC control register ADCR for the channel no 0 and store the output of ADC in a separate register

Source Code:

```
#include<LPC21xx.h>

int main()

{

    unsigned int val; //Declaring register for polling approach

    PINSEL1=0x00400000; // Selecting channel 0 AINO

    ADCR = 0x00210401; // configuring ADCR FOR CHANNEL 0

    // by polling approach

    while(!(ADDR & 0x80000000)) // loop will continue till done bit is not 1

    {

        val = ADDR>>6 & 0x3FF; // shifted by 6 because 15:6 bits are important, for

                                //getting last 10 digits we wii do AND operation

        with 3FF

    }

}
```

Output:

The image displays two screenshots of the A/D Converter configuration window. The top screenshot shows the initial configuration: ADCR: 0x00210401, SEL: 0x01, PDN and BURST are checked, CLKS: 11clk/10bit, CLKDIV: 0x04, START: None, A/D Clock: 600000. The bottom screenshot shows the configuration after selecting channel 1: ADDR: 0xC000FFC0, CHN: 0x00, DONE and OVERUN are checked, V3A: 3.3000, V/V3A: 0x03FF. Analog inputs remain the same.

2) Select channel no. 1 for giving analog input 3.3 V and configure ADC control register ADCR for the channel no 0 and store the output of ADC in a separate register.

```
#include<LPC21xx.h>
```

```
int main()
```

```
{
```

```
    unsigned int val; //Declaring register for pooling approach
```

```
    PINSEL1=0x01000000; // Selecting channel 1 AIN1
```

```
    ADCR = 0x00210402; // configuring ADCR FOR CHANNEL 1
```

```
    // by pooling approach
```

```
    while(!(ADDR & 0x80000000)) // loop will continue till done bit is not 1
```

```
    {
```

```
        val = ADDR>>6 & 0x3FF; // shifted by 6 because 15:6 bits are important, for  
        // 10 digits we wii do AND operation  
        with 3FF
```

}

}

Output:

A/D Converter

A/D Control

ADCR: 0x00210402 SEL: 0x02 ☒ PDN ☒ BURST ☐ EDGE

CLKS: 11clk/10bit CLKDIV: 0x04

START: None A/D Clock: 600000

A/D Data

ADDR: 0x0100FFC0 CHN: 0x01 ☐ DONE ☐ OVERUN

V3A: 3.3000 V/V3A: 0x03FF

Analog Inputs

AIN0: 0.0000 AIN1: 3.3000 AIN2: 0.0000 AIN3: 0.0000

A/D Converter

A/D Control

ADCR: 0x00210402 SEL: 0x02 ☒ PDN ☒ BURST ☐ EDGE

CLKS: 11clk/10bit CLKDIV: 0x04

START: None A/D Clock: 600000

A/D Data

ADDR: 0xC100FFC0 CHN: 0x01 ☒ DONE ☒ OVERUN

V3A: 3.3000 V/V3A: 0x03FF

Analog Inputs

AIN0: 0.0000 AIN1: 3.3000 AIN2: 0.0000 AIN3: 0.0000

A/D Converter

A/D Control

ADCR: 0x00210402 SEL: 0x02 ☒ PDN ☒ BURST ☐ EDGE

CLKS: 11clk/10bit CLKDIV: 0x04

START: None A/D Clock: 600000

A/D Data

ADDR: 0x8100FFC0 CHN: 0x01 ☒ DONE ☐ OVERUN

V3A: 3.3000 V/V3A:

Analog Inputs

AIN0: 0.0000 AIN1: 3.3000 AIN2: 0.0000 AIN3: 0.0000

3) Select channel no. 3 for giving analog input 2.9 V and configure ADC control register ADCR for the channel no 3 and store the output of ADC in a separate register.

Source code:

```
#include<LPC21xx.h>
```

```
int main()
```

```
{
```

```
unsigned int val; //Declaring register for pooling approach
```

```
PINSEL1=0x10000000; // Selecting channel 3 AIN3
```

```
ADCR = 0x00210408; // configuring ADCR FOR CHANNEL 3
```

```
// by pooling approach
```

```
while(!(ADDR & 0x80000000)) // loop will continue till done bit is not 1
```

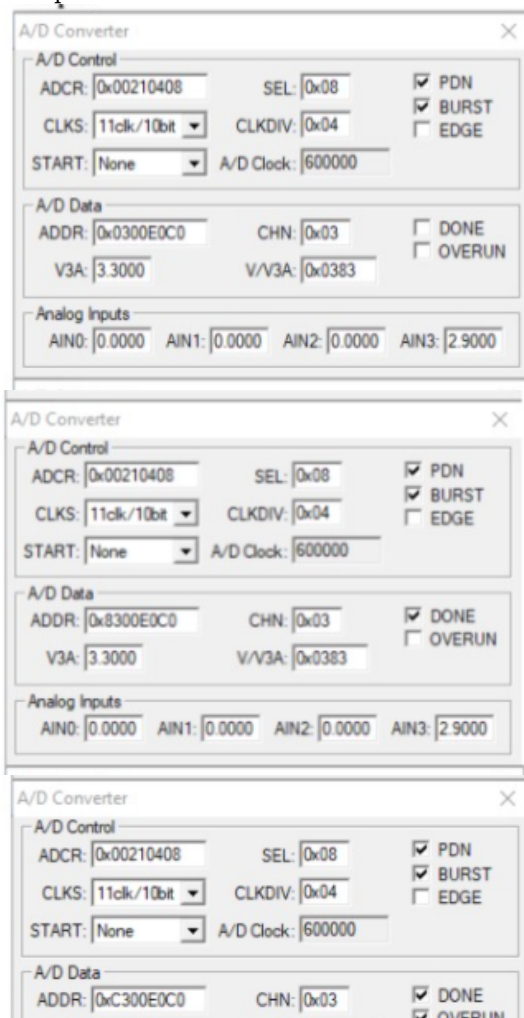
```
{
```

```
    val = ADDR>>6 & 0x3FF; // shifted by 6 because 15:6 bits are important, for  
    getting last                // 10 digits we wii do AND operation  
    with 3FF
```

```
}
```

```
}
```

Output:



V3A:	3.3000	V/V3A:	0x0383
Analog Inputs			
AIN0:	0.0000	AIN1:	0.0000
AIN2:	0.0000	AIN3:	2.9000

Inference:

In this experiment I learnt how to configure ADC with LPC2129, its registers ADCR.2, how to select channels.