EMBEDDED SYSTEMS END SEMESTER PROJECT

S5 CSE C

ANALOGUE TO DIGITAL CONVERTER

GROUP MEMBERS

Aishwarea Gupthan - AM.EN.U4CSE20203

Anagha Manoj - AM.EN.U4CSE20207

Arpitalaxmi - AM.EN.U4CSE20212

Hrishika Dayan - AM.EN.U4CSE20234

Keerthana A - AM.EN.U4CSE20241

Gopika Menon - AM.EN.U4CSE20245

TABLE OF CONTENTS

INTRODUCTION

An analogue Digital Converter, or ADC, is a data converter which allows digital circuits to interface with the real world by encoding an analogue signal into a binary code.

Microcontrollers help in communicating with other devices, like sensors motors, switches, memory, and even another microcontroller. As we all know many interface methods have been developed over years to solve the complex problems of balancing the needs of features, cost, size, power consumption, reliability, etc. but the ADC Analog-to-Digital converter remains famous among all.

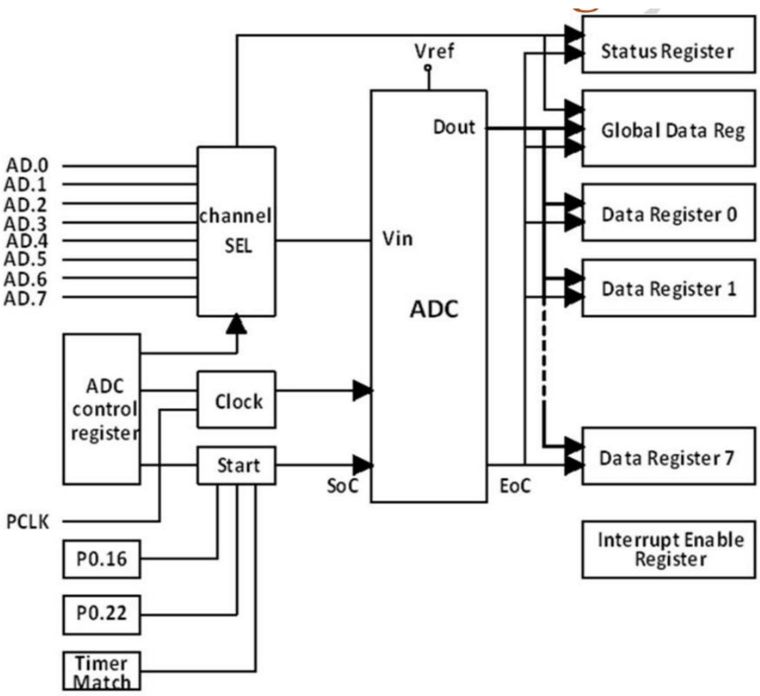
Digital signals propagate more efficiently than analogue signals, largely because digital impulses are well-defined and orderly. They're also easier for electronic circuits to distinguish from noise, which is chaotic. That is the chief advantage of digital communication modes. To implement ADC, we'll use the Keil uvision and Proteus software. ADC0 and ADC1 are two 10-bit ADCs incorporated into the LPC2148.

FEATURES OF ADC

The ADC has 6 registers. They are

1) ADCR
2) ADCGR
3) ADCINTEN
4) ADDR0 – ADDR7
5) ADDSTAT

BLOCK DIAGRAM



LPC2148 has 2 built-in ADC's.

The ADC reference voltage is measured across GND to VREF, meaning it can do the conversion within this range. Usually, the VREFP is connected to VDD.

ADxCR: A/D Clllllll Register: Used for Configuring the ADC

ADxGDR: A/D Global Data Register: This register contains the ADC's DONE bit and the result of the most recent A/D conversion

ADxINTEN: A/D Interrupt Enable Register

ADxDR0 - ADxDR7: A/D Channel Data Register: Contains the recent ADC value for the respective channel

ADxSTAT: A/D Status Register: Contains DONE & OVERRUN flag for all the ADC channels

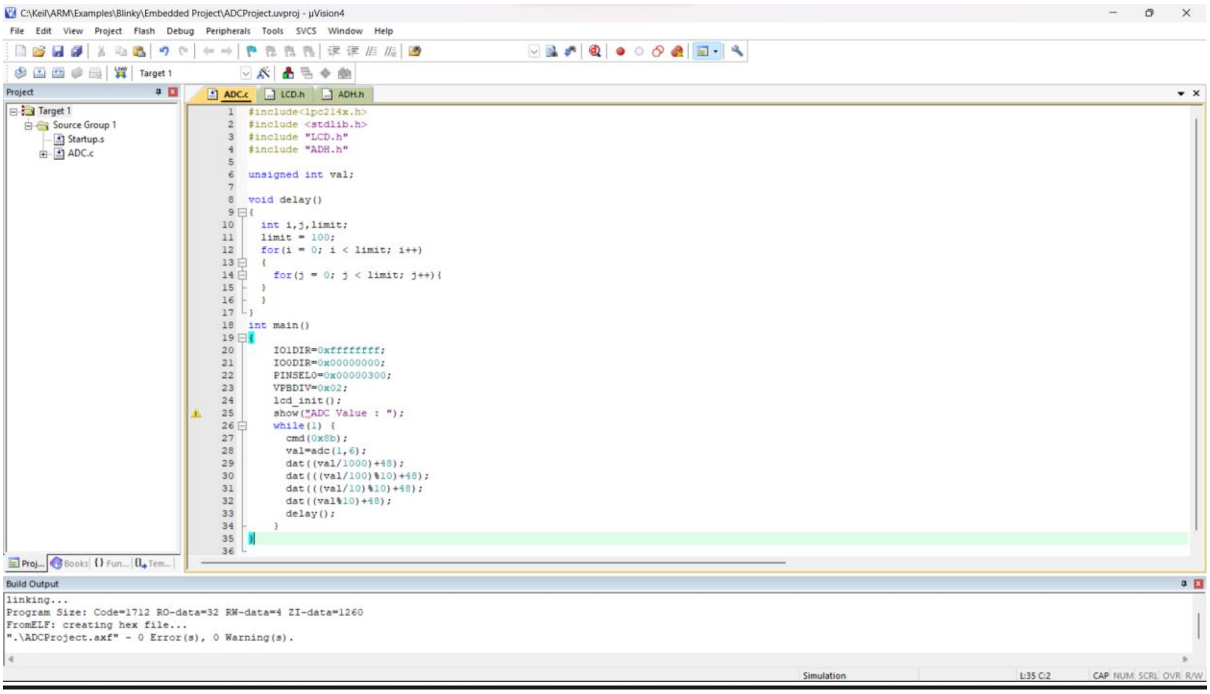
ADxGSR: A/D Global Start Register: This address can be written (in the AD0 address range) to start conversions in both A/D converters simultaneously.

WORKING

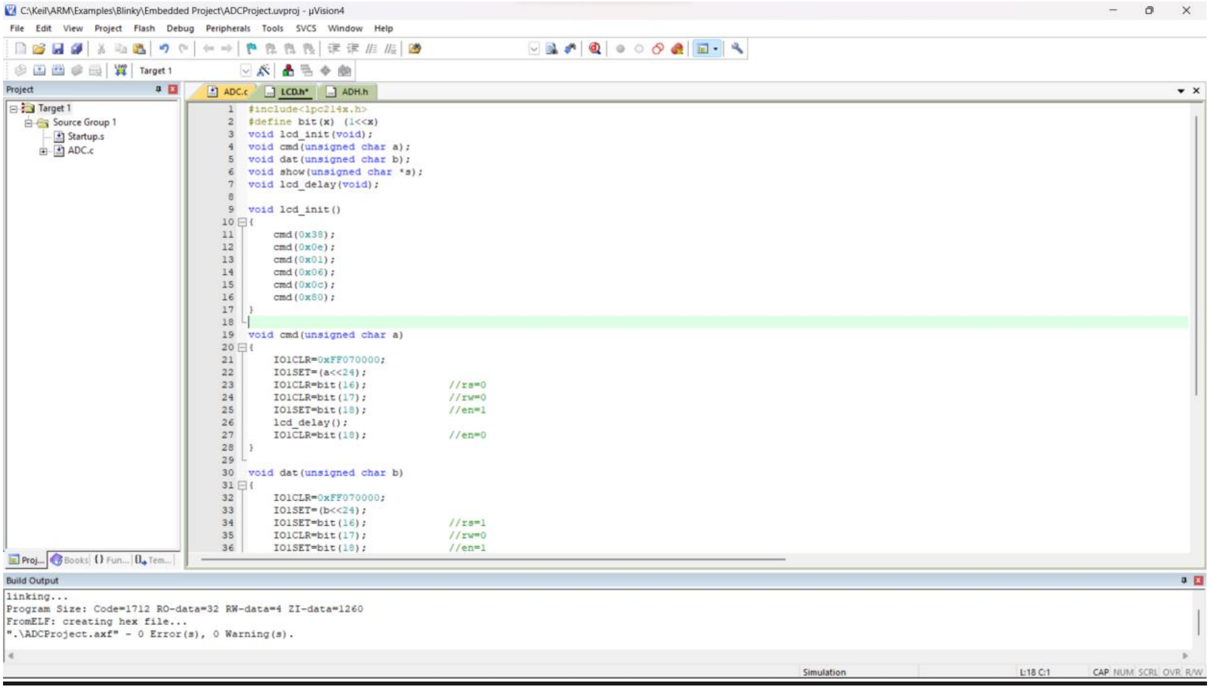
1. Configure the ADxCR (ADC Control Register)
2. Start ADC conversion by writing the appropriate value to START bits in ADxCR. (Example, writing 001 to START bits of the register 26:24, conversion is started immediately).
3. Monitor the DONE bit (bit number 31) of the corresponding ADxDRy (ADC Data Register) till it changes from 0 to 1. This signals the completion of the conversion. We can also monitor the DONE bit of ADGSR or the DONE bit corresponding to the ADC channel in the ADCxSTAT register.
4. Read the ADC result from the corresponding ADC Data Register.
5. ADxDRy. E.g. AD0DR1 contains the ADC result of channel 1 of ADC0**.**

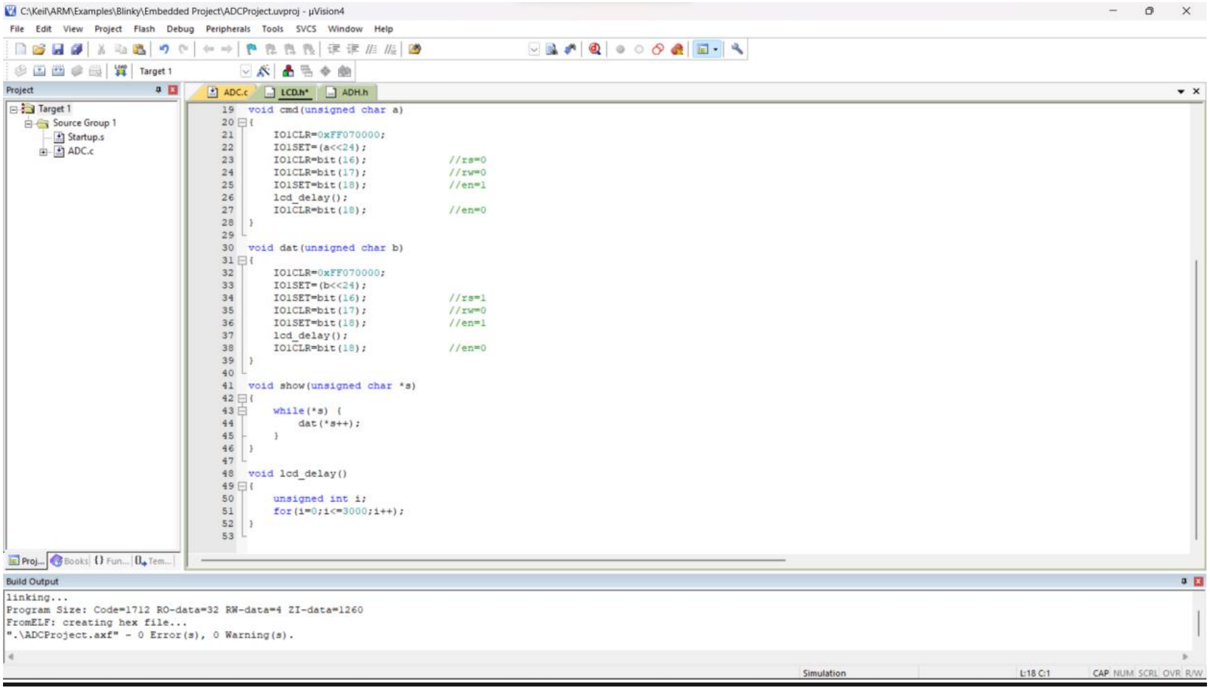
PROGRAM CODE

ADC.C

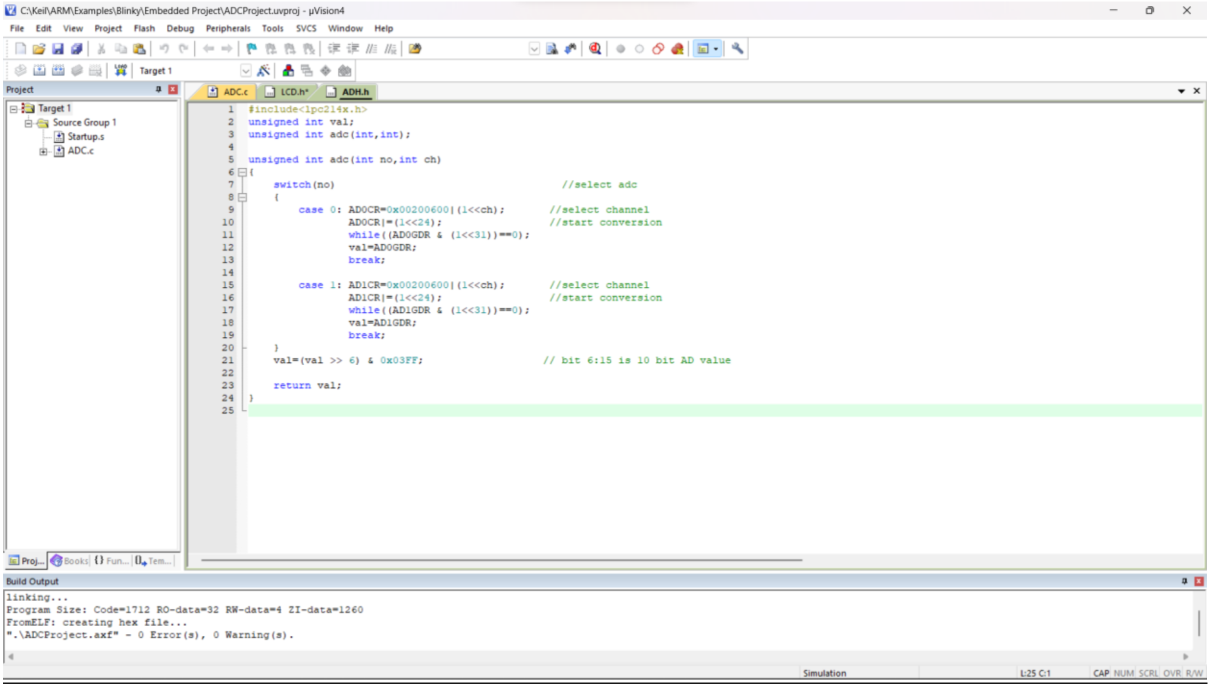


LCD.H

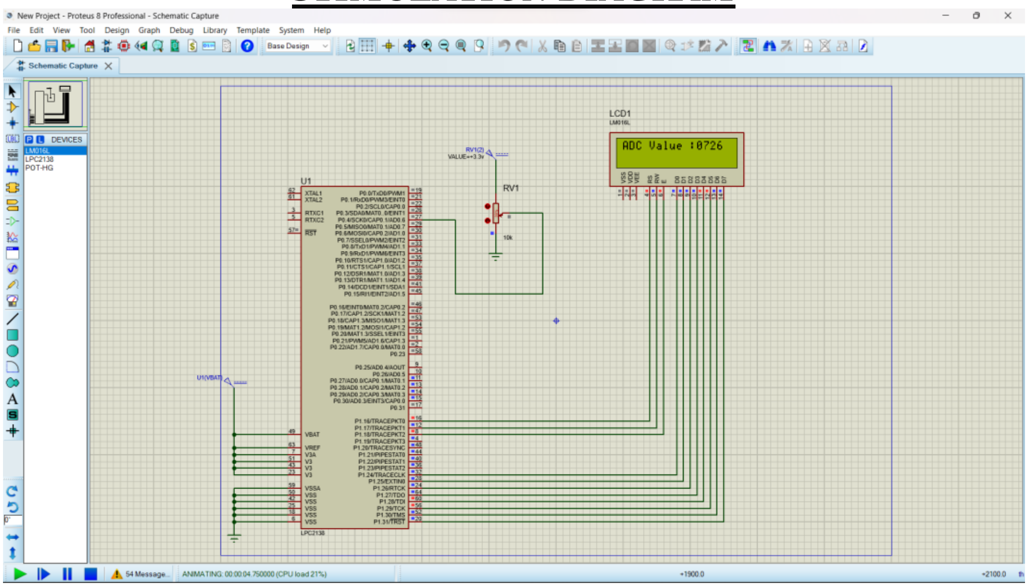

C:\Keil\ARM\Examples\Blinky\Embedded Project\ADCProject.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Target 1
Project
Target 1
Source Group 1
Startup.s
ADC.c
ADC.c
LCD.h*
ADH.h
#include<lpc214x.h>
#define bit(x) (1<<x)
void lcd_init(void);
void cmd(unsigned char a);
void dat(unsigned char b);
void show(unsigned char *s);
void lcd_delay(void);
void lcd_init()
{
cmd(0x38);
cmd(0x0e);
cmd(0x01);
cmd(0x06);
cmd(0x0c);
cmd(0x80);
}
void cmd(unsigned char a)
{
IO1CLR=0xFF070000;
IO1SET=(a<<24);
IO1CLR=bit(16); //rs=0
IO1CLR=bit(17); //rw=0
IO1SET=bit(18); //en=1
lcd_delay();
IO1CLR=bit(18); //en=0
}
void dat(unsigned char b)
{
IO1CLR=0xFF070000;
IO1SET=(b<<24);
IO1SET=bit(16); //rs=1
IO1CLR=bit(17); //rw=0
IO1SET=bit(18); //en=1
Build Output
linking...
Program Size: Code=1712 RO-data=32 RW-data=4 ZI-data=1260
FromELF: creating hex file...
".\ADCProject.axf" - 0 Error(s), 0 Warning(s).
Simulation
L:18 C:1
CAP NUM SCRL OVR R/W

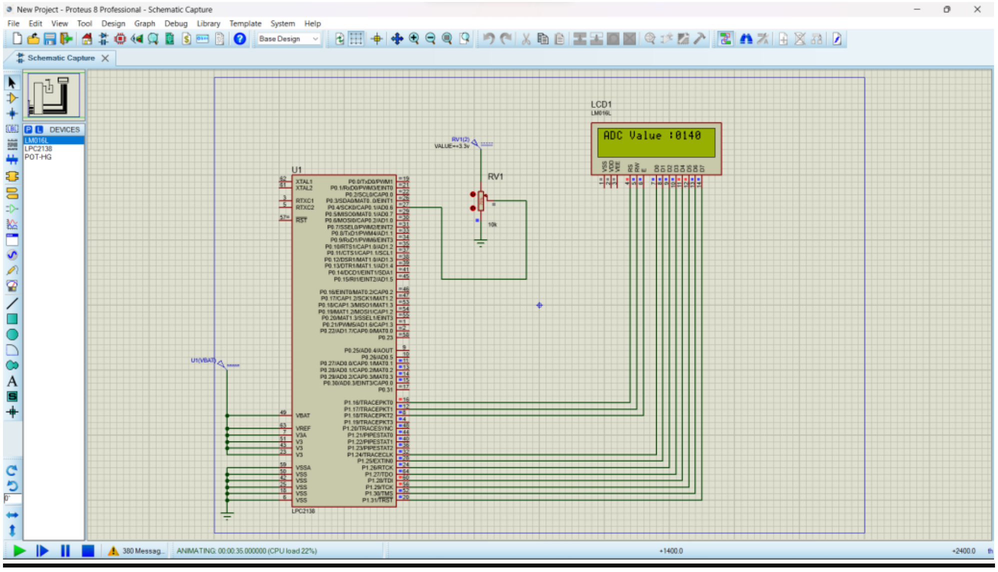

C:\Keil\ARM\Examples\Blinky\Embedded Project\ADCProject.uvproj - µVision4
File Edit View Project Flash Debug Peripherals Tools SVCS Window Help
Target 1
Project
Target 1
Source Group 1
Startup.s
ADC.c
ADC.c
LCD.h*
ADH.h
void cmd(unsigned char a)
{
IO1CLR=0xFF070000;
IO1SET=(a<<24);
IO1CLR=bit(16); //rs=0
IO1CLR=bit(17); //rw=0
IO1SET=bit(18); //en=1
lcd_delay();
IO1CLR=bit(18); //en=0
}
void dat(unsigned char b)
{
IO1CLR=0xFF070000;
IO1SET=(b<<24);
IO1SET=bit(16); //rs=1
IO1CLR=bit(17); //rw=0
IO1SET=bit(18); //en=1
lcd_delay();
IO1CLR=bit(18); //en=0
}
void show(unsigned char *s)
{
while(*s) {
dat(*s++);
}
}
void lcd_delay()
{
unsigned int i;
for(i=0;i<=3000;i++);
}
Build Output
linking...
Program Size: Code=1712 RO-data=32 RW-data=4 ZI-data=1260
FromELF: creating hex file...
".\ADCProject.axf" - 0 Error(s), 0 Warning(s).
Simulation
L:18 C:1
CAP NUM SCRL OVR R/W

ADH.H



STIMULATION DIAGRAM





ADC TYPES

The most common types of analog-to-digital converters available are:

1. Flash Analog to Digital converter.
2. Dual slope Analog to Digital converter.
3. Successive Approximation Analog to Digital Converter.

- **FLASH ADC**
  Flash ADC is one of the simplest ADCs. It is also known as the parallel ADC converter. It consists of a number of comparators. An encoder circuit is connected to the output of the comparators which gives us the binary output.

  Flash converter is the most efficient of all the converters in terms of speed. The number of comparators increases as the number of bits increases. We would require 7 comparators for 3-bit and 15 comparators for 4-bits. This is the weakness of flash ADC. A flash converter can produce a non-linear output which is an additional advantage. The voltage divider network consists of equal-value resistors which provide a proportional response. For special applications, the value of the resistors can be changed which will give a non-linear response.

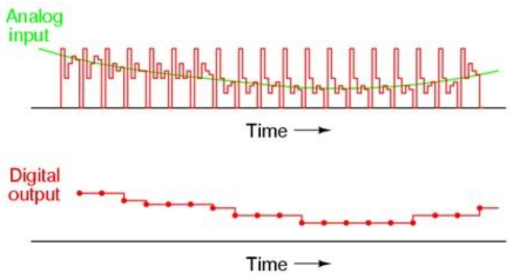
- **DUAL SLOPE ADC**
  A dual slope integrator first integrates and then disintegrates a voltage signal. It integrates an unknown voltage for a fixed time and disintegrates for variable time using a reference voltage.

- **DUAL-SLOPE INTEGRATION**
  The main advantage is that the error that occurs in a component during the integration is cancelled out during the phase of de-integration.

SUCCESSIVE APPROXIMATION ADC

This ADC does not count in the binary sequence, this register starts with the most significant bit and finishes at the least significant bit. The comparator's output is continuously monitored and compared with the analog signal input. This strategy gives much faster results.

The operation of this ADC can be observed in the following graph:



APPLICATIONS OF ADC

Some of the popular applications are:

- Audio applications:
  The music that we listen in our phone is stored in a digital form. The speaker accepts an electrical signal which is an analogue signal. Therefore, we need an ADC to convert a digital bit-stream of music into an analogue signal to play the music that we can hear through a mobile speaker.
- Call Receiver and Transmitter on mobile phone
- Video Streaming
- Data Acquisition

RESULT

We were able to stimulate the connection and produce output.

CONCLUSION

We use the method of successive approximation technique over here. The concept of digital to analogue-conversion has many applications. We have been using ADC every single day without realising it. We rely on ADC for even the very basic use of mobile phones. Hence ADC plays an important role in our day-to-day life.

REFERENCES

- **https://embetronicx.com/tutorials/microcontrollers/lpc2148/lpc2148-adc-tutorial/**
- **https://microcontrollerslab.com/analog-to-digital-adc-converter-working/**
- **https://en.wikipedia.org/wiki/Analog-to-digital_converter**