# EE2016 Microprocessor Lab & Theory July-Nov 2024

EE Department, IIT, Madras.

Experiment 8: ADC / DAC Implementation in ARM based LPC2148 Development Board (through C - Interface)

## 1 Aim

1. To understand C-interfacing (use C-programming) in an ARM platform
2. To study and implement ADC / DAC in ARM platform

## 2 Equipments, Hardwares / Softwares Required

The list of equipments, components required are:

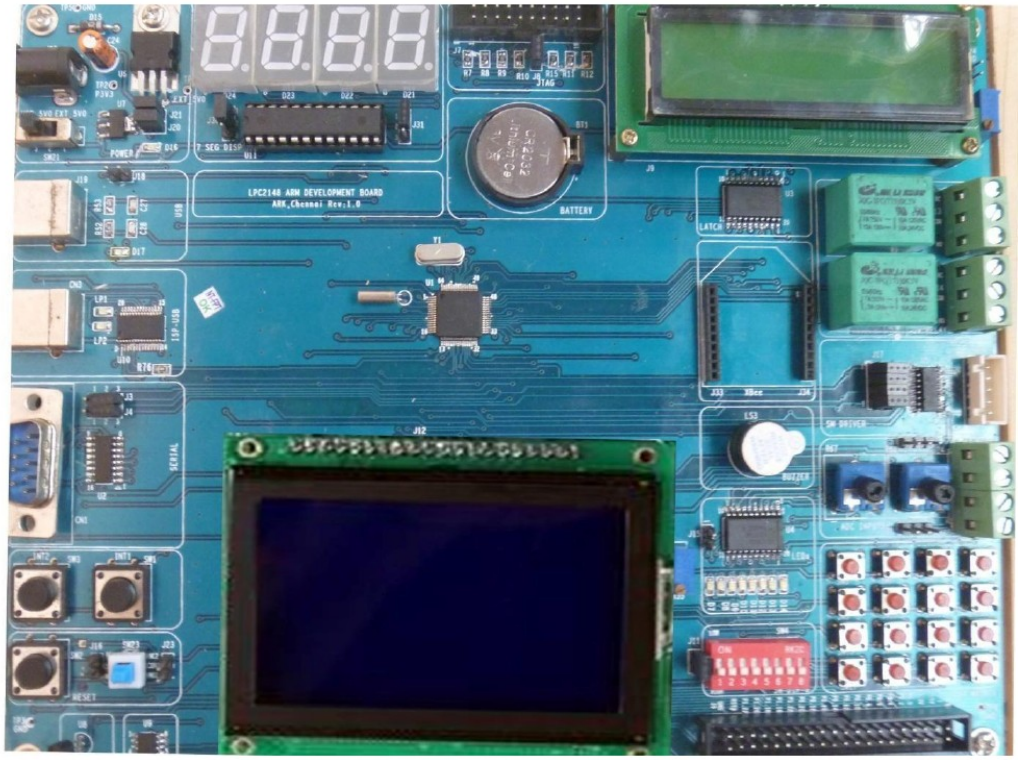
1. LPC2148 ARM Development board and accessaries
2. RS-232 cable
3. Keil microvision 5 (C - interface)
4. flash magic
5. Burn o-mat
6. DSO (Digital Storage Oscilloscope)
7. Sample programs for generating digital inputs. (For analog inputs, potentiometer is used)

## 3 Background Information

In this Section, we would discuss the following background information, which are very much essential to do the above experiment

1. ARK's LPC2148 ARM Development Board
2. ADC/ DAC Concepts
3. Specifications of ADC /DAC in LPC-2148 ARM microcontroller
4. Demo programs (which could be used to understand and write the C code for the tasks defined at the end).

## 3.1 ARK's ARM7 based LPC2148 Development Board - Anatomy



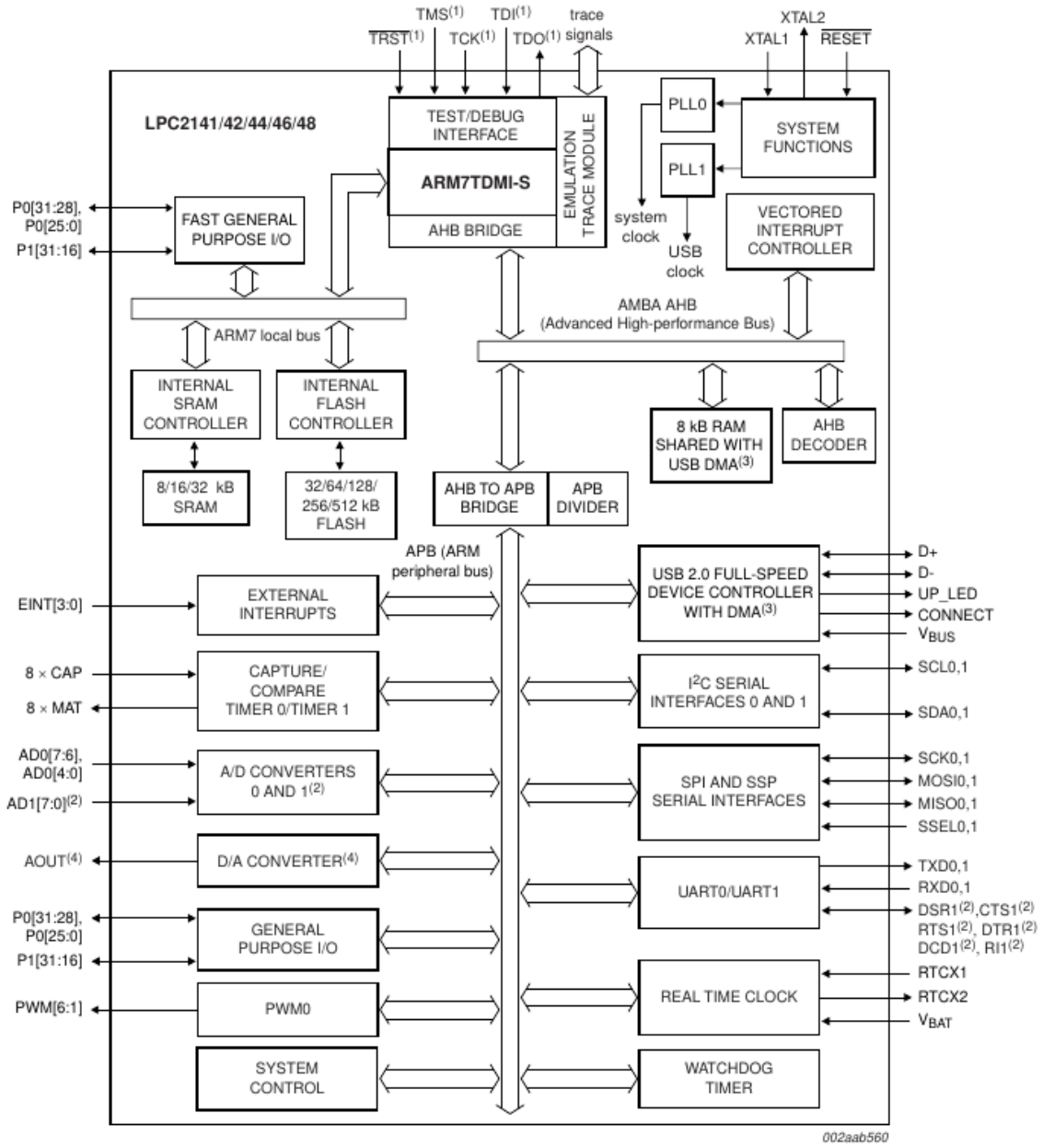

The microprocessor used in the microcontroller LPC2148 development board, ARM7TDMI-S core or CPU, whose expansion of the terminology ARM7TDMI-S is given below. LPC2148 is from NXP semiconductors while LPC stands for Low Power Consumption.

### 3.1.1 The ARM and ARM7TDMI-S

The ARM CPU core used inside the LPC2148 is called the ARM7TDMI-S (where T stands for Thumb Instructions, D for on-chip debugging support, M for multiplier, I for embedded In-circuit emulation hardware and S for the synthesizable option based on RTL provided). ARM7TDMI was the first of a range of processors introduced in 1995 by ARM Holdings, UK. ARM7TDMI is the first core to include the Thumb instruction set. The Thumb instruction set contains 16-bit instructions. These serve as a '*compact shorthand*' for a subset of the 32-bit instructions of the ARM. For 32-bit systems, ARM instructions are recommended for higher performance. Thumb instructions are appropriate for memory constrained systems.

### 3.1.2 Block Diagram of LPC2148

The LPC2148 block diagram is shown in Fig. below.




The instructions for the board to be used in the experiments are available as a separate pdf file.

### 3.1.3 Brief Description of the Software Environment

In this laboratory experiment, programming will be done in C. The instructions for writing C language programs in Keil µVision are as follows.

1. Open Keil uVision and Click project > New uVision Project
2. Select the 'Legacy device Database' under device dropdown. Search for LPC2148 and select it
3. Choose Yes when prompted to copy LPC2100.s Startup file
4. Right Click on Target 1, choose 'Options for Target 1'.
5. Under the Target tab tick 'Use Microlib'. Under Output tab tick 'Create Hex File'
6. Right click Source Group1 under Target in the left side of the keil window. Select Add new Item to group.
7. Select C file in the list, give a filename and save.
8. Write your program.

9. Go to Project tab and click on Build Target (or press F7).

### 3.1.4 How to Upload the Program to the Flash on LPC2148 in the Board

The instructions for uploading the program on to the flash memory on the LPC2148 are as follows.

1. Run FlashMagic.
2. Select LPC2148.
3. Choose the correct COM port to which the USB to Serial converter is connected. If you do not know the COM Port, use Device Manager to find out.
4. Set baud rate to 19200 and Oscillator to 12 MHz.
5. Choose the Hex file to be transferred.
6. On the board, put the "Program/Execute" slide Switch to "Program" mode (slide it down)
7. On FlashMagic click 'Start' to start programming.
8. Once programming is done, put the "Program/Execute" slide Switch to "Execute" mode (slide it up) and press reset to execute the program.

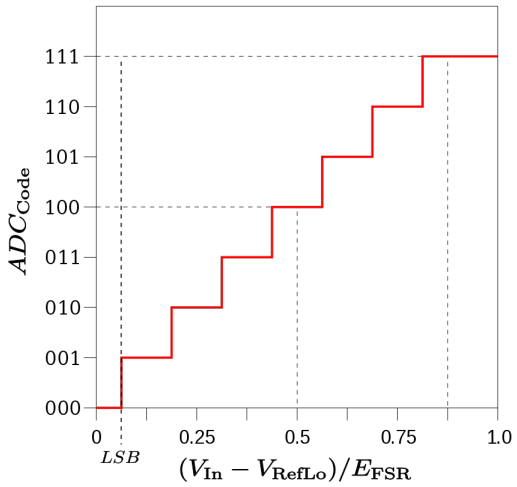
## 3.2 ADC / DAC: Concepts & their Specs in ARM-2378 Processor

Today virtually all of the userside equipments (communication, health care, IoT, automobile and most of industry) are digitized, while ALL the signals in nature are analog (both in its value and time of its evolution). [Strictly speaking, nothing in this world is digital, it is we humans have this convention of '*digital*' representation of signals with suitable transformation]. The engine which transforms (converts) an analog signal to digital format is called '*Analog to digital converter*' (ADC & vice versa is called as DAC).

Concepts from wiki:

### 3.2.1 ADC Theory

**Analog-to-Digital Converter (ADC) Definition** An ADC converts a continuous-time and continuous-amplitude analog signal to a discrete-time and discrete-amplitude digital signal. Note that the discretizations happens in both the domains. In the time domain, it results in (a) 'sampling' the signal, (b) the above sampling interval allows the ADC to do the conversion (before it is ready for the next sample), while in the amplitude domain, this mapping (conversion) involves quantization of the input, so it necessarily introduces a small amount of error or noise. The limitation on the ADC for conversion time limits the allowable bandwidth of the input signal (for this one needs to understand the Nyquist's sampling theorem).




The performance of an ADC is primarily characterized by its bandwidth and signal-to-noise ratio (SNR). The bandwidth of an ADC is characterized primarily by its sampling rate. The SNR of an ADC is influenced by many factors, including the resolution, linearity and accuracy (how well the quantization levels match the true analog signal), aliasing and jitter. The SNR of an ADC is often summarized in terms of its effective number of bits (ENOB), the number of bits of each measure it returns that are on average not noise. An ideal ADC has an ENOB equal to its resolution. ADCs are chosen to match the bandwidth and required SNR of the signal to be digitized. If an ADC operates at a sampling rate greater than twice the bandwidth of the signal, then per the Nyquist–Shannon sampling theorem, perfect reconstruction is possible.

The presence of quantization error limits the SNR of even an ideal ADC. However, if the SNR of the ADC exceeds that of the input signal, its effects may be neglected resulting in an essentially perfect digital representation of the analog input signal.

**Quantization Noise** Quantization error is the noise introduced by quantization in an ideal ADC. It is a rounding error between the analog input voltage to the ADC and the output digitized value. The noise is non-linear and signal-dependent. In an ideal analog-to-digital converter, where the quantization error is uniformly distributed between $-1/2$ LSB and $+1/2$ LSB, and the signal has a uniform distribution covering all quantization levels, the Signal-to-quantization-noise ratio (SQNR) can be calculated from

SQNR $= 20\log_{10}(2^Q) \approx 6.02 \cdot Q$ dB [2] where Q is the number of quantization bits.

#### 3.2.2 DAC Theory

A DAC converts an abstract finite-precision number (usually a fixed-point binary number) into a physical quantity (e.g., a voltage or a pressure). In particular, DACs are often used to convert finite-precision time series data to a continually varying physical signal.

The performance of the DAC largely depends on its counterpart ADC. Both contributed to digital revolution, as it made possible, the interfacing of physical quantities in the real world (basic sciences and engineering) with computers which are basically digital.

# 4 Tasks

## 4.1 ADC

Given a real-time (analog) signal from a sensor, convert it into a digital signal (Implement an ADC). Decrease the step size? Do you see any change in the bits used to represent the whole range? What is the quantization error?

## 4.2 DAC

Given the LPC2148 ARM development board and given program for square wave, modify it to generate

1. RAMP (saw tooth) wave
2. Triangular wave
3. Sine wave (using lookup table or using formulae)

# 5 Demo Program

## 5.1 ADC / DAC

The c-code has been uploaded in course website in moodle.

# 6 Procedure

Analog to digital (ADC) & DAC Converter

1. Write a C program, which does the serial transmission of frames (a group of contiguous bits) to the PC
2. Edit the above program file in Keil software. In Keil software one can edit, recompile and run etc.
3. Compile it in Keil platform
4. Dump it in ViARM 2378
5. Connect the serial cable from ViARM 2378 and PC
6. Run the program on the development kit by resetting

# 7 Results

1. Run the program and ask the TA to see the output
2. Take a snapshot using your mobile and make a report