ITU Control & Automation Eng. Dept.
KON309E Microcontroller Systems
Experiment 4: ADC and USART

Aim: Using the USART to report periodic analog voltage samples to the PC.

In this experiment, we will take sequence samples from the analog to digital converter on two channels, and report the values back to the PC

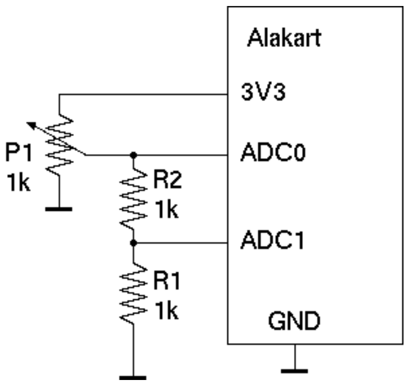
For this experiment, you will need to consult the usual reference documents:

- LPC824 user manual
- LPC82x datasheet
- Alakart schematic diagram

Write the code using the peripheral support libraries (Xpressso SDK) or direct register programming or a combination of both.

PART I

Construct the circuit as shown in the Figure. Check your wiring and make sure that **no connection is made to the 5V supply**.



- The ADC peripheral should be configured to sample CH0 and CH1 **using Sequence A**.
- The serial port USART0 should be configured for a baud rate of 38400 and as a serial terminal.

Write a program that will do the following in the main loop:

1. Prints out a message such as: "Press a key to start conversion."
2. Starts the ADC Sequence A when a key is pressed from the PC keyboard
3. Waits until one conversion on Sequence A is complete.
4. Converts the results to voltages on the two channels using the reference voltage of 3V3. (Determine the resolution of the ADC by consulting the reference manual. Use an `int16_t`, and represent the voltage in millivolts, as the terminal program cannot print floating point numbers.)
5. Prints out the result of the conversion in the format : "ADC0=xxxx mV, ADC1=yyyy mV"
6. Repeat.

Remember that the terminal program in the PC must also be set to the **same baud rate as Alakart**. Otherwise, garbled output, or even, no output will be seen.

Write in your report:

- What values must be written to which registers so that ADC Sequence A can be configured to sample CH0 and CH1.
- How the serial port is set to 38400 Baud.
- The frequencies of "main clock" and "system clock" by checking the configuration functions. Write which functions you check and how you find the result.

Store the code for this part in a folder named "EX4_PART1".

PART II

In PART I, we built an analog to digital converter that samples Sequence A once when a key is pressed on your keyboard. In PART II, we will make the sampling triggered by the SCT0 timer peripheral, **in hardware** and get the result using an INT. This is typically how it is implemented in consumer products.

Keep the same circuit as in PART I but add a LED to a suitable PIO0 pin with a resistor. This time the following peripherals must be set:

- ADC as in PART I with "Sequence A" set to sample ADC0 and ADC1. However, the ADC must be configured to be triggered from SCT0, OUT 3
- Serial port USART0 terminal as in PART I, but with Baud rate 57600 Baud.
- SCT0 set as an event generator and its output 3 changes at end of count event. The duration of SCT0 must be calculated and set so that the ADC is triggered every 500ms. Remember that system clock is **configured to 24MHz** in the projects provided with the serial port examples and not 30MHz.
- "ADC conversion complete" INT is enabled and configured.
- PIO0 is configured such that the pin where the LED is connected is an output and the LED is turned OFF.

Write a program that will do the following:

1. ADC Sequence A is triggered for conversion by the SCT0 timer every 500ms (for 2 conversion sequences / second).
2. At the completion of the ADC sequence, an end of conversion INT is triggered.
3. Within the ISR: ADC results are stored in a structure, a global volatile variable is set to True and the LED is toggled.
4. Within the main loop of the program: The global variable is repetitively queried (checked) to see if it has been set to True by the ISR. If it has been set to True, then: The ADC results are converted to actual voltages in mV,

and printed in the same format as in PART I, and (very important!:) the global variable is set back to False.

As a result, when the program is ran, we should see the LED flashing at 1s, and the result of the conversion appears on the screen in millivots every 500ms. **Note that** the conversion result is retrieved from the ADC within the ISR, but it is printed out in the main loop. The ISR must signal the main loop that the conversion is complete using a global variable that is called a "flag".

Use a chronometer to check that you really get the correct timing. (Time 20 blinks and make sure it is close to 20s)

Write in your report:

1. How the SCT0 is configured so that it can trigger the ADC every 500ms. Provide the equation for the calculation by consulting the reference manual. What values must be written to which registers?

2. Where is the INT vector defined? (Write the name of the file and copy the line where it is defined, to your report)

3. Describe your program and show how each software function and hardware peripheral interacts with the others.

Store the code for this part in a folder named "EX4_PART2".

PART III: Build an oscilloscope

This is the fun part and not compulsory. See the program "SerialPlot" by Hasan Yavuz Ozderya (See: https://hasanyavuz.ozderya.net/?p=244). It plots data coming from the serial port onto the screen in a graph format, similar to an oscilloscope. It can plot data that comes from the serial port in human readable form, such as data that is ordered like this:

3425, 653

3423, 654

3426, 655

etc.

Change the sampling time to **10ms** by adjusting SCT0 registers, re-configure your code, install Serial Plot, and configure it to display your measurements as a time graph.

If you implement this part, describe in your report what you intended to do and how you implemented the idea.

You do not need to provide a code for this part; it will be evaluated by the class assistant and fellow students.

A bonus of +5 points will be given.

Note: Please pay attention to the folowing:

- **Insert Alakart into the breadboard also** when preparing your circuit.
- Connect V_{DD} (3.3V) and **Ground** pins of the microcontroller to the breadboard’s (**+**) and (**–**) sockets via jumpers.

Report Content

Your report must be a formal account of what you did in the experiment. Make sure that it includes the following items:

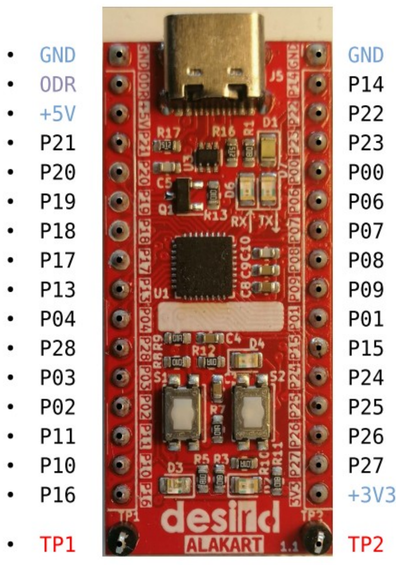
- How you have configured the processor i.e.:
  - What pins are inputs, what pins are outputs, etc.
  - Which peripheral devices were explicitly powered up,
  - How the GPIO and other peripherals were configured,
  - What values were written to which registers,
  - Which default settings of the peripherals were used.

These must appear in all of your reports.

- Propose a SCT0 counter value and **show how you calculate the "prescaler" and "match register" values for that frequency**.
- **Any inclusion of code must be as formatted text.**
(Photographs or screenshots are not accepted!)
- What problems you have encountered and how you solved them.

Appendix1: Pin connections of components on Alakart

- LEDs:
  - RED: D4 on GPIO PIN12
  - Blue: D3 on GPIO PIN16
  - Green: D2 on GPIO PIN27
  - White: D1 Power on
  - Green: D6 Transmit to PC
  - Red: D7 Transmit from PC
- Buttons:
  - S1: Reset
  - S2: ISP (enter boot mode) Also: User button.
- Red Pins: Test
  - TP1: GPIO PIN16
  - TP2: GPIO PIN27
- Purple Pin: Open drain FET
  - ODR: GPIO PIN21



Note:

- PIO0_10, PIO0_11 are open drain pins. Research what this means for the experiment.
- PIO0_2, and PIO0_3 are debugger pins by default. PinPIO0_5 is Reset pin by default. You can use them if you wish, but will **need to disable their default functionality in PINENABLE0 register** first.