

8. Analog to digital converters

There are three 12-bit analog to digital converter (ADC) blocks available. The conversion time for each ADC block is about 1 μ s. There is a rich variety of possible ADC configurations, and here we test the simplest one to measure two voltages simultaneously using two ADCs.

A simplified block diagram for one ADC block is given in Fig. 1 (excerpt from Fig. 28, RM0090, page 208). The complete block includes one more ADC with similar features and additional settings.

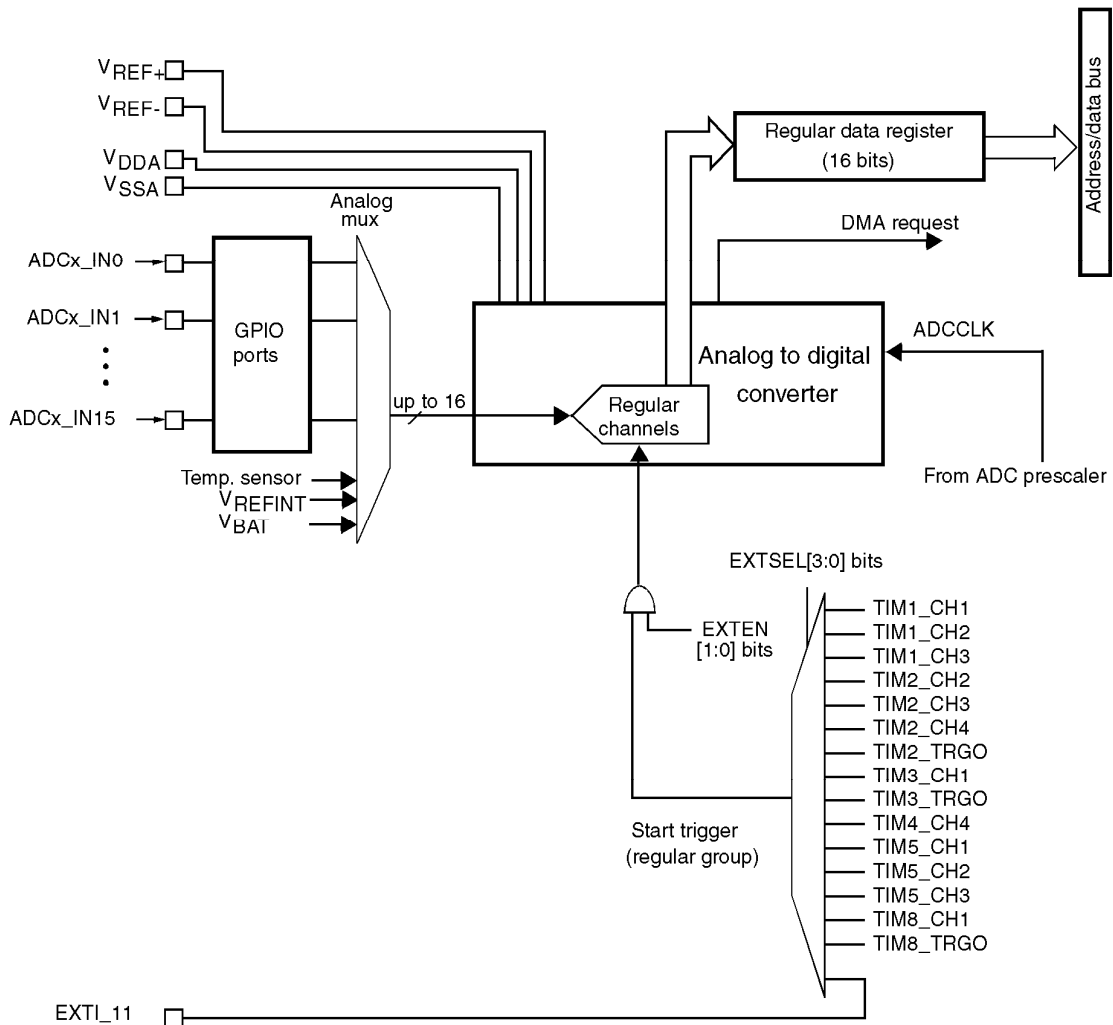


Figure 1: A block diagram of a single ADC

The actual ADC is located in the center of the figure. Its operation is supported by power supply lines and the reference lines, all coming from pins of the microcontroller on the top left. Additionally,

a clock signal is required. The analog input of the ADC is fed by the output of the multiplexer (analog mux). The inputs to the multiplexer are the IO pins of the microcontroller; there are altogether 16 pins available to input analog signals, and the multiplexer houses additional three inputs to allow the measurement of the chip temperature, power and reference lines. Appropriate input pin is selected by sets of bits (five of them) in ADC regular sequence register (ADCx_SQR3), lower five bits. The ADC writes the result of conversion into a regular data register (ADCx_DR). Here x stands for the index of the ADC used (1, 2 or 3). It should be noted that analog signals do not mix with digital signals well. It is compulsory to turn off the digital circuitry for the pin designated to be used for analog signals, or analog signals will be hampered. This can be done by setting bits in register MODE for ports.

The ADC block includes hardware to support up to 16 consecutive conversions at pins stated in advance; when one intends to implement consecutive conversion mode then the number of consecutive conversions and the desired multiplexer settings must be stated in registers ADCx_SQR1 and ADCx_SQR2. However, there is only one register to hold result of a conversion, so a user program or DMA (direct memory access) hardware must be set in order to move consecutive results from the data register before next conversion is finished.

The parameters of conversion can be set by manipulating bits in registers ADCx_CR1 and ADCx_CR2, while sampling time for each conversion in the sequence is individually settable by bits in the register ADCx_SMPR1 and ADCx_SMPR2. Also, the conversion can be started by various signals coming from either hardware of the microcontroller or software, and is selected by multiplexer at the bottom of Fig. 1. See details in RM0090, chapter 11.

The demo program has been prepared, and its listing is given in Fig. 2. The program uses ADC1 and ADC2 to measure voltages at pins PA2 and PA3. It starts with the initialization of both ADCs, port A and LCD screen, and then continues into the infinite loop. Within the loop a software command first starts the measurement, and then some time is wasted in order to wait for the results of conversion. Finally, results from both ADCs are copied to the LCD.

Some initialization is common to all ADCs, the corresponding bits are located in the ADC common control register (ADC_CCR), other initialization is individual to each ADC, and corresponding bits are located in separate registers for each ADC.

The demo program uses two ADCs in simultaneous mode; therefore both ADCs are forced to start conversions following one software command. This option is called “Regular simultaneous mode only”, and is marked by a string of bits 00110 located at lower five bits of ADC common control register (ADC_CCR, common register for all three ADCs). The clock for all ADCs is left at the default value, no pre-scaling is activated.

The sampling time for the input signal is also left at the default value and is the shortest possible lasting three ADC clock cycles; if required the sampling time can be changed by writing a combination of three bits per conversion in a sequence into register ADCx_SMPR1 and ADCx_SMPR2 for each ADC separately.

```

#include "stm32f4xx.h"
#include "LCD2x16.c"

/* inputs are PA2 & PA3 -> must be enabled for analog, PA must have clock! */

int main () {
unsigned int j;

/*-----*/
/* Software triggered, dual, simultaneous sampling */
/*-----*/

RCC->APB2ENR |= 0x00000100 + 0x00000200; // Enable clock for ADC1 & ADC2
//RCC->AHB1ENR |= 0x00000001 + 0x00000002; // Enable clock for GPIOA & GPIOB

ADC->CCR      = 0x00000006; // No DMA, Regular simultaneous mode only

ADC1->CR2     = 0x00000001; // Control Register 2: ADC1 ON
ADC1->SQR3    = 0x00000002; // regular Sequence Register 3

ADC2->CR2     = 0x00000001; // Control Register 2: ADC2 ON
ADC2->SQR3    = 0x00000003; // regular Sequence Register 3

GPIOA->MODER  |= 0x000000f0; // MODE Register -> PA2, PA3 are analog inputs

LCD_init();
LCD_string("ADC1=", 0);
LCD_string("ADC2=", 0x40);

while (1) {
    ADC1->CR2 |= ADC_CR2_SWSTART; // simultaneous Start Conversion
    for (j = 0; j<100; j++){}; // wait for conversion to complete
    LCD_Int16((int)ADC1->DR, 0x06, 1); // write result on LCD, 1st line
    LCD_Int16((int)ADC2->DR, 0x46, 1); // write result on LCD, 2nd line
};
}

```

Figure 2: A listing of the program to utilize both ADCs