Laboratory 4: Analog-to-digital converter (ADC) and Pulse-width modulation (PWM)

Objectives

1. Understand the concept of Digital-to-analog converter (ADC)
2. Understand the concept of Pulse-width modulation (PWM)

Digital-to-analog converter (ADC)

An analog (aka. Analog signal) is continuous signal. Unlink a digital signal which observed values are either HIGH or LOW, analog signal give an approximate level of value (e.g. 2.3 volts). In real life, everything is analog i.e. temperature (degree Celsius), brightness (Lux), sound (decibel), pressure (pascal), and speed (km/s)

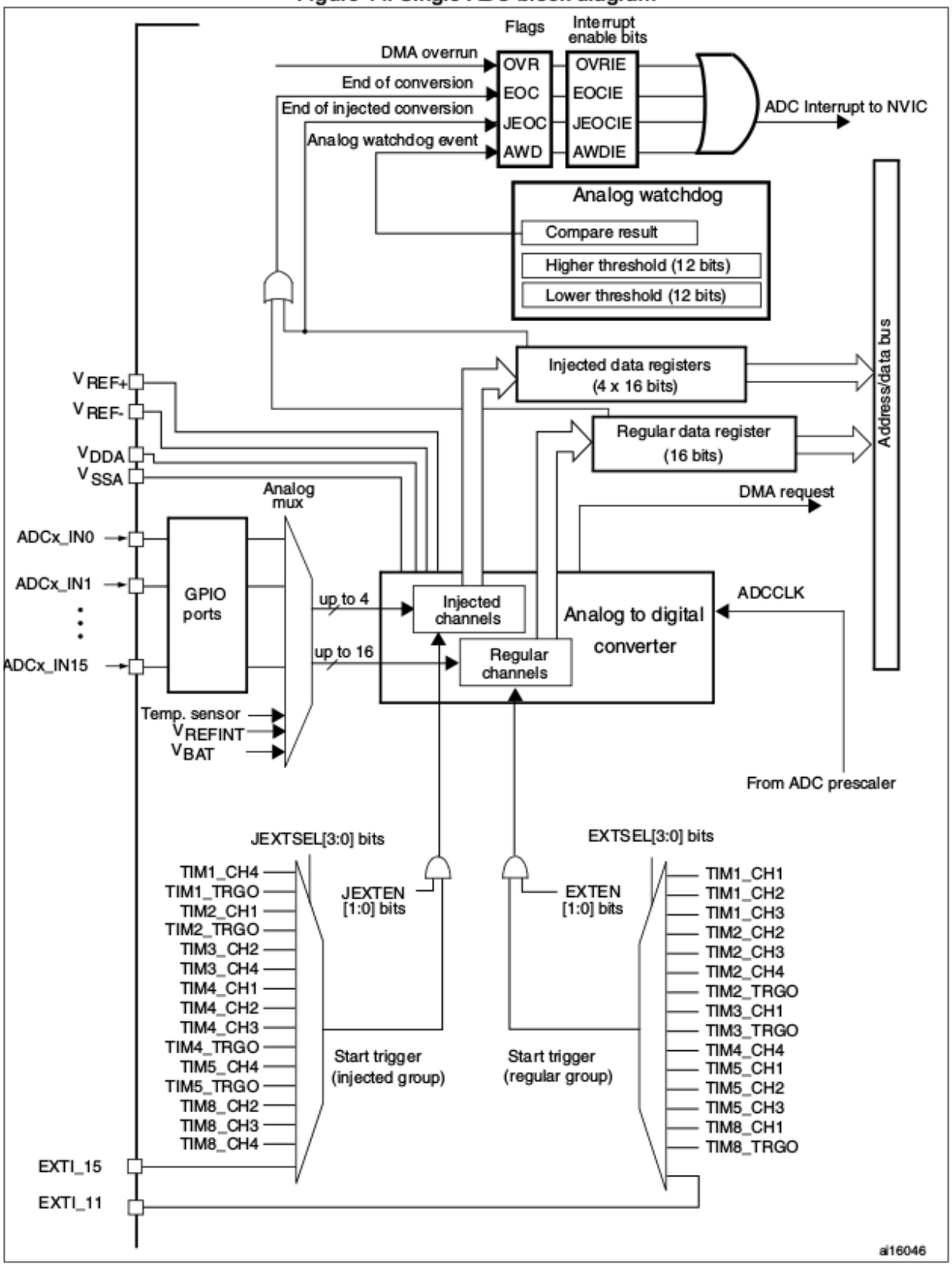
Since a computer is a digital device, any analog signal has be first converted to digital signal with a special device namely Analog-to-Digital converter (ADC). The resolution of the converter indicates the number of discrete values it can produce over the range of analog values. Thus, when we observed a value from ADC, the real voltage is a faction of value read multiplied by the reference voltage. For example, a 10-bit ADC can give a value between 0 and 1023. If a reference voltage is 3.3 volts, a 512 from ADC means 3.3*512/1023 volts.

STM32F4xx family's ADC

The 12-bit ADC is a successive approximation analog-to-digital converter. It has up to 19 multiplexed channels allowing it to measure signals from 16 external sources, two internal sources, and the V_{BAT} channel. The A/D conversion of the channels can be performed in single, continuous, scan or discontinuous mode. The result of the ADC is stored into a left- or right-aligned 16-bit data register.

The analog watchdog feature allows the application to detect if the input voltage goes beyond the user-defined, higher or lower thresholds.

Figure 44. Single ADC block diagram



Source : **Reference Manuals – RM0090**: STM32F405/415, STM32F407/417, STM32F427/437 and STM32F429/439 advanced ARM®-based 32-bit MCUs

Photoresistor or light-dependent resistor (LDR)

In this laboratory, we also use LDR (together with potentiometer) as an analog signal. "LDR is a light-controlled variable resistor. The resistance of a photoresistor decreases with increasing incident light intensity; in other words, it exhibits photoconductivity. A photoresistor can be applied in light-sensitive detector circuits, and light- and dark-activated switching circuits." definition taken from wikipedia.org

Pulse-width modulation (PWM)

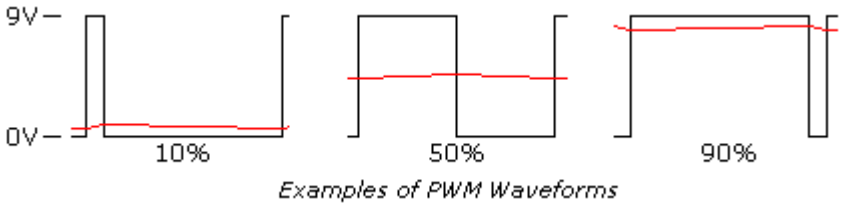
Pulse-Width Modulation allows microcontroller to control the duty cycle of a pulse. It can be used to controller the brightness of LED, speed of motor, or any analog

signal.

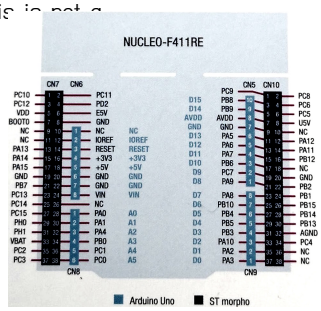
PWM can have many of the characteristics of an analog control system, in that the digital signal can be free wheeling. PWM does not have to capture data, although there are exceptions to this with higher end controllers.

One of the parameters of any square wave is duty cycle. Most square waves are 50%, this is the norm when discussing them, but they don't have to be symmetrical. The ON time can be varied completely between signal being off to being fully on, 0% to 100%, and all ranges between.

Shown below are examples of a 10%, 50%, and 90% duty cycle. While the frequency is the same for each, this is not a requirement.



Examples of PWM Waveforms



The reason PWM is popular is simple. Many loads, such as resistors, integrate the power into a number matching the percentage. Conversion into its analog equivalent value is straightforward. LEDs are very nonlinear in their response to current, give an LED half its rated current you you still get more than half the light the LED can produce. With PWM the light level produced by the LED is very linear. Motors, which will be covered later, are also very responsive to PWM.

Source: http://www.allaboutcircuits.com/textbook/semiconductors/chpt-11/pulse-width-modulation/

Lab Exercises

1 Create a new STM32 Project on System Workbench IDE to DIM a red LED (LD5) by increasing the duty cycle from 0% duty cycle by 1% every 0.01 second. When the PWM reaches the 100% duty cycle, decrease the duty cycle by 1% every 0.01 second to 0% duty cycle. Repeat the following step forever. (use 100 microseconds period for PWM)

2 Connect LDR to STM32F4Discovery Board and create a new STM32 Project on System Workbench IDE to read the value from LDR and display to serial terminal via UART.

3 Create a new STM32 Project on System Workbench IDE to control a red LED (LD5) brightness with PWM from the value of LDR with this equation.

$$\text{PWM Duty Cycle} = (1 - (C / P)) * 100\ \%$$

C = (max environment brightness) – (current environment brightness)

P = (max environment brightness) - (min environment brightness)