Study on GPIO, ADC and HAL

WHAT IS A GPIO PIN?

GPIO stands for general purpose input/output. It is a type of pin found on an integrated circuit that does not have a specific function. While most pins have a dedicated purpose, such as sending a signal to a certain component, the function of a GPIO pin is customizable and can be controlled by the software.

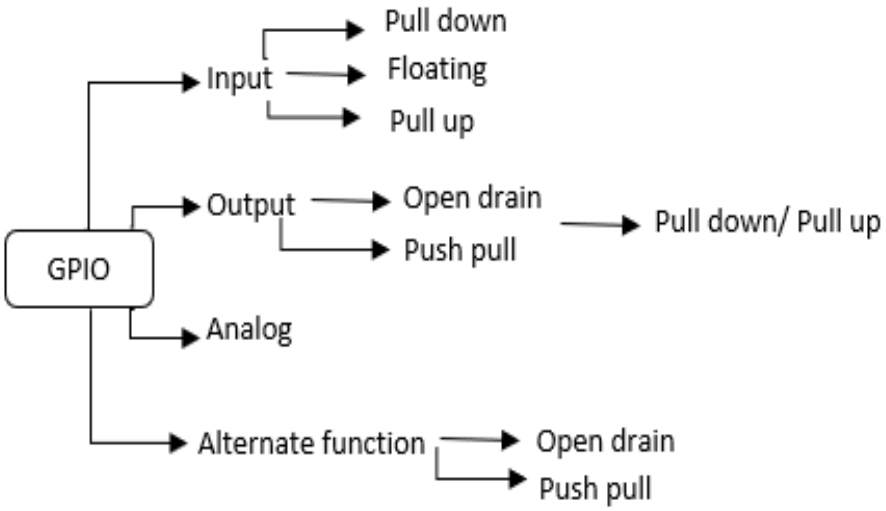
Usually, GPIO pins can have any of the 4 functions: -

1. Input Function
2. Output Function
3. Analog Function
4. Alternate Function

Input Pins have pull-up, pull-down and floating forms. Pull-up and Pull-down input pins are connected internally to pull-up and pull-down resistors which ensures the GPIO to always be in a valid state. This is especially needed when the GPIO input voltage is low, or a high impedance state which needs to be offset. Floating Pins are unconnected pins which are normally used when external pull-up/pull-down resistors are needed.

Output Pins have push-pull and open drain type pins. Push-pull pins are those which are connected internally to both a pmos and a nmos which can either pull down or pull up the logic level depending on the use. This can be completely controlled by GPIOx_ODR function and be partially controlled by the GPIOx_BSSR function by giving it an operation range. Open drain type pins aren't connected to pmos, instead it needs to be connected to an external pull-up resistor which will do the pmos's task. The pull-up resistor can also be internal of a value 40k ohms and can be activated with the GPIOx_PURPDR function.

Analog Pins gives you a port pin with no pullup, pull down or Schmit trigger function.



WHAT IS AN ADC PIN?

The analog-to-digital converters inside STM32 products allow the microcontroller to accept an analog value like sensor output and convert the signal into the digital domain. The ADCs can convert signals into 5 mega samples per second in 12-bit mode.

There are also several different triggering methods. To offload the CPU, the ADC has an analog watchdog for monitoring thresholds. The ADC also offers oversampling to extend the number of bits presented in the final conversion value. For power-sensitive applications, the ADC offers several low-power features.

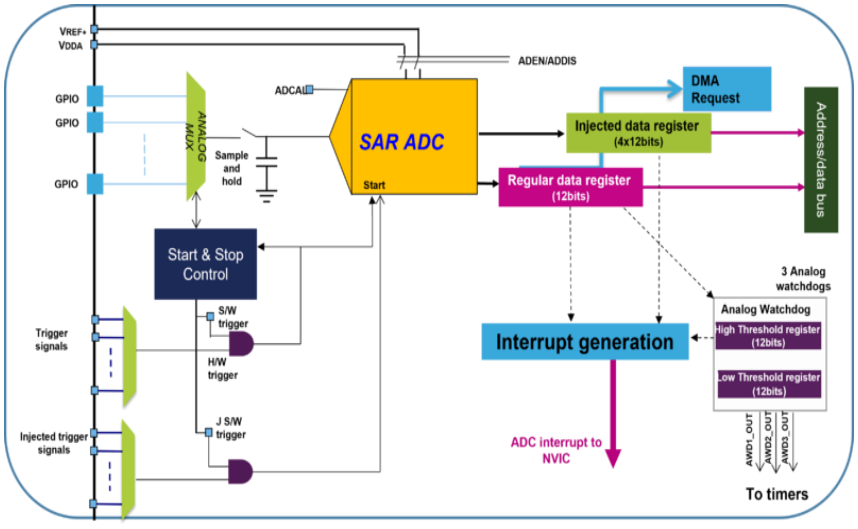
Auto-delayed mode makes the ADC wait until the last conversion data is read before starting the next conversion. This avoids unnecessary conversions and thus reduces power consumption. Power consumption is a function of the sampling frequency. For low sampling rates, the current consumption is reduced almost proportionally.

The sampling time can be programmed individually for each input channel of the analog-to-digital converters. The sampling times listed in this slide in ADC clock cycles are available. Longer sampling times ensure that signals with a higher impedance are correctly sampled and converted.

The ADC supports several conversion modes:

1. **Single mode**, which converts only one channel, in Single-shot or Continuous mode.
2. **Scan mode**, which converts a complete set of predefined programmed input channels, in Single-shot or Continuous mode.
3. **Discontinuous mode**, which converts only a single channel at each trigger signal from the list of pre-defined programmed input channels.

The ADCs support hardware oversampling. They can sample 2 to 256 times without CPU support. The converted data is accumulated in a register and the output can be processed by the data shifter and the truncator. 12-bit data can be extended to be presented as a 16-bit data register. This functionality can be used as an averaging function or for data rate reduction and signal-to-noise ratio improvement as well as for basic filtering.



WHAT IS THE DIFFERENCE BETWEEN ADC AND GPIO ANALOG?

ADC, Analog Digital Converter. A single voltage sample through successive approximation becomes represented by a 12-bit binary value that can be manipulated. ADC = binary representation of sampled voltage.

The GPIO analogue option is not to be confused with an ADC input, but in principle has two functions:

1. Disconnecting the external pin from the digital function. This is useful, for example, to disconnect the Schmitt trigger present at the digital input from a floating input, because this can lead to high frequency switching of this trigger with associated current consumption.
2. Connecting any analogue peripherals (e.g. opamp, comparator, ADC) that may be present, but do not necessarily have to be in the respective case.

A GPIO, Analog input is usable as an analog input or output pin. This alternative configuration mode does not have an ADC or DAC hooked up to the pin. GPIO Analog puts the pin in a low power state while an ADC_IN0 pin will allow you to rank the channel for ADC conversion.

WHAT IS THE HAL LIBRARY?

HAL is the abbreviation of Hardware Abstraction Layer

The HAL library is an abstraction layer embedded software launched by ST for STM32, which can better ensure the maximum portability of STM32 products. The library provides a consistent set of middleware components such as RTOS, USB, TCP/IP and graphics. The HAL library is open-source code released under a non-restrictive BSD license (Berkeley Software Distribution).

Some examples in STM32 where HAL is used:

1. Hardware Event: A UART peripheral receives a byte of data, triggering a UART interrupt.
2. Interrupt Function Execution: The UART ISR (USARTx_IRQHandler) is automatically executed. This function is part of the STM32's interrupt handling and defined in the startup file or the HAL driver.
3. HAL Handling: Within the ISR, the HAL processes the interrupt, reads the received byte to clear the interrupt flag, and performs any other necessary hardware-level processing.
4. Callback Invocation: After handling the hardware-level events, the HAL then calls a user-defined callback function, such as HAL_UART_RxCpltCallback(), if reception was completed.
5. User Application Response: Inside this callback function, which you have defined in your application, you can write code to process the received byte. This processing happens outside the ‘real-time’ constraints of the ISR.