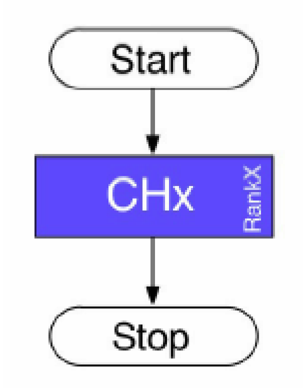
산업기술의 최고 명문대학

한국산업기술대학교 메카트로닉스공학과

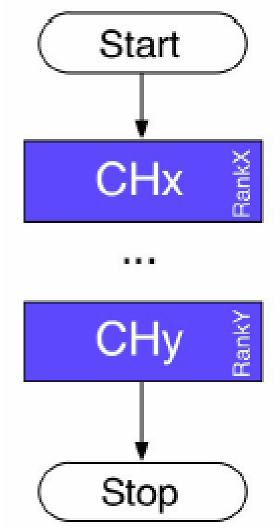
메카트로닉스공학과 교수 이양희

Analog Digital Converter(ADC)

- 1) Single Channel, Single Conversion :
  - 한개의 Channel 에 한번만 Conversion 을 수행

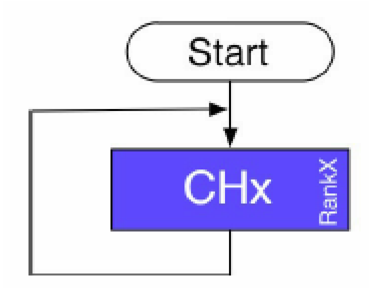


- 2) Scan Channel, Single Conversion
  - 다수의 Channel 에 한번만 Conversion 을 수행하는 경우

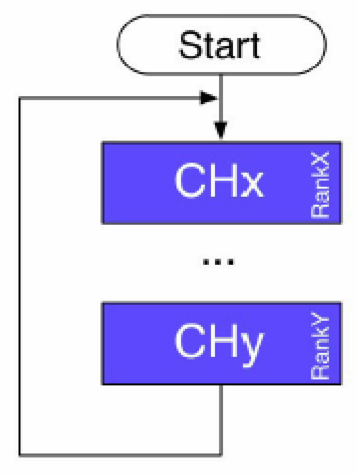


https://m.blog.naver.com/PostView.nhn?blogId=eziya76&logNo=221472568302&targetKeyword=&targetRecommendationCode=1

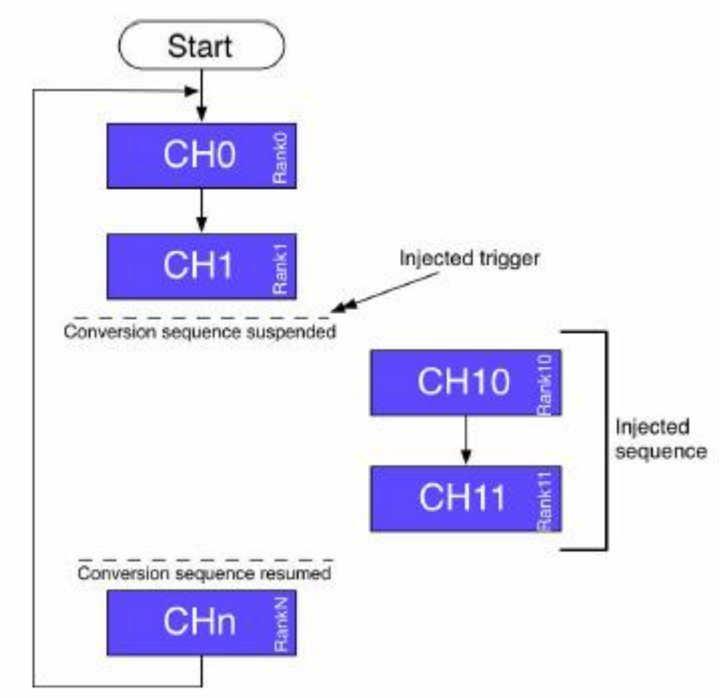
- 3) 한개의 Channel 에 반복 Conversion 을 수행하는 경우 (Single Channel, Continuous Conversion)



- 4) Scan Channel, Continuous Conversion
  - 다수의 Channel 에 반복 Conversion 을 수행하는 경우



- 다수 Channel 에 대해서 ADC 를 수행하는 경우 각 Channel 간 우선 순위는 Rank 값에 의해서 결정됨
- STM32 는 Regular Group 과 Inject Group 의 개념을 적용하고 있음.
- Regular Group 은 최대 16개의 Channel 로 구성되며, Injected Group 은 최대 4개의 Channel 로 구성될 수 있음.
- Injected Group 의 우선순위가 높기 때문에 Regular Group 의 ADC 수행 중 Injected Group ADC 요청이 발생하면 Regular Group 은 선점되고 Injected Group 의 ADC 가 우선 수행된 후 재개 됨.



ADC의 구조와 기능

AD 변환기의 구조

개요

- This section applies to the whole STM32F4xx family, unless otherwise specified.
- **ADC introduction**
  - The 12-bit ADC is a successive approximation analog-to-digital converter.
  - 19 multiplexed channels : 16 external sources + two internal sources +VBAT 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.

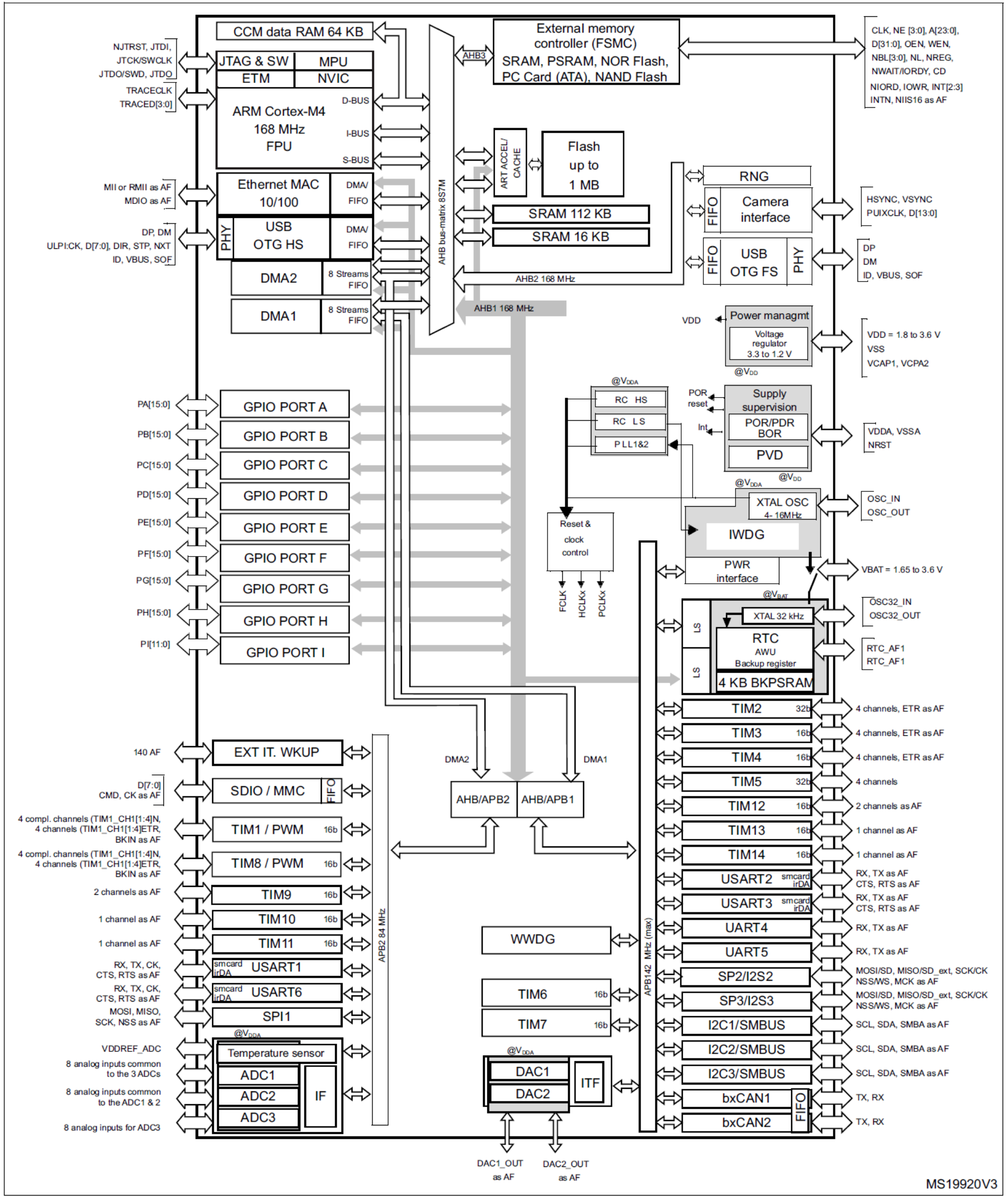
Cortex-M ADC main features

- 12-bit, 10-bit, 8-bit or 6-bit configurable resolution
- Interrupt generation at the end of conversion, end of injected conversion, and in case of analog watchdog or overrun events
- Single and continuous conversion modes
- Scan mode for automatic conversion of channel 0 to channel ‘n’
- Data alignment with in-built data coherency
- Channel-wise programmable sampling time
- External trigger option with configurable polarity for both regular and injected Conversions
- Discontinuous mode
- Dual/Triple mode (on devices with 2 ADCs or more)
- Configurable DMA data storage in Dual/Triple ADC mode
- Configurable delay between conversions in Dual/Triple interleaved mode
- ADC conversion type (refer to the datasheets)
- ADC supply requirements: 2.4 V to 3.6 V at full speed and down to 1.8 V at slower speed
- ADC input range: VREF– ≤ VIN ≤ VREF+
- DMA request generation during regular channel conversion
- *Note:VREF–, if available (depending on package), must be tied to* V_{SSA}.

STM32F40xxx block diagram & ADC

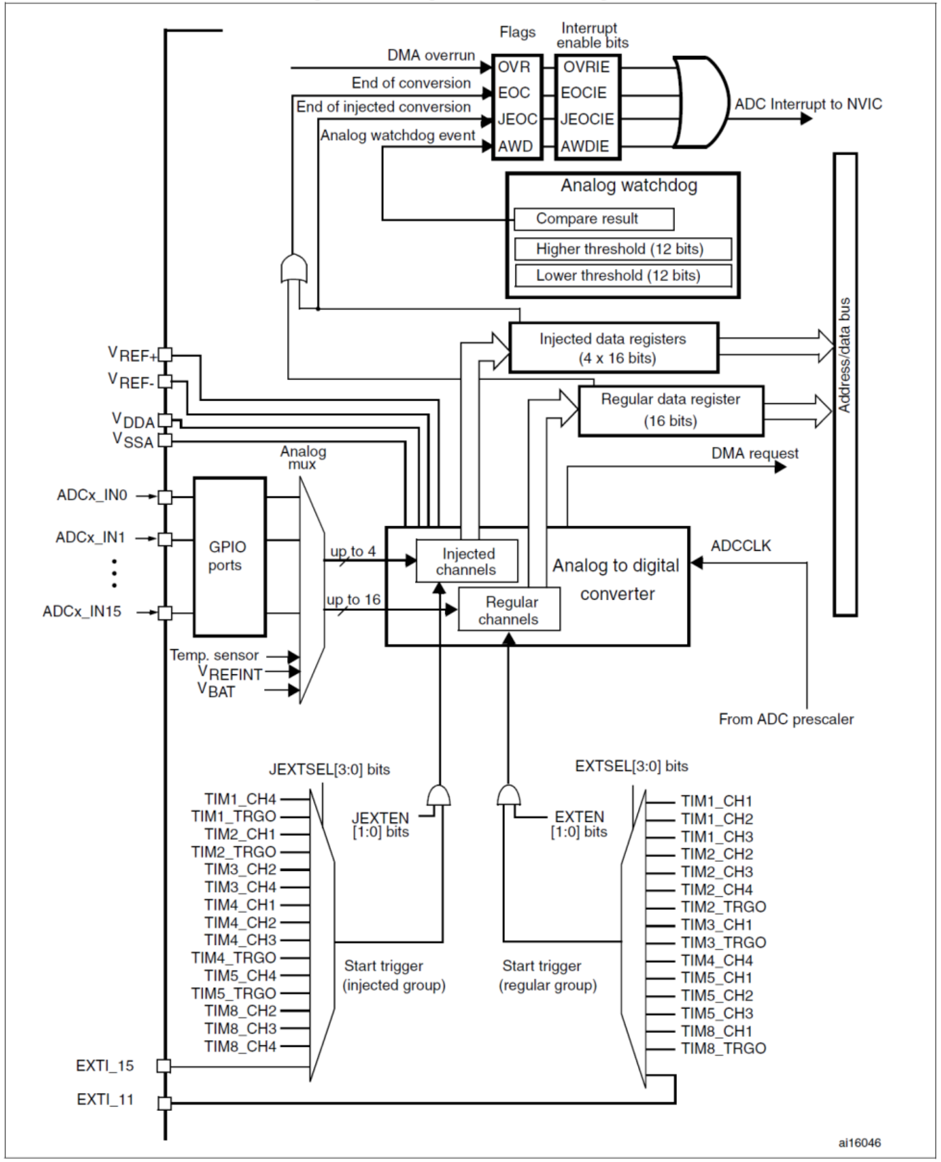
DS8626-p19

Figure 5. STM32F40xxx block diagram



- 전원
- 아나로그입력
- 트리거

Figure 44. Single ADC block diagram



ADC characteristics

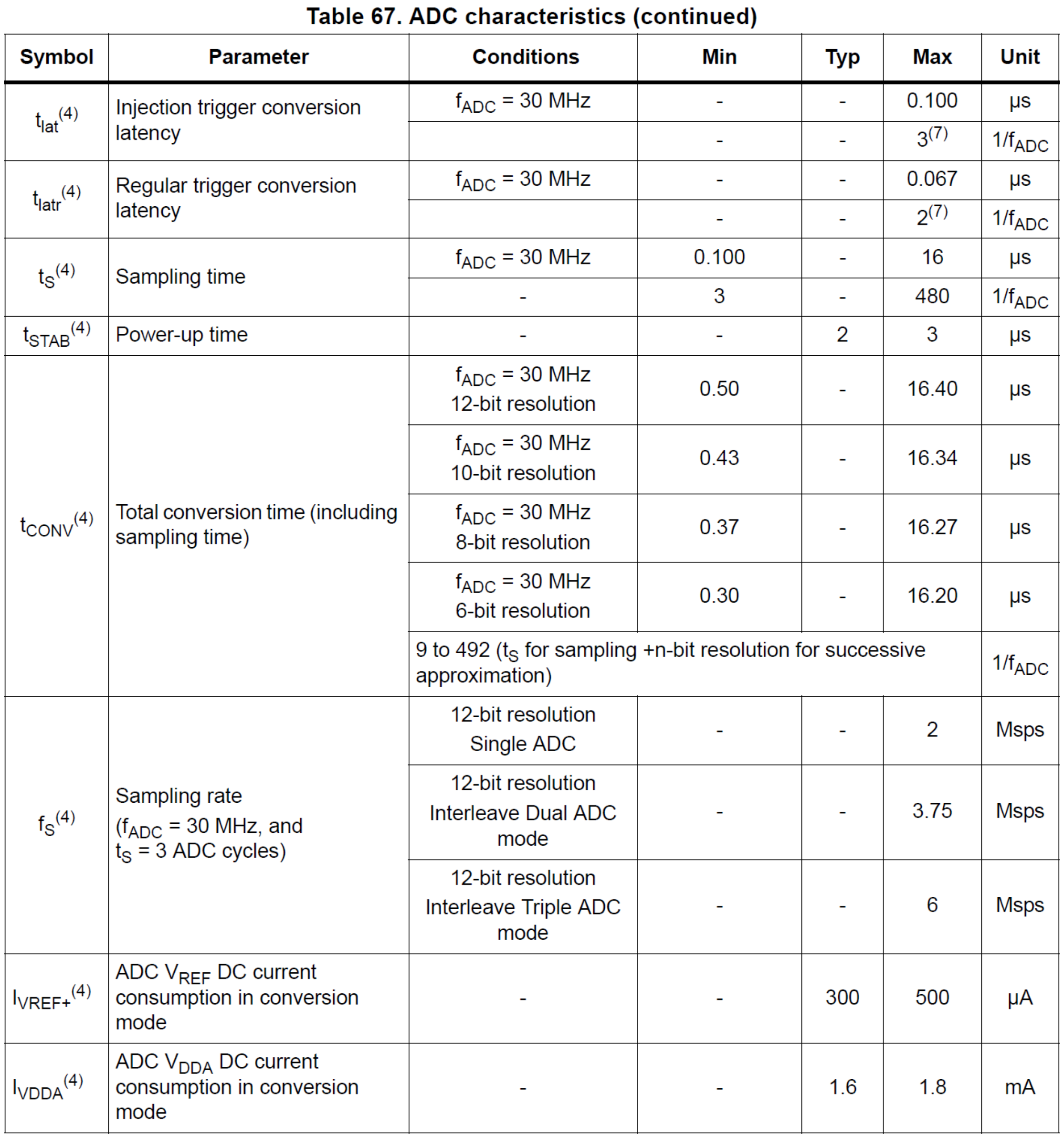


Table 67. ADC characteristics (continued)

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
t_{lat}[4]	Injection trigger conversion latency	f_{ADC} = 30 MHz	-	-	0.100	µs
			-	-	3[7]	$1/f_{ADC}$
t_{latr}[4]	Regular trigger conversion latency	f_{ADC} = 30 MHz	-	-	0.067	µs
			-	-	2[7]	$1/f_{ADC}$
t_S[4]	Sampling time	f_{ADC} = 30 MHz	0.100	-	16	µs
		-	3	-	480	$1/f_{ADC}$
t_{STAB}[4]	Power-up time	-	-	2	3	µs
t_{CONV}[4]	Total conversion time (including sampling time)	f_{ADC} = 30 MHz 12-bit resolution	0.50	-	16.40	µs
		f_{ADC} = 30 MHz 10-bit resolution	0.43	-	16.34	µs
		f_{ADC} = 30 MHz 8-bit resolution	0.37	-	16.27	µs
		f_{ADC} = 30 MHz 6-bit resolution	0.30	-	16.20	µs
		9 to 492 (t_S for sampling +n-bit resolution for successive approximation)				$1/f_{ADC}$
f_S[4]	Sampling rate (f_{ADC} = 30 MHz, and t_S = 3 ADC cycles)	12-bit resolution Single ADC	-	-	2	Msps
		12-bit resolution Interleave Dual ADC mode	-	-	3.75	Msps
		12-bit resolution Interleave Triple ADC mode	-	-	6	Msps
I_{VREF+}[4]	ADC V_{REF} DC current consumption in conversion mode	-	-	300	500	µA
I_{VDDA}[4]	ADC V_{DDA} DC current consumption in conversion mode	-	-	1.6	1.8	mA

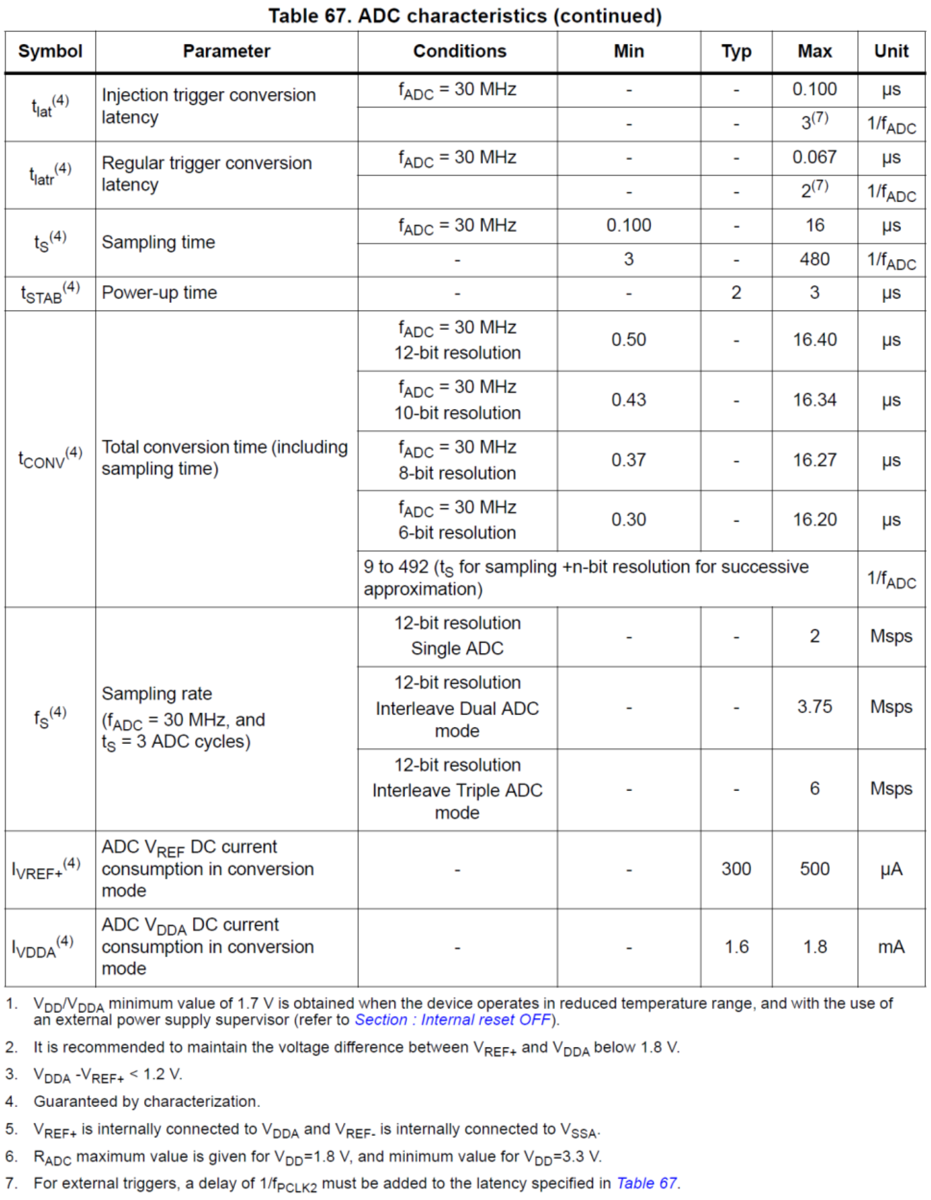


Table 67. ADC characteristics (continued)

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
t_{lat}[4]	Injection trigger conversion latency	f_{ADC} = 30 MHz	-	-	0.100	µs
			-	-	3[7]	$1/f_{ADC}$
t_{latr}[4]	Regular trigger conversion latency	f_{ADC} = 30 MHz	-	-	0.067	µs
			-	-	2[7]	$1/f_{ADC}$
t_S[4]	Sampling time	f_{ADC} = 30 MHz	0.100	-	16	µs
		-	3	-	480	$1/f_{ADC}$
t_{STAB}[4]	Power-up time	-	-	2	3	µs
t_{CONV}[4]	Total conversion time (including sampling time)	f_{ADC} = 30 MHz 12-bit resolution	0.50	-	16.40	µs
		f_{ADC} = 30 MHz 10-bit resolution	0.43	-	16.34	µs
		f_{ADC} = 30 MHz 8-bit resolution	0.37	-	16.27	µs
		f_{ADC} = 30 MHz 6-bit resolution	0.30	-	16.20	µs
		9 to 492 (t_S for sampling +n-bit resolution for successive approximation)				$1/f_{ADC}$
f_S[4]	Sampling rate (f_{ADC} = 30 MHz, and t_S = 3 ADC cycles)	12-bit resolution Single ADC	-	-	2	Msps
		12-bit resolution Interleave Dual ADC mode	-	-	3.75	Msps
		12-bit resolution Interleave Triple ADC mode	-	-	6	Msps
I_{VREF+}[4]	ADC V_{REF} DC current consumption in conversion mode	-	-	300	500	µA
I_{VDDA}[4]	ADC V_{DDA} DC current consumption in conversion mode	-	-	1.6	1.8	mA

1. V_{DD}/V_{DDA} minimum value of 1.7 V is obtained when the device operates in reduced temperature range, and with the use of an external power supply supervisor (refer to *Section : Internal reset OFF*).
2. It is recommended to maintain the voltage difference between V_{REF+} and V_{DDA} below 1.8 V.
3. V_{DDA} -V_{REF+} < 1.2 V.
4. Guaranteed by characterization.
5. V_{REF+} is internally connected to V_{DDA} and V_{REF-} is internally connected to V_{SSA}.
6. R_{ADC} maximum value is given for V_{DD}=1.8 V, and minimum value for V_{DD}=3.3 V.
7. For external triggers, a delay of $1/f_{PCLK2}$ must be added to the latency specified in *Table 67*.

❖ADC 전원연결

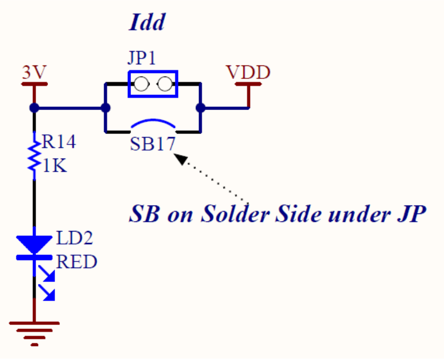
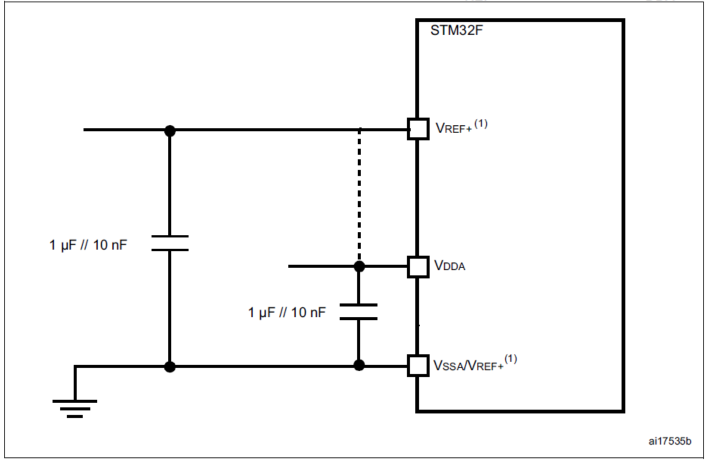
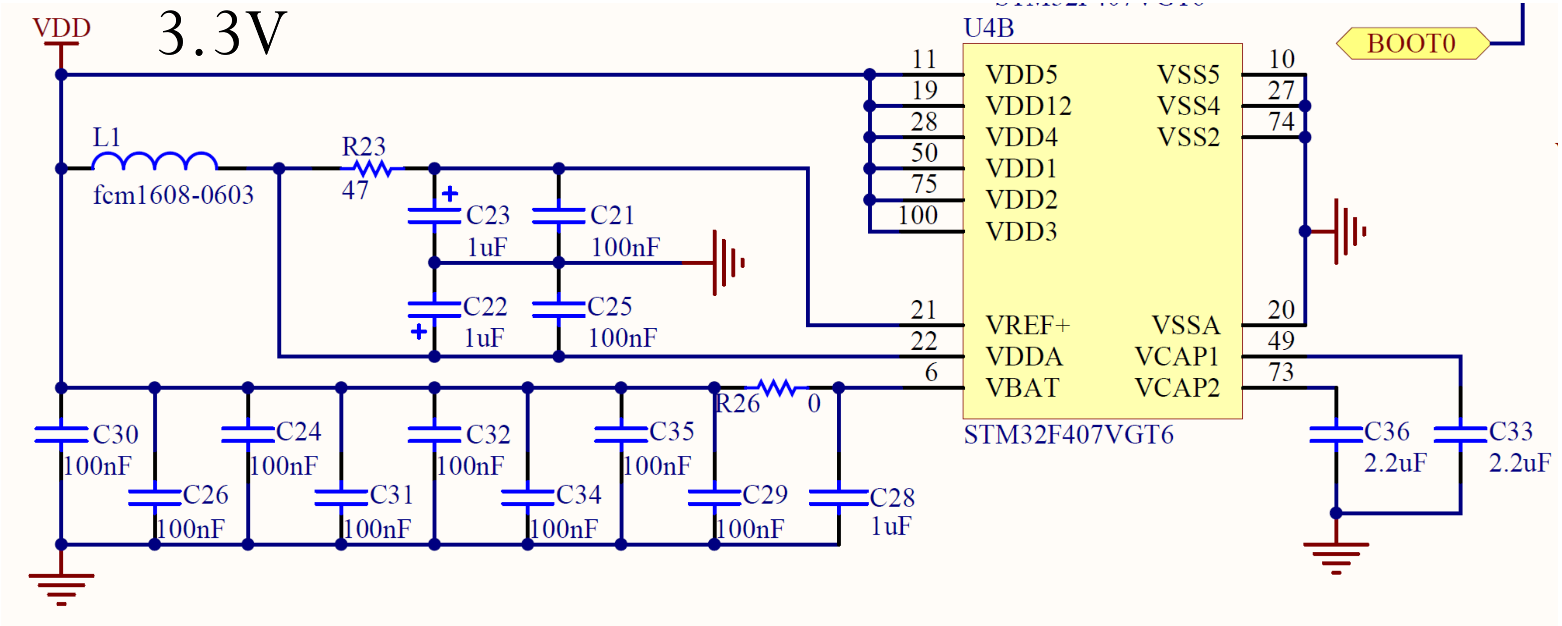


Figure 51. Power supply and reference decoupling (V_{REF+} not connected to V_{DDA})

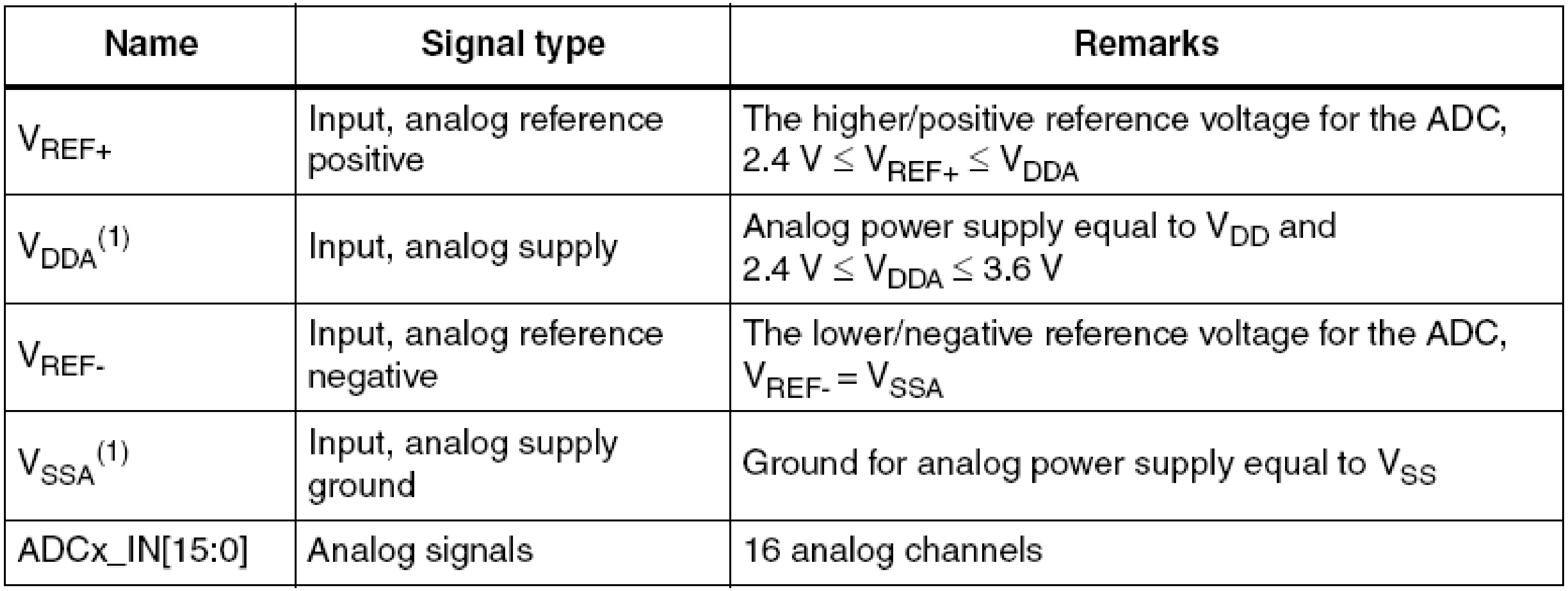


1. V_{REF+} and V_{REF-} inputs are both available on UFBGA176. V_{REF+} is also available on LQFP100, LQFP144, and LQFP176. When V_{REF+} and V_{REF-} are not available, they are internally connected to V_{DDA} and V_{SSA}.



STM32F407 주요 특징

- 12 비트의 분해능(resolution)
- 변환 종료 시 또는 아날로그 워치독 이벤트 발생 시에 인터럽트가 발생함
- 단일 또는 연속 변환 모드
- 스캔 모드에서는 채널 0부터 n까지 자동 변환이 가능
- ADC clock frequency : 0.6MHz < 30 <36 @VDDA = 2.4 to 3.6V
- ADC 공급 전압 요구사항 :
  - Full speed 상황에서 2.4V ~ 3.6V,
  - Slow speed 상황에서 1.8V 이하
- ADC input 범위 : Vref-(Vssa에 연결됨) ~ Vref+(Vdda에 연결됨 또는 외부 전압 사항)



Name	Signal type	Remarks
V_{REF+}	Input, analog reference positive	The higher/positive reference voltage for the ADC, $2.4\text{ V} \leq V_{REF+} \leq V_{DDA}$
V_{DDA}[1]	Input, analog supply	Analog power supply equal to V_{DD} and $2.4\text{ V} \leq V_{DDA} \leq 3.6\text{ V}$
V_{REF-}	Input, analog reference negative	The lower/negative reference voltage for the ADC, $V_{REF-} = V_{SSA}$
V_{SSA}[1]	Input, analog supply ground	Ground for analog power supply equal to V_{SS}
ADCx_IN[15:0]	Analog signals	16 analog channels

주요 기능

Channel selection

- conversions in two groups:regular and injected.
  - A group consists of a sequence of conversions that can be done on any channel and in any order. For instance, ADC_IN3, ADC_IN8, ADC_IN2, ADC_IN2, ADC_IN0, ADC_IN2, ADC_IN2, ADC_IN15.
- **regular group :**
  - is composed of up to 16 conversions.
  - ADC_SQRx에서 regular channels과 변환순서 설정
  - 총 변환갯수 : ADC_SQR1의 L[3:0] bits 에 설정
- **injected group**
  - is composed of up to 4 conversions.
  - ADC_JSQR 에서 injected channels과 변환순서 설정
  - 총변환갯수 : ADC_JSQR 의 L[1:0] bits 에 설정
- If the ADC_SQRx or ADC_JSQR registers are modified during a conversion, the current conversion is reset and a new start pulse is sent to the ADC to convert the newly chosen group.

❖ Temperature sensor, VREFINT and VBAT internal channels

- the temperature sensor is internally connected to channel ADC1_IN16.(STM32F40x and STM32F41x devices)
- The internal reference voltage VREFINT is connected to ADC1_IN17.
- For the STM32F42x and STM32F43x devices, the temperature sensor is internally connected to ADC1_IN18 channel which is shared with VBAT. Only one conversion, temperature sensor or VBAT, must be selected at a time. When the temperature sensor and VBAT conversion are set simultaneously, only the VBAT conversion is performed.
- The internal reference voltage VREFINT is connected to ADC1_IN17.
- The VBAT channel (connected to channel ADC1_IN18) can also be converted as an injected or regular channel.
- *Note: The temperature sensor, VREFINT and the VBAT channel are available only on the master ADC1 peripheral.*

Single conversion mode

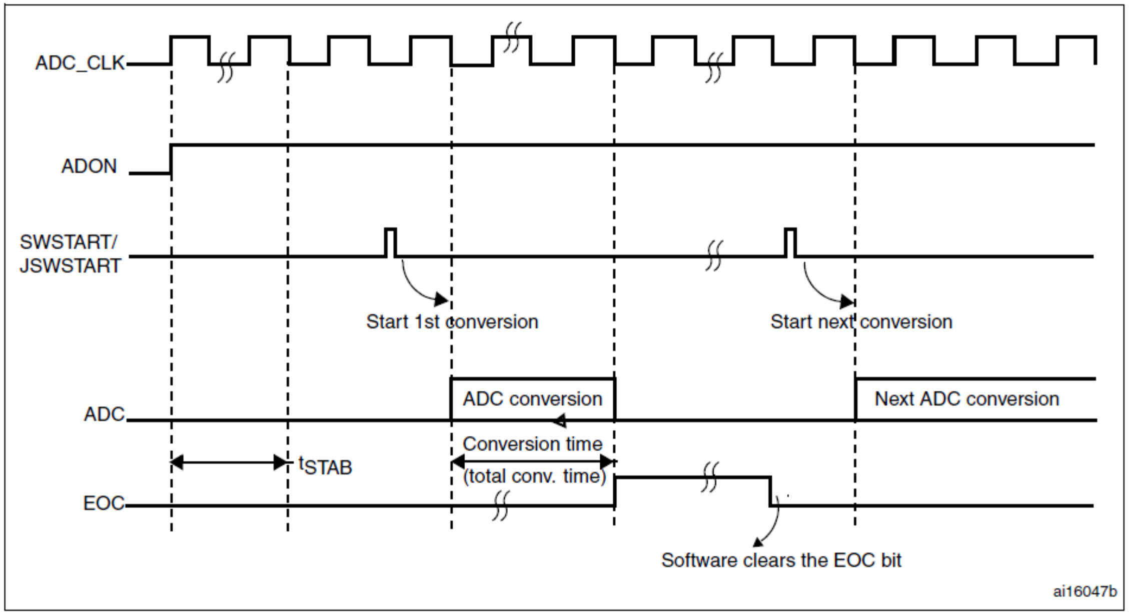
RM0090-p391

- ADC does one conversion.
- This mode is started with the CONT bit at 0 by either:
  - setting the SWSTART bit in the ADC_CR2 register (for a regular channel only)
  - setting the JSWSTART bit (for an injected channel)
  - external trigger (for a regular or injected channel)
- Once the conversion of the selected channel is complete:
  - If a regular channel was converted:
    - The converted data are stored into the 16-bit ADC_DR register
    - The EOC (end of conversion) flag is set
    - An interrupt is generated if the EOCIE bit is set
  - If an injected channel was converted:
    - The converted data are stored into the 16-bit ADC_JDR1 register
    - The JEOC (end of conversion injected) flag is set
    - An interrupt is generated if the JEOCIE bit is set
- Then the ADC stops.

Continuous conversion mode

- the ADC starts a new conversion as soon as it finishes one.
- This mode is started with the CONT bit at 1 either
  - by external trigger or
  - by setting the SWSTRT bit in the ADC_CR2 register (for regular channels only).
- After each conversion:
  - • If a regular group of channels was converted:
    - The last converted data are stored into the 16-bit ADC_DR register
    - The EOC (end of conversion) flag is set
    - An interrupt is generated if the EOCIE bit is set
- *Note: Injected channels cannot be converted continuously. The only exception is when an injected channel is configured to be converted automatically after regular channels in continuous mode (using JAUTO bit), refer to Auto-injection section)*

❖동작 타이밍 선도


ADC_CLK
ADON
SWSTART/
JSWSTART
Start 1st conversion
Start next conversion
ADC
ADC conversion
Next ADC conversion
Conversion time
(total conv. time)
tSTAB
EOC
Software clears the EOC bit
ai16047b

스캔(Scan) 모드

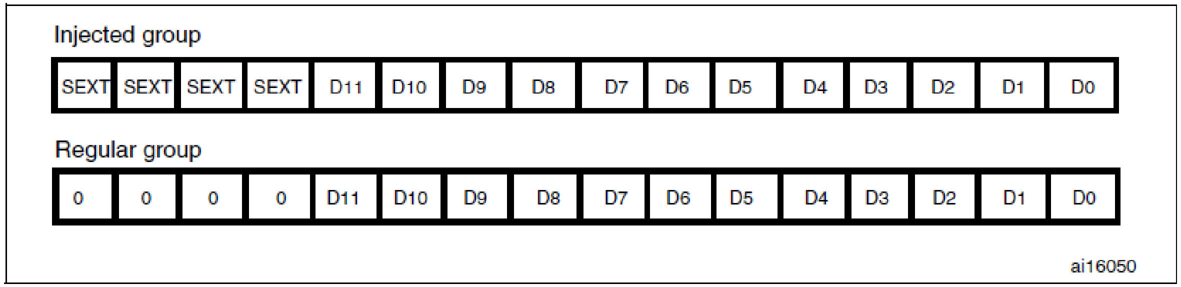
- 채널 그룹 내의 모든 채널들을 순서대로 스캔하여 변환하는 모드.
- ADC_CR1 레지스터의 SCAN 비트를 1로 두면 스캔 모드로 설정
- 이 모드에서는 채널 그룹(인젝티드 그룹 또는 레귤러 그룹)에 포함된 모든 채널들이 미리 정해진 순서를 따라 변환된다. 연속 변환 모드인 경우는 스캔 모드의 동작이 연속적으로 일어나게 된다.

비연속 모드

- 연속 모드는 채널 그룹 내의 일부 채널을 미리 정해진 순서대로 변환하는 모드이다. 외부 트리거 신호가 입력되면 n개(n <=8) 채널의 변환만 이루어진다. 다시 외부 트리거 신호가 입력되면 다음의 n개의 변환이 이루어지며, n개가 되지 않더라도 정해진 변환 순서의 끝이 되면 그대로 종료된다.
- (변환의 예) n = 3이고, 채널의 변환 순서 = 0, 1, 2, 3, 6, 7, 9, 10 일 경우
  - - 1번째 트리거 입력 : 채널 0, 1, 2 가 변환됨
  - - 2번째 트리거 입력 : 채널 3, 6, 7 이 변환됨
  - - 3번째 트리거 입력 : 채널 9, 10 이 변환되고 변환 종료(EOC) 이벤트 발생
  - - 4번째 트리거 입력 : 다시 채널 0, 1, 2 가 변환됨
  - - 이 후는 동일한 방식의 동작이 반복됨

23

❖ 데이터 정렬



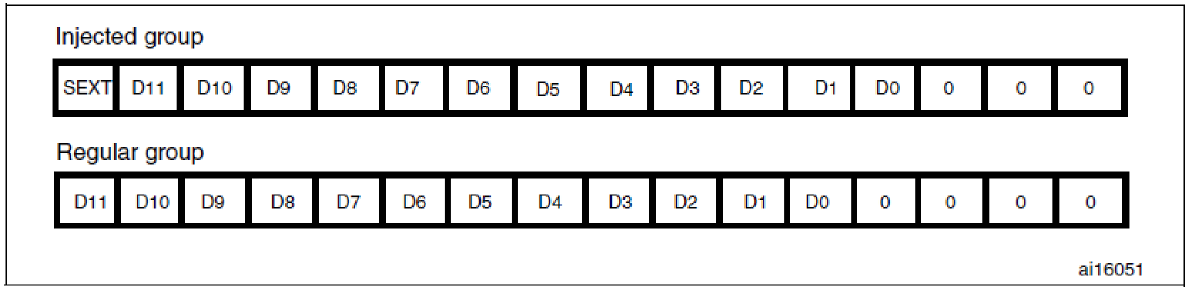
Injected group

SEXT	SEXT	SEXT	SEXT	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

Regular group

0	0	0	0	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

ai16050



Injected group

SEXT	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0

Regular group

D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	0	0	0	0

ai16051

❖ **AD 변환기는 12비트이므로**

- ➢ 오른쪽 정렬의 경우,
  - 인젝티드 그룹은 최상위 비트 4개가 SEXT라는 값으로 설정. SEXT는 변환 결과가 음수인가 양수인가에 대한 부호를 나타내는 값.
  - 레귤러 그룹의 경우는 최상위 비트 4개가 모두 0으로 설정.
- ➢ 왼쪽 정렬의 경우,
  - 인젝티드 그룹 : 최상위 비트 1개가 SEXT로 설정되며 최하위 비트 3개는 0으로 설정.
  - 레귤러 그룹: 최하위 비트 4개가 모두 0으로 설정

❖외부 트리거에 의한 변환 시작

- ➢ADC 컨트롤 레지스터2(ADC_CR2)의 EXTSEL[2:0] 비트를 설정을 하면 레귤러 채널에 대하여 외부 트리거 신호에 의한 변환 시작.

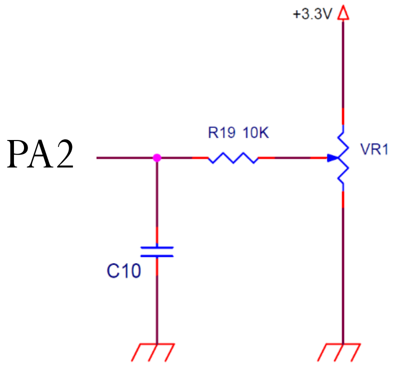
❖ADC 인터럽트

- ➢AD 변환이 완료된 경우는 인터럽트가 발생.
- ➢ADC1과 ADC2의 인터럽트는 동일한 인터럽트 벡터를 가지며 ADC3은 이와는 다른 인터럽트 벡터를 가진다.

예제

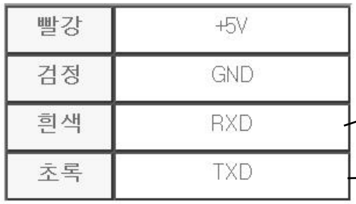
예제1 : ADC 출력 값 UART로 출력

- 가변 저항(VR1)을 조절하여 ADC1로 입력되는 전압의 값을 변화시킨다. 그러면 ADC1의 변환값출력을 읽는다
- 이값을 시리얼 포트로 출력한다.
- 폴링(polling) 방식으로 ADC1을 동작시킨다.
- 사용자 프로그램내에서 HAL_ADC_Start() 함수를 이용하여 ADC1을 동작시키면(즉, AD 변환을 시작하면), 이 AD 변환이 완료될 때까지 프로그램은 다른 동작을 수행하지 않고 대기한다.
- 이 대기하는 동작은 HAL_ADC_PollForConversion() 함수를 이용하여 이루어진다.
- 1sec 마다 ADC 변환하여 UART로 출력



PA2

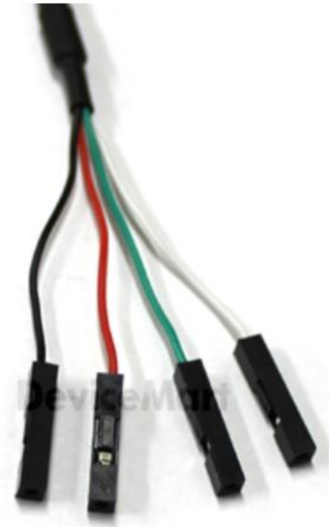
- 마이크로컨트롤러, 라즈베리파이, 와이파이 라우터 시리얼 콘솔 케이블
- PL2303TA 칩셋 사용(PL2303HX과 같으며 윈도우8 지원) 5V 500mA 사용 가능



빨강	+5V
검정	GND
흰색	RXD
초록	TXD

TxD에 연결

RxD에 연결

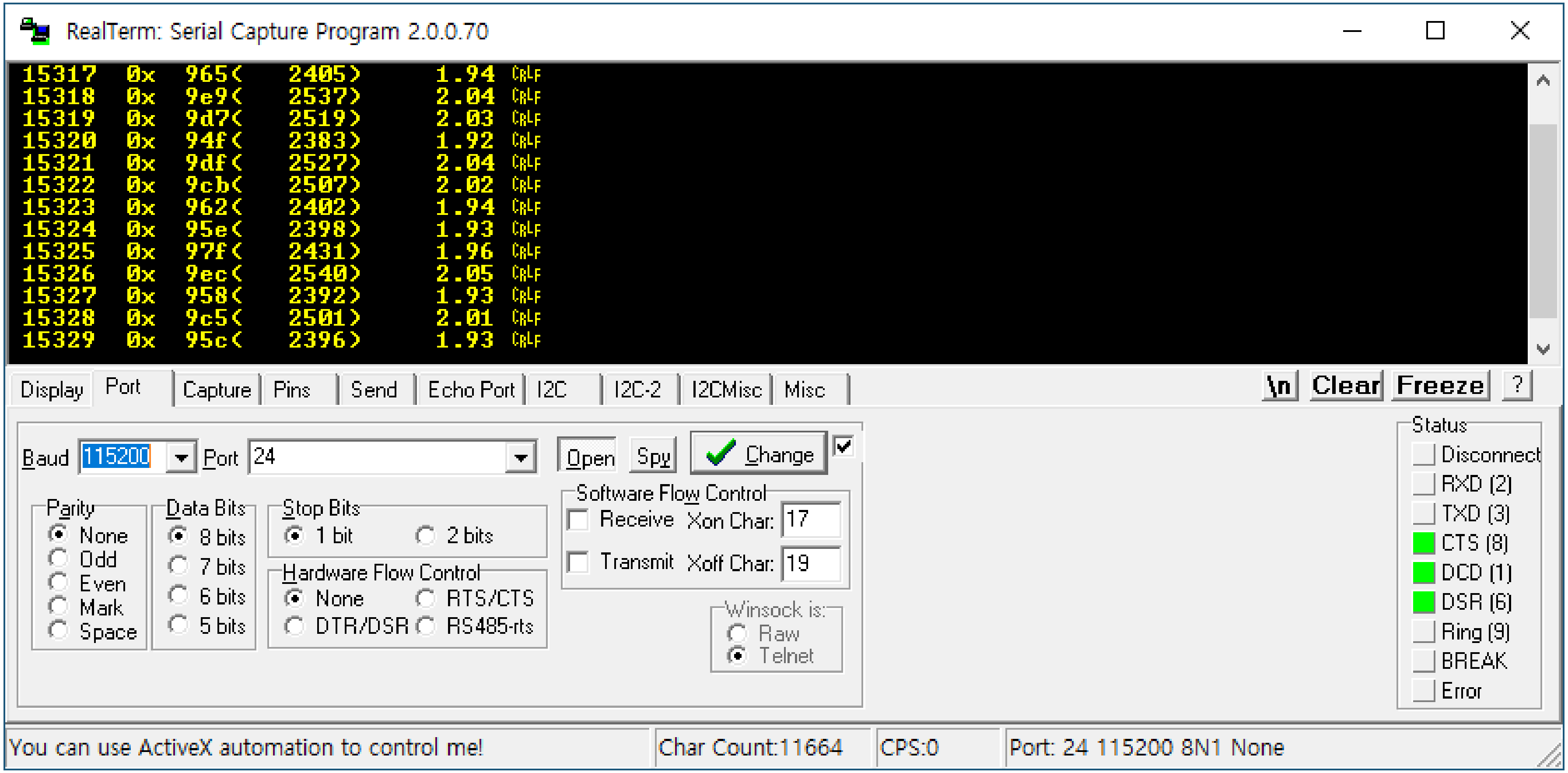


ADC 관련 초기화

ADC 변환 소스

RealTerm

❖포트설정 및 baud rate



RealTerm: Serial Capture Program 2.0.0.70

```
15317  0x  965<   2405>     1.94 CRLF
15318  0x  9e9<   2537>     2.04 CRLF
15319  0x  9d7<   2519>     2.03 CRLF
15320  0x  94f<   2383>     1.92 CRLF
15321  0x  9df<   2527>     2.04 CRLF
15322  0x  9cb<   2507>     2.02 CRLF
15323  0x  962<   2402>     1.94 CRLF
15324  0x  95e<   2398>     1.93 CRLF
15325  0x  97f<   2431>     1.96 CRLF
15326  0x  9ec<   2540>     2.05 CRLF
15327  0x  958<   2392>     1.93 CRLF
15328  0x  9c5<   2501>     2.01 CRLF
15329  0x  95c<   2396>     1.93 CRLF
```

Baud 115200 Port 24

Parity: ● None ○ Odd ○ Even ○ Mark ○ Space

Data Bits: ● 8 bits ○ 7 bits ○ 6 bits ○ 5 bits

Stop Bits: ● 1 bit ○ 2 bits

Hardware Flow Control: ● None ○ RTS/CTS ○ DTR/DSR ○ RS485-rts

Software Flow Control: □ Receive Xon Char: 17 □ Transmit Xoff Char: 19

Winsock is: ○ Raw ● Telnet

☑

Status: □ Disconnect □ RXD (2) □ TXD (3) 🟩 CTS (8) 🟩 DCD (1) 🟩 DSR (6) □ Ring (9) □ BREAK □ Error

You can use ActiveX automation to control me!

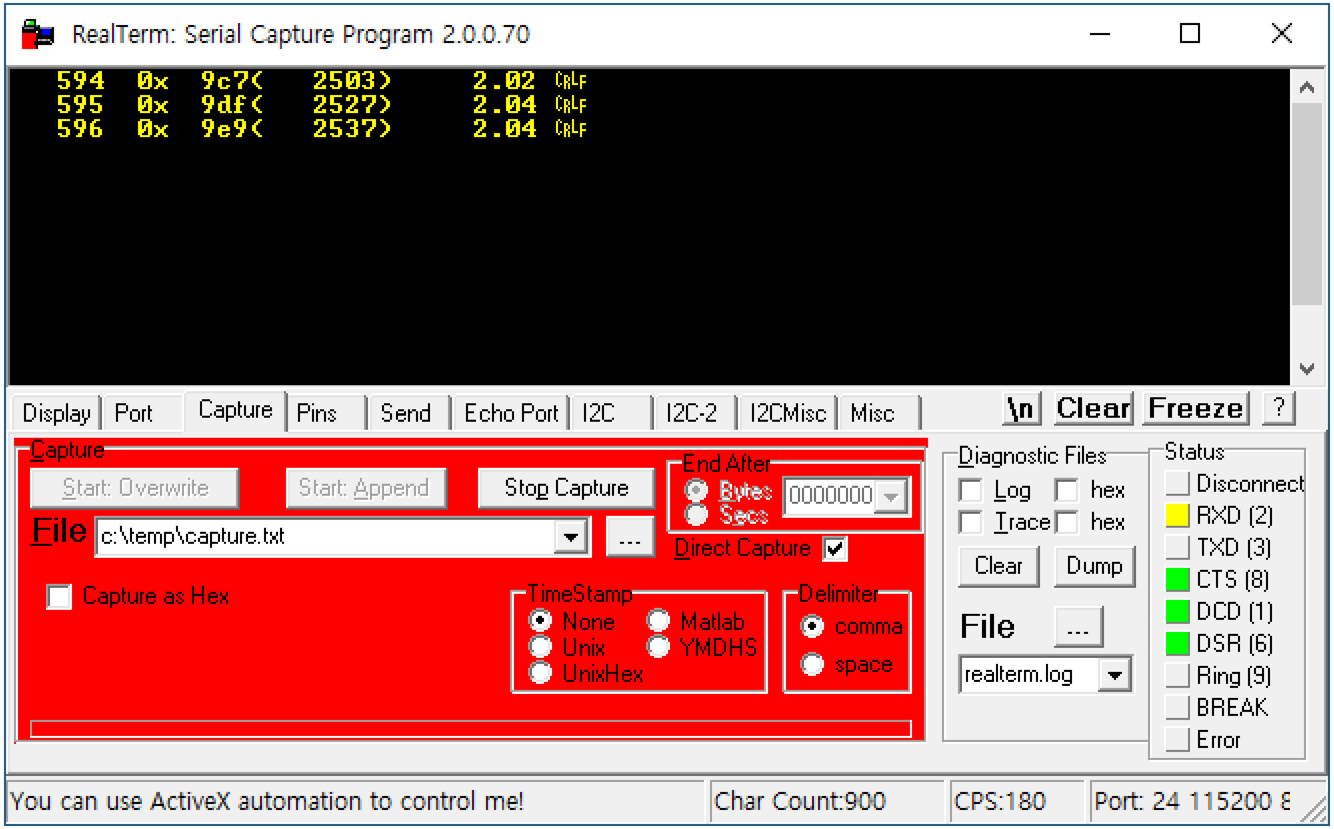
Char Count:11664

CPS:0

Port: 24 115200 8N1 None

시리얼 캡쳐

- ❖ 시리얼포트 출력 캡쳐 저장
- ❖ 저장된 텍스트화일을 엑셀에서 읽어서 그래프 그릴 수 있음



RealTerm: Serial Capture Program 2.0.0.70

```
  594  0x  9c7(   2503)      2.02 CRLF
  595  0x  9df(   2527)      2.04 CRLF
  596  0x  9e9(   2537)      2.04 CRLF
```

Capture

File c:\temp\capture.txt

End After: ○ Bytes ○ Secs 0000000

Direct Capture ☑

□ Capture as Hex

TimeStamp: ● None ○ Matlab ○ Unix ○ YMDHS ○ UnixHex

Delimiter: ● comma ○ space

Diagnostic Files: □ Log □ hex □ Trace □ hex

File realterm.log

Status: □ Disconnect 🟨 RXD (2) □ TXD (3) 🟩 CTS (8) 🟩 DCD (1) 🟩 DSR (6) □ Ring (9) □ BREAK □ Error

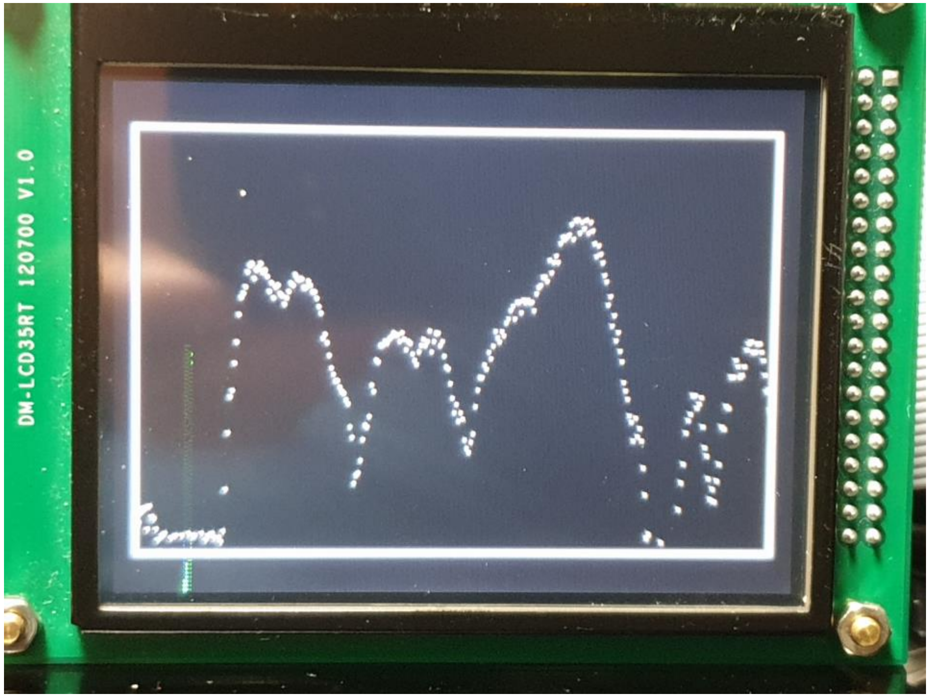
You can use ActiveX automation to control me! | Char Count:900 | CPS:180 | Port: 24 115200 8

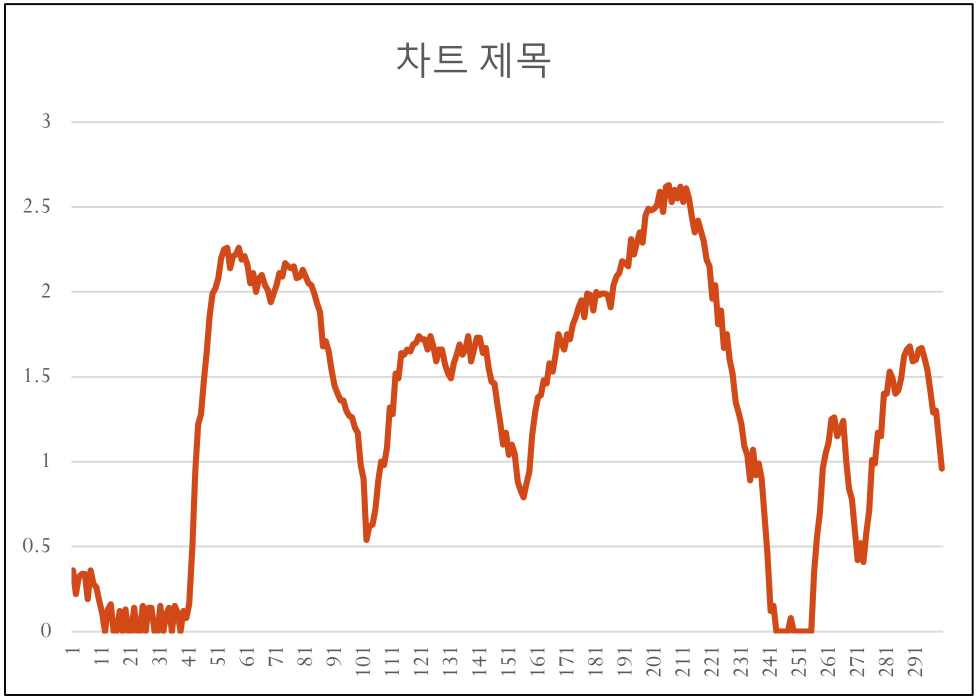
Report

35

- **프로젝트명 : 201513001HKD <= 반드시 학번+이름영문약자**
  - 다를 경우 인정안함
- **ADC_IN2(PA2)로 입력된 아나로그값을 LCD와 Excel 그래프로 그리기**
  - 0.1초 간격으로 300샘플만 읽어와서 그리기
  - 값은 UART6로 16진수 10진수 실수값으로 출력
    - 엑셀에서 읽기 편하도록 공백또는 콤마(,)등을 넣어서 출력
  - 이 출력을 RealTerm으로 캡쳐 및 저장해서 엑셀로 그리기
  - 리포트
    - 작성부분 소스분석
    - LCD 화면과 엑셀그래프 화면 캡쳐 비교
    - 작성된 리포트 파일을 작업폴더에 넣고 폴더명을 압축 eclass에 제출

❖LCD 출력과 엑셀그림 비교





```
HAL : HAL_UART_Transmit(&huart6, (uint8_t *) string, strlen(string), 1000);
```

Temperature sensor

- The temperature sensor can be used to measure the ambient temperature (TA) of the device.
- • On STM32F42x and STM32F43x devices, the temperature sensor is internally connected to the same input channel, ADC1_IN18, as VBAT: ADC1_IN18 is used to convert the sensor output voltage or VBAT into a digital value. Only one conversion, temperature sensor or VBAT, must be selected at a time. When the temperature sensor and the VBAT conversion are set simultaneously, only the VBAT conversion is performed.
- *Note: The TSVREFE bit must be set to enable the conversion of both internal channels: the ADC1_IN16 or ADC1_IN18 (temperature sensor) and the ADC1_IN17 (VREFINT).*

Main features

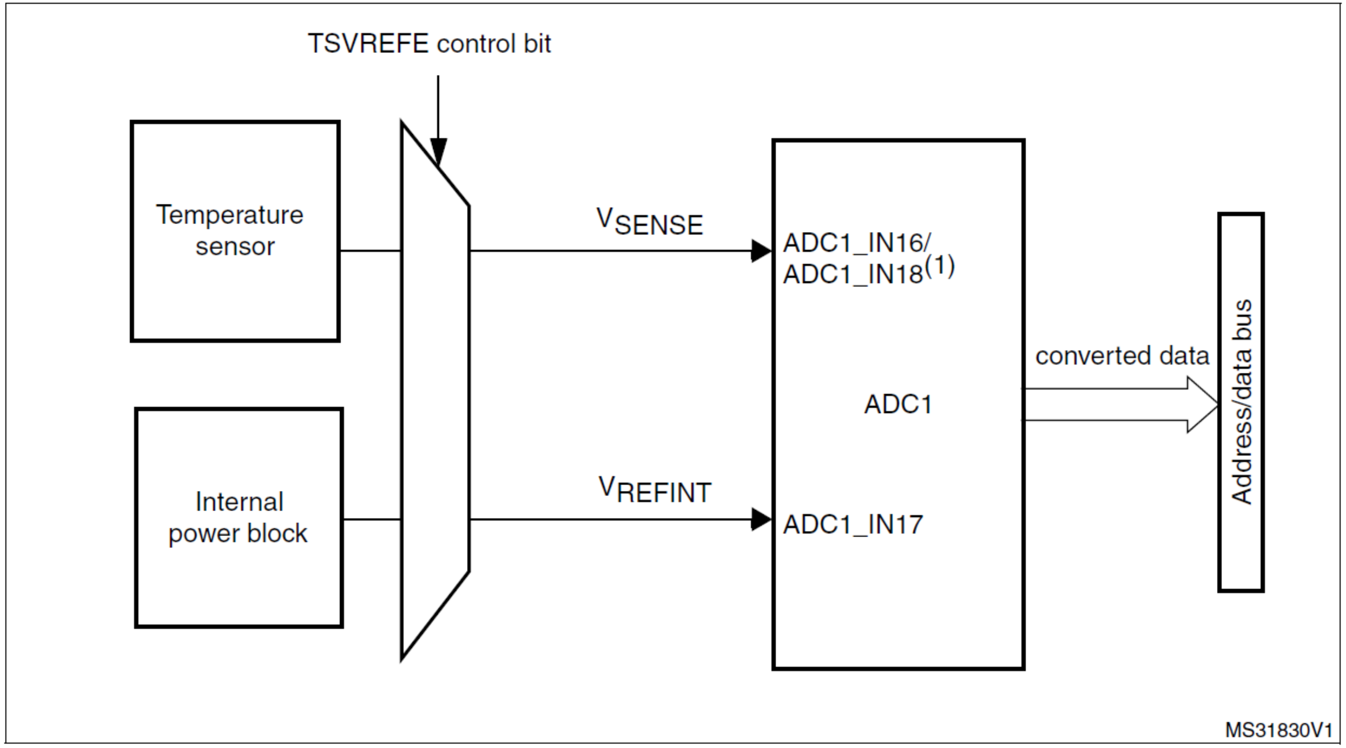
- • Supported temperature range: –40 to 125 °C
- • Precision: ±1.5 °C

Reading the temperature

- 3. Select ADC1_IN16 or ADC1_IN18 input channel.
- 4. Select a sampling time greater than the minimum sampling time specified in the datasheet.
- 5. Set the TSVREFE bit in the ADC_CCR register to wake up the temperature sensor from power down mode
- 6. Start the ADC conversion by setting the SWSTART bit (or by external trigger)
- 7. Read the resulting VSENSE data in the ADC data register
- 8. Calculate the temperature using the following formula:
  - Temperature (in °C) = {(VSENSE – V25) / Avg_Slope} + 25
    - – V25 = VSENSE value for 25° C
    - – Avg_Slope = average slope of the temperature vs. VSENSE curve (given in mV/°C or μV/¡ÆC)
  - Refer to the datasheet's electrical characteristics section for the actual values of V25 and Avg_Slope. *Note: The sensor has a startup*

40

Figure 63. Temperature sensor and V_{REFINT} channel block diagram



Battery charge monitoring

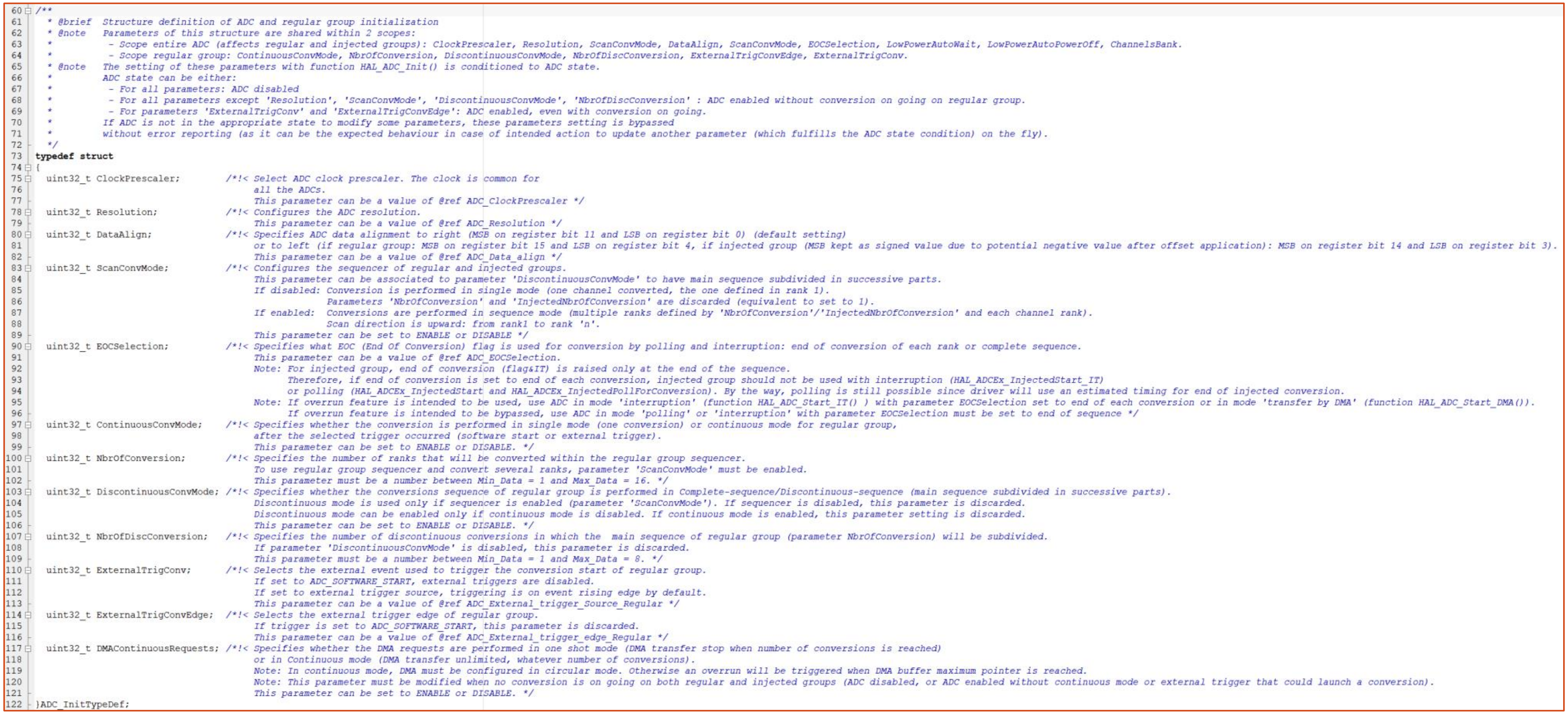
- The VBATE bit in the ADC_CCR register is used to switch to the battery voltage. As the VBAT voltage could be higher than VDDA, to ensure the correct operation of the ADC, the VBAT pin is internally connected to a bridge divider.
- When the VBATE is set, the bridge is automatically enabled to connect:
  - • VBAT/2 to the ADC1_IN18 input channel, on STM32F40xx and STM32F41xx devices
  - • VBAT/4 to the ADC1_IN18 input channel, on STM32F42xx and STM32F43xx devices
- *Note: On STM32F42xx and STM32F43xx devices, VBAT and temperature sensor are connected to the same ADC internal channel (ADC1_IN18). Only one conversion, either temperature sensor or VBAT, must be selected at a time. When both conversion are enabled simultaneously, only the VBAT conversion is performed.*

AD 변환기 관련 HAL 드라이버

43

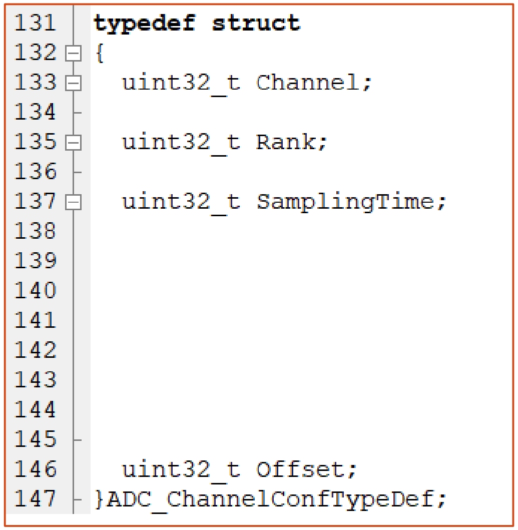
ADC 설정용 구조체

ADC_InitTypeDef : ADC의 초기 설정을 위한 구조체

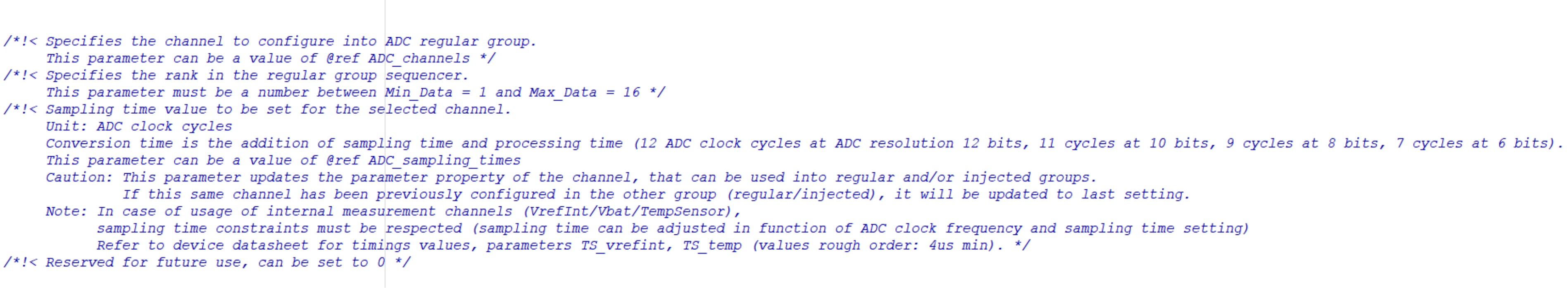


```c
/**
  * @brief  Structure definition of ADC and regular group initialization
  * @note   Parameters of this structure are shared within 2 scopes:
  *           - Scope entire ADC (affects regular and injected groups): ClockPrescaler, Resolution, ScanConvMode, DataAlign, ScanConvMode, EOCSelection, LowPowerAutoWait, LowPowerAutoPowerOff, ChannelsBank.
  *           - Scope regular group: ContinuousConvMode, NbrOfConversion, DiscontinuousConvMode, NbrOfDiscConversion, ExternalTrigConvEdge, ExternalTrigConv.
  * @note   The setting of these parameters with function HAL_ADC_Init() is conditioned to ADC state.
  *         ADC state can be either:
  *           - For all parameters: ADC disabled
  *           - For all parameters except 'Resolution', 'ScanConvMode', 'DiscontinuousConvMode', 'NbrOfDiscConversion' : ADC enabled without conversion on going on regular group.
  *           - For parameters 'ExternalTrigConv' and 'ExternalTrigConvEdge': ADC enabled, even with conversion on going.
  *         If ADC is not in the appropriate state to modify some parameters, these parameters setting is bypassed
  *         without error reporting (as it can be the expected behaviour in case of intended action to update another parameter (which fulfills the ADC state condition) on the fly).
  */
typedef struct
{
  uint32_t ClockPrescaler;        /*!< Select ADC clock prescaler. The clock is common for
                                       all the ADCs.
                                       This parameter can be a value of @ref ADC_ClockPrescaler */
  uint32_t Resolution;            /*!< Configures the ADC resolution.
                                       This parameter can be a value of @ref ADC_Resolution */
  uint32_t DataAlign;             /*!< Specifies ADC data alignment to right (MSB on register bit 11 and LSB on register bit 0) (default setting)
                                       or to left (if regular group: MSB on register bit 15 and LSB on register bit 4, if injected group (MSB kept as signed value due to potential negative value after offset application): MSB on register bit 14 and LSB on register bit 3).
                                       This parameter can be a value of @ref ADC_Data_align */
  uint32_t ScanConvMode;          /*!< Configures the sequencer of regular and injected groups.
                                       This parameter can be associated to parameter 'DiscontinuousConvMode' to have main sequence subdivided in successive parts.
                                       If disabled: Conversion is performed in single mode (one channel converted, the one defined in rank 1).
                                                    Parameters 'NbrOfConversion' and 'InjectedNbrOfConversion' are discarded (equivalent to set to 1).
                                       If enabled:  Conversions are performed in sequence mode (multiple ranks defined by 'NbrOfConversion'/'InjectedNbrOfConversion' and each channel rank).
                                                    Scan direction is upward: from rank1 to rank 'n'.
                                       This parameter can be set to ENABLE or DISABLE */
  uint32_t EOCSelection;          /*!< Specifies what EOC (End Of Conversion) flag is used for conversion by polling and interruption: end of conversion of each rank or complete sequence.
                                       This parameter can be a value of @ref ADC_EOCSelection.
                                       Note: For injected group, end of conversion (flag&IT) is raised only at the end of the sequence.
                                             Therefore, if end of conversion is set to end of each conversion, injected group should not be used with interruption (HAL_ADCEx_InjectedStart_IT)
                                             or polling (HAL_ADCEx_InjectedStart and HAL_ADCEx_InjectedPollForConversion). By the way, polling is still possible since driver will use an estimated timing for end of injected conversion.
                                       Note: If overrun feature is intended to be used, use ADC in mode 'interruption' (function HAL_ADC_Start_IT() ) with parameter EOCSelection set to end of each conversion or in mode 'transfer by DMA' (function HAL_ADC_Start_DMA()).
                                             If overrun feature is intended to be bypassed, use ADC in mode 'polling' or 'interruption' with parameter EOCSelection must be set to end of sequence */
  uint32_t ContinuousConvMode;    /*!< Specifies whether the conversion is performed in single mode (one conversion) or continuous mode for regular group,
                                       after the selected trigger occurred (software start or external trigger).
                                       This parameter can be set to ENABLE or DISABLE. */
  uint32_t NbrOfConversion;       /*!< Specifies the number of ranks that will be converted within the regular group sequencer.
                                       To use regular group sequencer and convert several ranks, parameter 'ScanConvMode' must be enabled.
                                       This parameter must be a number between Min_Data = 1 and Max_Data = 16. */
  uint32_t DiscontinuousConvMode; /*!< Specifies whether the conversions sequence of regular group is performed in Complete-sequence/Discontinuous-sequence (main sequence subdivided in successive parts).
                                       Discontinuous mode is used only if sequencer is enabled (parameter 'ScanConvMode'). If sequencer is disabled, this parameter is discarded.
                                       Discontinuous mode can be enabled only if continuous mode is disabled. If continuous mode is enabled, this parameter setting is discarded.
                                       This parameter can be set to ENABLE or DISABLE. */
  uint32_t NbrOfDiscConversion;   /*!< Specifies the number of discontinuous conversions in which the  main sequence of regular group (parameter NbrOfConversion) will be subdivided.
                                       If parameter 'DiscontinuousConvMode' is disabled, this parameter is discarded.
                                       This parameter must be a number between Min_Data = 1 and Max_Data = 8. */
  uint32_t ExternalTrigConv;      /*!< Selects the external event used to trigger the conversion start of regular group.
                                       If set to ADC_SOFTWARE_START, external triggers are disabled.
                                       If set to external trigger source, triggering is on event rising edge by default.
                                       This parameter can be a value of @ref ADC_External_trigger_Source_Regular */
  uint32_t ExternalTrigConvEdge;  /*!< Selects the external trigger edge of regular group.
                                       If trigger is set to ADC_SOFTWARE_START, this parameter is discarded.
                                       This parameter can be a value of @ref ADC_External_trigger_edge_Regular */
  uint32_t DMAContinuousRequests; /*!< Specifies whether the DMA requests are performed in one shot mode (DMA transfer stop when number of conversions is reached)
                                       or in Continuous mode (DMA transfer unlimited, whatever number of conversions).
                                       Note: In continuous mode, DMA must be configured in circular mode. Otherwise an overrun will be triggered when DMA buffer maximum pointer is reached.
                                       Note: This parameter must be modified when no conversion is on going on both regular and injected groups (ADC disabled, or ADC enabled without continuous mode or external trigger that could launch a conversion).
                                       This parameter can be set to ENABLE or DISABLE. */
}ADC_InitTypeDef;
```

ADC_ChannelConfTypeDef : ADC의 채널 설정 구조체



```
typedef struct
{
  uint32_t Channel;

  uint32_t Rank;

  uint32_t SamplingTime;








  uint32_t Offset;
}ADC_ChannelConfTypeDef;
```

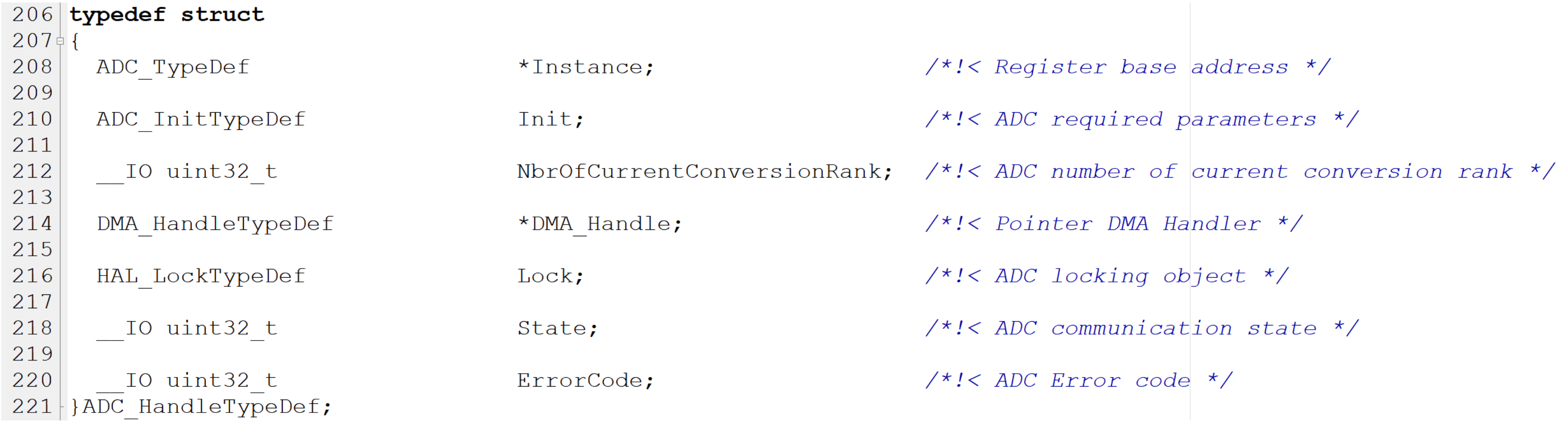


```
/*!< Specifies the channel to configure into ADC regular group.
     This parameter can be a value of @ref ADC_channels */
/*!< Specifies the rank in the regular group sequencer.
     This parameter must be a number between Min_Data = 1 and Max_Data = 16 */
/*!< Sampling time value to be set for the selected channel.
     Unit: ADC clock cycles
     Conversion time is the addition of sampling time and processing time (12 ADC clock cycles at ADC resolution 12 bits, 11 cycles at 10 bits, 9 cycles at 8 bits, 7 cycles at 6 bits).
     This parameter can be a value of @ref ADC_sampling_times
     Caution: This parameter updates the parameter property of the channel, that can be used into regular and/or injected groups.
              If this same channel has been previously configured in the other group (regular/injected), it will be updated to last setting.
     Note: In case of usage of internal measurement channels (VrefInt/Vbat/TempSensor),
           sampling time constraints must be respected (sampling time can be adjusted in function of ADC clock frequency and sampling time setting)
           Refer to device datasheet for timings values, parameters TS_vrefint, TS_temp (values rough order: 4us min). */
/*!< Reserved for future use, can be set to 0 */
```

ADC_AnalogWDGConfTypeDef :analog watchdog 설정구조체

47

ADC_HandleTypeDef: 핸들 설정구조체



```
typedef struct
{
  ADC_TypeDef                   *Instance;                    /*!< Register base address */

  ADC_InitTypeDef               Init;                         /*!< ADC required parameters */

  __IO uint32_t                 NbrOfCurrentConversionRank;   /*!< ADC number of current conversion rank */

  DMA_HandleTypeDef             *DMA_Handle;                  /*!< Pointer DMA Handler */

  HAL_LockTypeDef               Lock;                         /*!< ADC locking object */

  __IO uint32_t                 State;                        /*!< ADC communication state */

  __IO uint32_t                 ErrorCode;                    /*!< ADC Error code */
}ADC_HandleTypeDef;
```

ADC 구동용 HAL 함수

1) 초기화(Initialization) 및 초기화 해제(de-initialization)용 함수

- · HAL_ADC_Init(), HAL_ADC_DeInit()
  - - ADC의 초기화, 초기화 해제
- · HAL_ADC_MspInit(), HAL_ADC_MspDeInit()
  - - ADC Msp의 초기화, 초기화 해제

2) 입출력용 함수·

- HAL_ADC_Start()
  - - ADC의 레귤러 그룹의 동작을 시작한다
- HAL_ADC_Stop()
  - - ADC의 레귤러 그룹의 동작을 정지한다.
- **HAL_ADC_PollForConversion()**
  - **- ADC의 레귤러 그룹의 변환이 완료될 때 까지 기다린다.**
- · HAL_ADC_Start_IT()
  - - ADC의 레귤러 그룹의 동작을 시작하고 관련 인터럽트를 인에이블시킨다.
- · HAL_ADC_Stop_IT()
  - - ADC의 레귤러 그룹의 동작을 정지하고 관련 인터럽트를 정지시킨다.
- · HAL_ADC_Start_DMA()
  - ADC의 레귤러 그룹의 동작을 시작하고 관련 DMA를 사용한다.
- · HAL_ADC_Stop_DMA()
  - ADC의 레귤러 그룹의 동작을 정지하고 관련 DMA를 중지한다

50

- · **HAL_ADC_GetValue()**
  - **- 레귤러 그룹의 ADC 변환 결과 값을 반환**
- · HAL_ADC_IRQHandler()
  - - ADC 관련 인터럽트의 핸들러
- · HAL_ADC_Stop_IT()
  - - ADC의 레귤러 그룹의 동작을 정지하고 관련 인터럽트를 정지시킨다.
- · HAL_ADC_Start_DMA()
  - - ADC의 레귤러 그룹의 동작을 시작하고 관련 DMA를 사용한다.
- · HAL_ADC_Stop_DMA()
  - - ADC의 레귤러 그룹의 동작을 정지하고 관련 DMA를 중지한다
- · HAL_ADC_GetValue()
  - - 레귤러 그룹의 ADC 변환 결과 값을 반환
- · HAL_ADC_IRQHandler()
  - - ADC 관련 인터럽트의 핸들러

❖ 3) 주변장치 제어 함수

- HAL_ADC_ConfigChannel()
  - ADC의 레귤러 그룹의 채널의 동작조건을 설정
- HAL_ADC_AnalogWDGConfig()
  - Analog watchdog의 동작조건을 설정

❖ 4) 주변장치 상태 함수

- 동작중인 ADC의 상태를 확인할 수 있는 함수는 다음과 같다.
- HAL_ADC_GetState()
  - ADC가 변환중인지, 변환이 완료되었는지 등의 상태값을 읽어온다.
- HAL_ADC_GetError()
  - ADC의 에러 코드를 읽어온다.

52

❖ 5) ADCEx 관련 함수

- ➢ HAL_ADCEx_Calibration_Start()
  - ADC의 자동 Calibration을 실행,ADC가 비활성화된 상태에서 사용. 즉, HAL_ADC_Start() 함수의 실행 이전이나, HAL_ADC_Stop() 함수의 실행 이후에 사용하여야 한다.

❖ 6) 콜백 함수 (Callback function)

- ➢ 콜백 함수는 다음과 같으며 이들은 반드시 사용자 프로그램에서 작성.
- ➢ HAL_ADC_ErrorCallback()
- ➢ HAL_ADC_LevelOutOfWindowCallback() (analog watchdog 콜백)
- ➢ HAL_ADC_ConvCpltCallback()
- ➢ HAL_ADC_ConvHalfCpltCallback
- ➢ HAL_ADCEx_InjectedConvCpltCallback()

구동용 함수의 사용 방법

❖ 1) ADC 관련 파라미터의 설정(CubeMX에서 생성)

- ➢ ADC 인터페이스를 설정한다.
- ➢ ADC 핀을 설정한다.
- ➢ __HAL_RCC_GPIOx_CLK_ENABLE()를 사용하여 ADC GPIOs 클럭을 활성화
- ➢ HAL_GPIO_Init()를 사용하여 ADC 핀들을 analog 모드로 설정
- ➢ ADC 인터럽트를 사용하는 경우는 다음을 설정한다.
- ➢ HAL_NVIC_EnableIRQ(ADCx_IRQn)를 사용하여 ADC의 NVIC를 설정
- ➢ ADC 인터럽트 핸들러 함수인 HAL_ADC_IRQHandler()를 ADCx_IRQHandler()에 상응되는 ADC 인터럽트에 사용
- ➢ DMA를 사용하는 경우는 다음을 설정한다.
- ➢ HAL_DMA_Init()를 사용하여 DMA(DMA 채널, 일반 또는 순환)를 설정
- ➢ HAL_NVIC_EnableIRQ(DMAx_Channelx_IRQn)를 사용하여 DMA의 NVIC 설정
- ➢ ADC 인터럽트 핸들러 함수인 HAL_ADC_IRQHandler()를 DMAx_Channelx_IRQHandler()에 상응되는 DMA 인터럽트에 사용

54

❖2) ADC, regular/injected, channel 파라미터의 설정

- HAL_ADC_Init()을 사용하여 ADC 파라미터와 레귤러 그룹의 파라미터를 설정
- HAL_ADC_ConfigChannel()을 사용하여 채널을 설정
- HAL_ADCEx_InjectedConfigChannel()을 사용하여 인젝티드 그룹의 파라미터 및 채널들을 설정
- HAL_ADC_AnalogWDGConfig()을 사용하여 아날로그 워치독 파라미터를 설정
- HAL_ADCEx_MultiModeConfigChannel()을 사용하여 ADC의 멀티모드를 사용

❖ 3) ADC 변환 수행

- HAL_ADCEx_Calibration_Start()을 사용하여 자동적으로 ADC calibration을 수행.
- Polling, Interruption, DMA의 모드를 사용하기 위해 ADC 드라이버를 사용.
- 3-1) Polling 방식의 경우
  - ADC 주변장치를 활성화하고 HAL_ADC_Start()를 사용하여 변환을 시작.
  - 레귤러 HAL_ADC_PollForConversion() /인젝티드 그룹 : HAL_ADCEx_InjectedPollForConversion() 을 사용하여 ADC 변환이 완료될 때 까지 대기
  - HAL_ADC_GetValue() (인젝티드 그룹 : HAL_ADCEx_InjectedGetValue())을 사용하여 변환 결과 값을 가져온다.
  - HAL_ADC_Stop()을 사용하여 변환을 멈추고 ADC 주변장치를 비활성화 한다.
- 3-2) Interruption 방식의 경우
  - HAL_ADC_Start_IT()을 사용하여 ADC 주변장치를 활성화하고 변환을 시작한다.
  - 레귤러그룹 HAL_ADC_ConvCpltCallback() /인젝티드 그룹 : HAL_ADCEx_InjectedConvCpltCallback())으로ADC 변환 완료를 대기
  - 레귤러그룹 HAL_ADC_GetValue() /인젝티드 그룹 : HAL_ADCEx_InjectedGetValue())을 사용하여 변환 결과 값을 획득
  - HAL_ADC_Stop_IT()을 사용하여 변환을 정지하고 ADC 주변장치를 비활성화.

❖ 3-3) DMA 방식의 경우

- HAL_ADC_Start_DMA()을 사용하여 ADC 주변창지를 활성화하고 변환을 시작
- HAL_ADC_ConvCpltCallback() (인젝티드 그룹 : HAL_ADCEx_InjectedConvCpltCallback())을 사용하여 ADC 변환 완료를 대기
- 변환 결과는 DMA의 도착지점의 주소 값으로 전달
- HAL_ADC_Stop_DMA()을 사용하여 변환을 정지하고 ADC 주변장치를 비활성화

❖ 3-4) DMA : ADC 멀티모드의 경우

- HAL_ADC_Start()을 사용하여 ADC 주변장치(slave)를 활성화하고 변환을 시작한다.
- HAL_ADCEx_MultiModeStart_DMA()을 사용하여 ADC 주변장치(master)를 활성화하고 변환을 시작
- HAL_ADC_ConvCpltCallback() 또는 HAL_ADC_ConvHalfCpltCallback()을 사용하여 ADC 변환을 대기
- 변환 결과 값이 자동적으로 DMA 도착 주소 값으로 전달된다.
- HAL_ADCEx_MultiModeStop_DMA()을 사용하여 ADC 주변장치(master)를 비활성화하고 변환을 정지
- HAL_ADC_Stop_IT()을 사용하여 ADC 주변장치(slave)를 비활성화하고 변환을 정지

주요 함수의 상세 설명

ADC 구동용 함수

1) 초기화(Initialization) 및 초기화 해제(de-initialization)용 함수

- HAL_ADC_Init (ADC_HandleTypeDef * hadc)
  - 함수 설명
    - 파라미터 hadc의 설정값에 따라 ADC와 Regular 그룹을 초기화한다.
  - 파라미터
    - hadc : ADC handle
  - 반환 값
    - HAL status
- HAL_ADC_DeInit (ADC_HandleTypeDef * hadc)
  - 함수 설명
    - ADC MSP와 ADC 주변장치를 해제한다.
  - 파라미터
    - hadc : ADC handle
  - 반환 값
    - HAL status

59

HAL_ADC_Start (ADC_HandleTypeDef * hadc)

- 함수 설명
  - ADC를 활성화하고 레귤러 그룹의 변환을 시작한다.
- 파라미터
  - hadc : ADC handle
- 반환 값
  - HAL status

HAL_ADC_Stop (ADC_HandleTypeDef * hadc)

- 함수 설명
  - ADC를 비활성화하고 변환을 정지한다.
- 파라미터
  - hadc : ADC handle
- 반환 값
  - HAL status

- **HAL_ADC_PollForConversion (ADC_HandleTypeDef * hadc, uint32_t Timeout)**
  - 함수 설명
    - 레귤러 그룹의 변환이 완료될 때까지 기다린다.
  - 파라미터
    - hadc : ADC handle
    - Timeout : 타임 아웃 시간(단위는 msec)
  - 반환 값
    - HAL status

HAL_ADC_Start_IT (ADC_HandleTypeDef * hadc)

함수 설명

- - ADC를 활성화하고 레귤러 그룹의 변환을 시작한다. 그리고 관련 인터럽트를 인에이블시킨다.

파라미터

- - hadc : ADC handle

반환 값

- - 없음

HAL_ADC_Stop_IT (ADC_HandleTypeDef * hadc)

함수 설명

- ADC를 비활성화하고 레귤러 그룹의 변환을 정지한다. 그리고 관련 인터럽트를 디스에이블시킨다.

파라미터

- hadc : ADC handle

반환 값

- 없음

- **HAL_ADC_Start_DMA (ADC_HandleTypeDef * hadc, uint32_t* pData, uint32_t Length)**
  - 함수 설명
    - ADC를 활성화하고 레귤러 그룹의 변환을 시작한다. 그리고 DMA를 인에이블하고 변환 결과를 DMA를 통해 전달한다.
  - 파라미터
    - hadc : ADC handle
    - pData : 해당 데이터
    - Length : 데이터 크기
  - 반환 값
    - 없음
- **HAL_ADC_Stop_DMA (ADC_HandleTypeDef * hadc)**
  - 함수 설명
    - ADC를 비활성화하고 레귤러 그룹의 변환을 정지한다. 그리고 DMA를 디스에이블한다.
  - 파라미터
    - hadc : ADC handle
  - 반환 값
    - 없음

- **HAL_ADC_GetValue (ADC_HandleTypeDef * hadc)**
  - 함수 설명
    - - 레귤러 그룹의 ADC 변환 결과 값을 읽어온다.
  - 파라미터
    - - hadc : ADC handle
  - 반환 값
    - - ADC 변환 결과 값
- **HAL_ADC_IRQHandler (ADC_HandleTypeDef * hadc)**
  - 함수 설명
    - - ADC 인터럽트의 처리를 위한 인터럽트 핸들러 함수
  - 파라미터
    - - hadc : ADC handle
  - 반환 값
    - - 없음

HAL_ADC_ConvCpltCallback (ADC_HandleTypeDef * hadc)

- 함수 설명
  - - DC가 Non blocking 모드로 동작할 경우 변환이 완료되면 호출되는 callback 함수
- 파라미터
  - - hadc : ADC handle
- 반환 값
  - - 없음

4) 주변장치 상태 함수

- **HAL_ADC_GetState(ADC_HandleTypeDef * hadc)**
  - 함수 설명
    - ADC의 동작상태 값을 반환한다.
  - 파라미터
    - hadc : ADC handle
  - 반환 값
    - HAL state
- **HAL_ADC_GetError (ADC_HandleTypeDef * hadc)**
  - 함수 설명
    - ADC의 에러 상태를 반환한다.
  - 파라미터
    - hadc : ADC handle
  - 반환 값
    - ADC 에러 코드

ADC 구동용 매크로(macro) 및 정의(define)

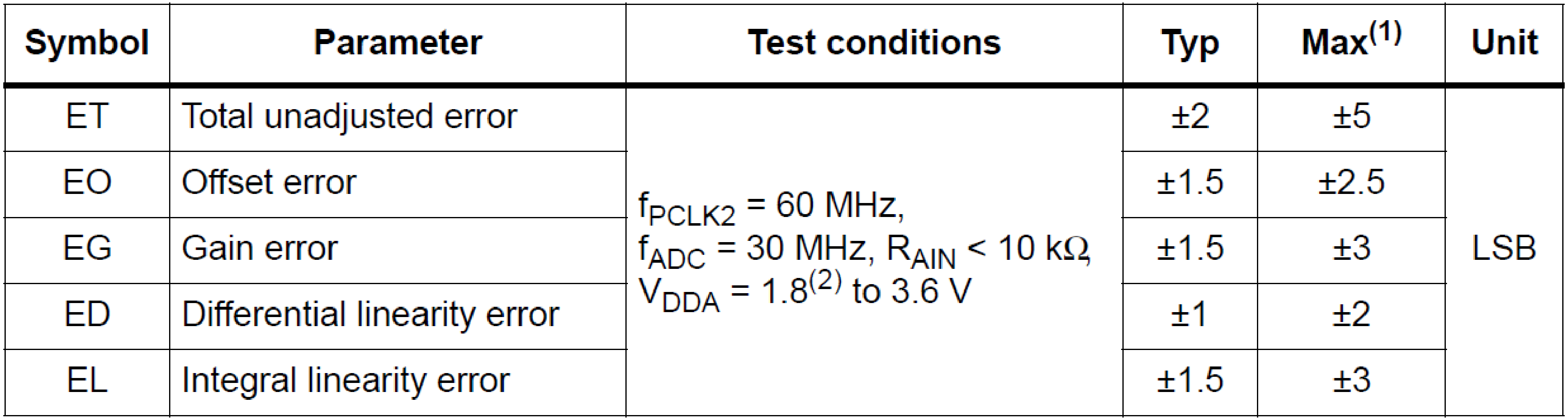
- **ADC 변환 그룹**
  - ADC_REGULAR_GROUP
  - ADC_INJECTED_GROUP
  - ADC_REGULAR_INJECTED_GROUP
- **ADC 에러 코드**
  - HAL_ADC_ERROR_NONE
  - HAL_ADC_ERROR_INTERNAL
  - HAL_ADC_ERROR_OVR
  - HAL_ADC_ERROR_DMA
- **ADC 매크로**
  - __HAL_ADC_ENABLE
  - __HAL_ADC_DISABLE
  - __HAL_ADC_ENABLE_IT
  - __HAL_ADC_DISABLE_IT
  - __HAL_ADC_GET_IT_SOURCE
  - __HAL_ADC_GET_FLAG
  - __HAL_ADC_CLEAR_FLAG
  - __HAL_ADC_RESET_HANDLE_STATE

12-bit ADC characteristics

Ds8626

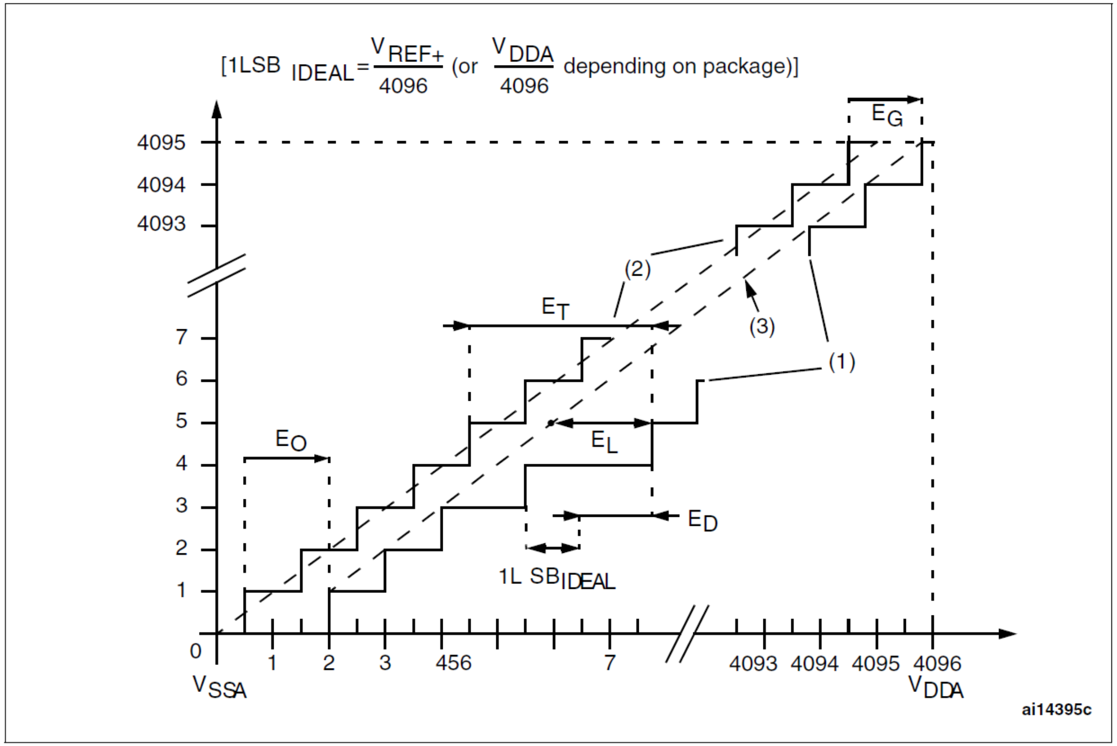
$$R_{AIN} = \frac{(k - 0.5)}{f_{ADC} \times C_{ADC} \times \ln(2^{N+2})} - R_{ADC}$$

Table 68. ADC accuracy at f_{ADC} = 30 MHz



Symbol	Parameter	Test conditions	Typ	Max[1]	Unit
ET	Total unadjusted error	f_{PCLK2} = 60 MHz, f_{ADC} = 30 MHz, R_{AIN} < 10 kΩ, V_{DDA} = 1.8[2] to 3.6 V	±2	±5	LSB
EO	Offset error		±1.5	±2.5	
EG	Gain error		±1.5	±3	
ED	Differential linearity error		±1	±2	
EL	Integral linearity error		±1.5	±3	

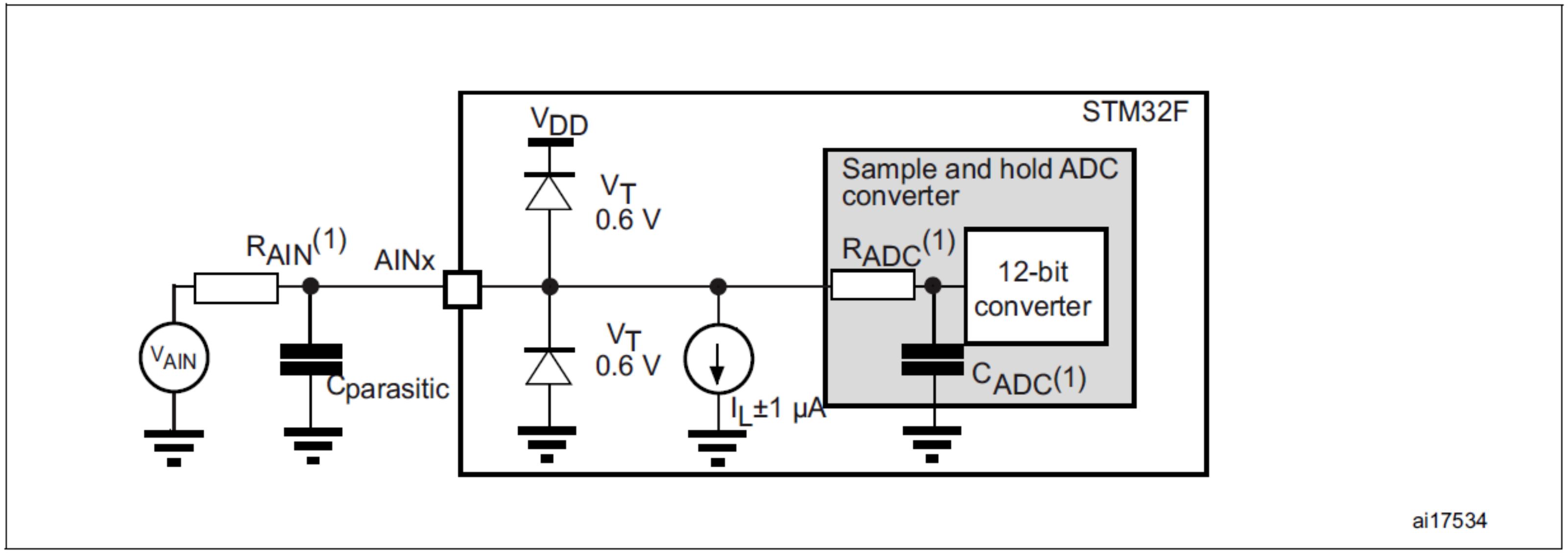
Figure 49. ADC accuracy characteristics



1. See also *Table 68*.
2. Example of an actual transfer curve.
3. Ideal transfer curve.
4. End point correlation line.
5. E_T = Total Unadjusted Error: maximum deviation between the actual and the ideal transfer curves.
   EO = Offset Error: deviation between the first actual transition and the first ideal one.
   EG = Gain Error: deviation between the last ideal transition and the last actual one.
   ED = Differential Linearity Error: maximum deviation between actual steps and the ideal one.
   EL = Integral Linearity Error: maximum deviation between any actual transition and the end point correlation line.

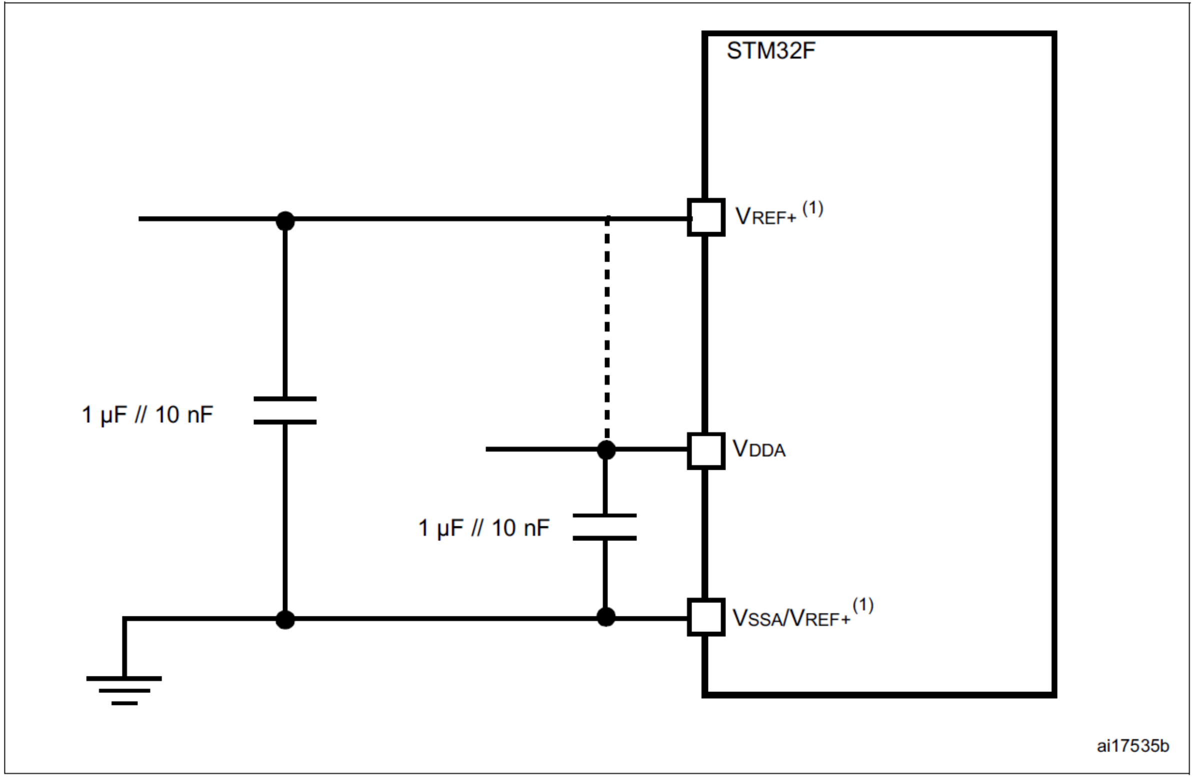
70

Figure 50. Typical connection diagram using the ADC



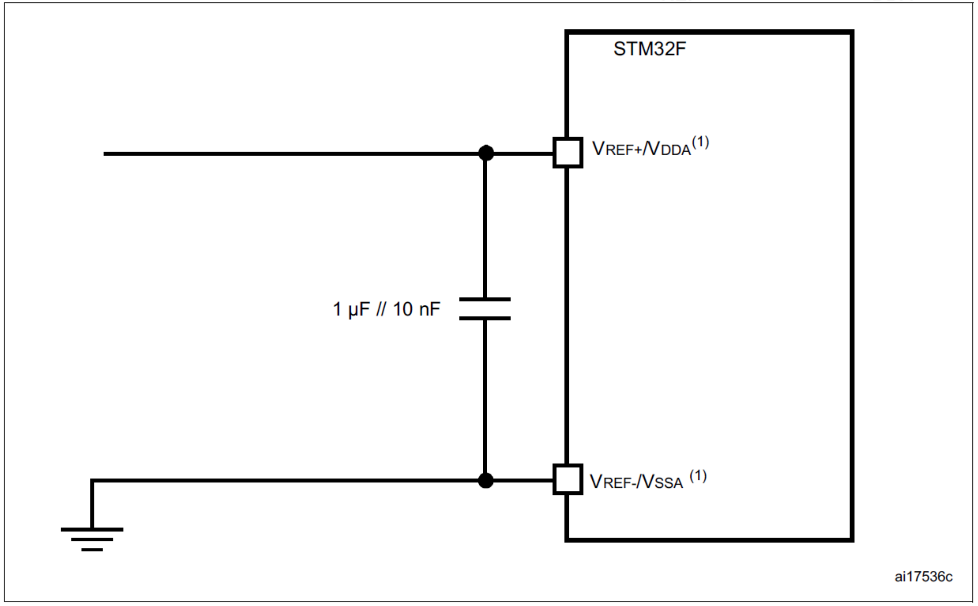
1. Refer to *Table 67* for the values of R_{AIN}, R_{ADC} and C_{ADC}.
2. $C_{parasitic}$ represents the capacitance of the PCB (dependent on soldering and PCB layout quality) plus the pad capacitance (roughly 5 pF). A high $C_{parasitic}$ value downgrades conversion accuracy. To remedy this, f_{ADC} should be reduced.

Figure 51. Power supply and reference decoupling (V_{REF+} not connected to V_{DDA})



1. V_{REF+} and V_{REF-} inputs are both available on UFBGA176. V_{REF+} is also available on LQFP100, LQFP144, and LQFP176. When V_{REF+} and V_{REF-} are not available, they are internally connected to V_{DDA} and V_{SSA}.

Figure 52. Power supply and reference decoupling (V_{REF+} connected to V_{DDA})



1. V_{REF+} and V_{REF-} inputs are both available on UFBGA176. V_{REF+} is also available on LQFP100, LQFP144, and LQFP176. When V_{REF+} and V_{REF-} are not available, they are internally connected to V_{DDA} and V_{SSA}.

Temperature sensor characteristics

74

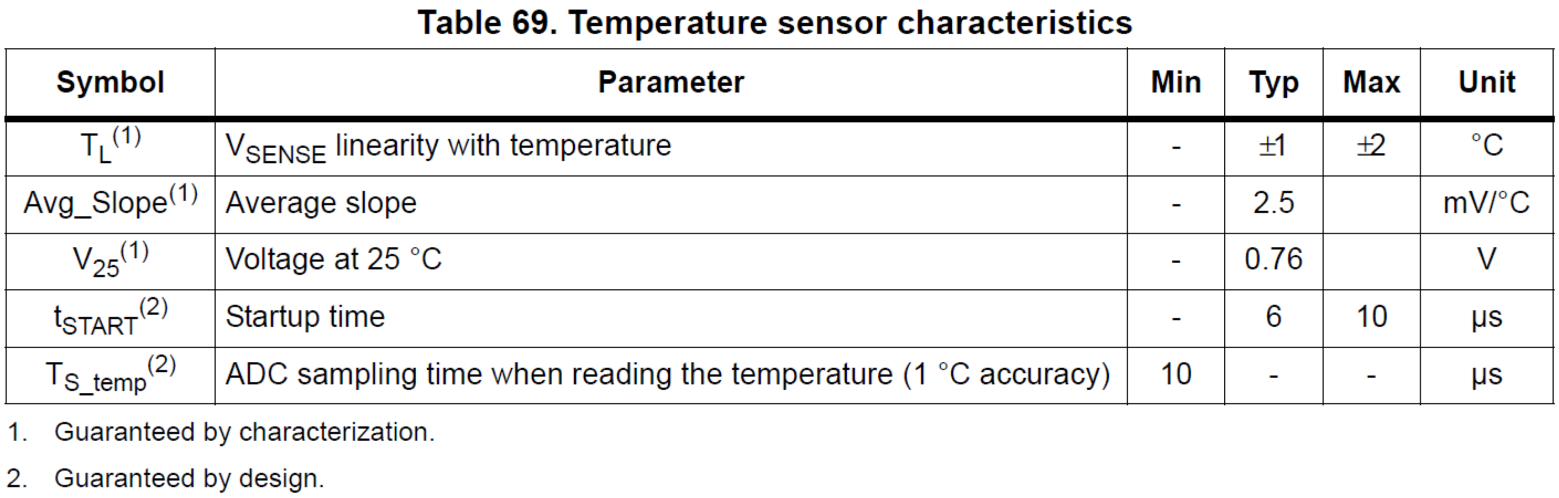


Table 69. Temperature sensor characteristics

Symbol	Parameter	Min	Typ	Max	Unit
T_L[1]	V_{SENSE} linearity with temperature	-	±1	±2	°C
Avg_Slope[1]	Average slope	-	2.5		mV/°C
V_{25}[1]	Voltage at 25 °C	-	0.76		V
t_{START}[2]	Startup time	-	6	10	µs
T_{S_temp}[2]	ADC sampling time when reading the temperature (1 °C accuracy)	10	-	-	µs

1. Guaranteed by characterization.
2. Guaranteed by design.

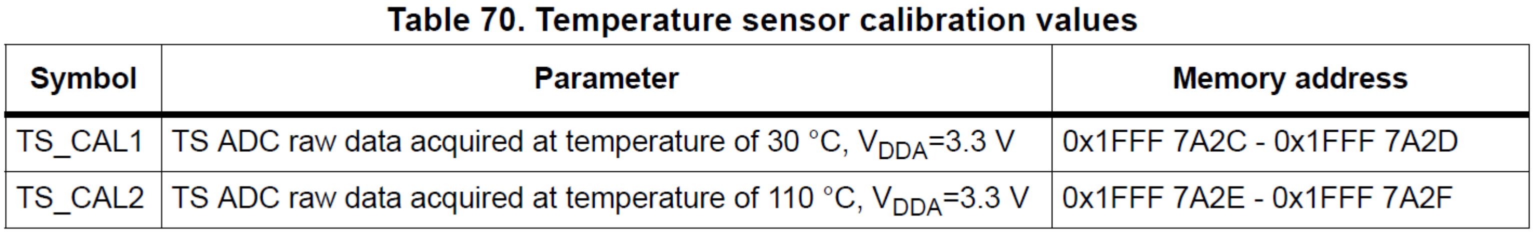


Table 70. Temperature sensor calibration values

Symbol	Parameter	Memory address
TS_CAL1	TS ADC raw data acquired at temperature of 30 °C, V_{DDA}=3.3 V	0x1FFF 7A2C - 0x1FFF 7A2D
TS_CAL2	TS ADC raw data acquired at temperature of 110 °C, V_{DDA}=3.3 V	0x1FFF 7A2E - 0x1FFF 7A2F

VBAT monitoring characteristics

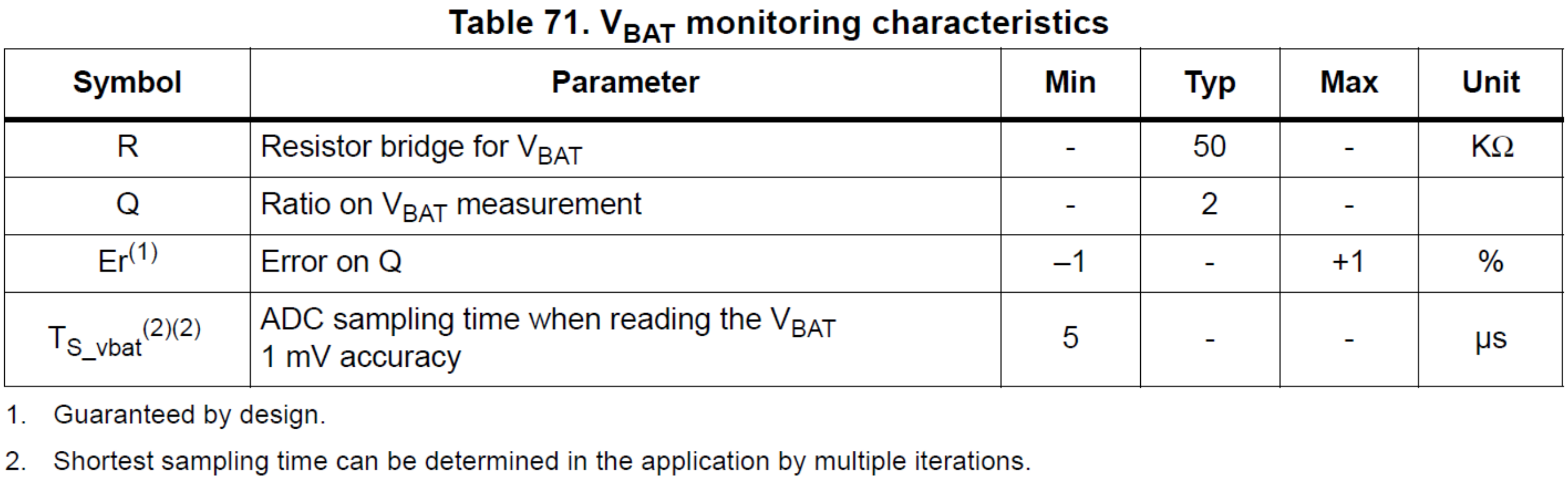


Table 71. V_{BAT} monitoring characteristics

Symbol	Parameter	Min	Typ	Max	Unit
R	Resistor bridge for V_{BAT}	-	50	-	KΩ
Q	Ratio on V_{BAT} measurement	-	2	-	
Er[1]	Error on Q	–1	-	+1	%
T_{S_vbat}[2][2]	ADC sampling time when reading the V_{BAT} 1 mV accuracy	5	-	-	µs

[1] Guaranteed by design.

[2] Shortest sampling time can be determined in the application by multiple iterations.

Embedded reference voltage

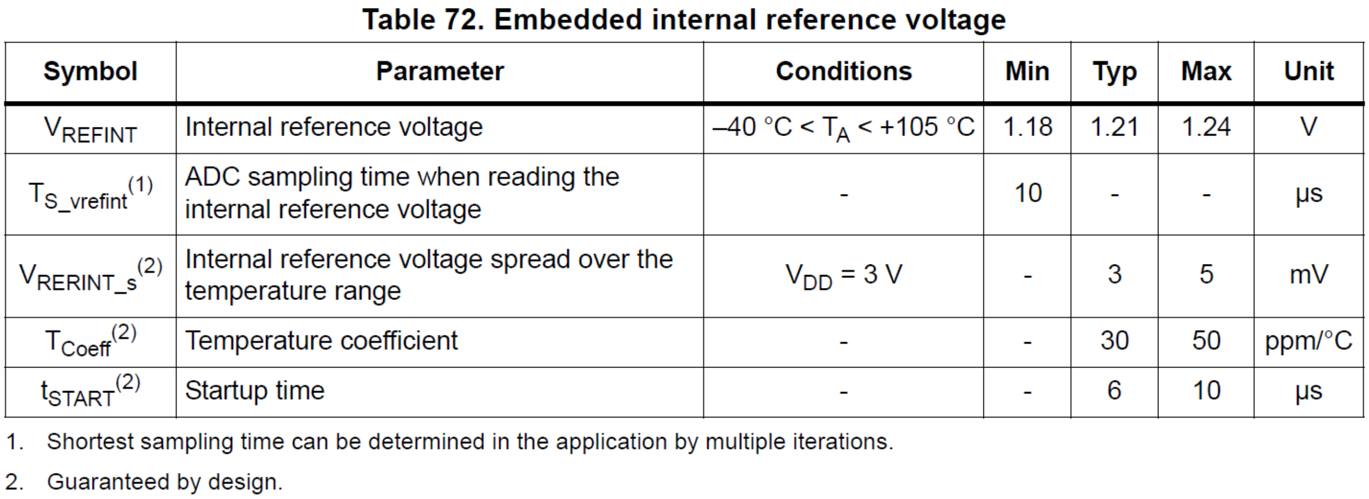


Table 72. Embedded internal reference voltage

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{REFINT}	Internal reference voltage	-40 °C < T_A < +105 °C	1.18	1.21	1.24	V
$T_{S\_vrefint}$[1]	ADC sampling time when reading the internal reference voltage	-	10	-	-	µs
V_{RERINT_s}[2]	Internal reference voltage spread over the temperature range	$V_{DD} = 3$ V	-	3	5	mV
T_{Coeff}[2]	Temperature coefficient	-	-	30	50	ppm/°C
t_{START}[2]	Startup time	-	-	6	10	µs

1. Shortest sampling time can be determined in the application by multiple iterations.
2. Guaranteed by design.

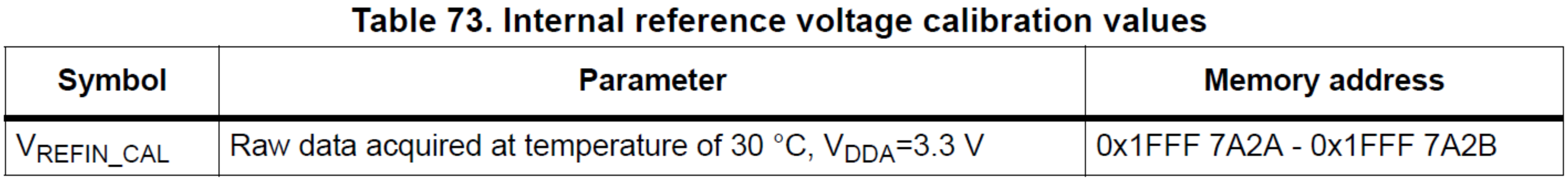


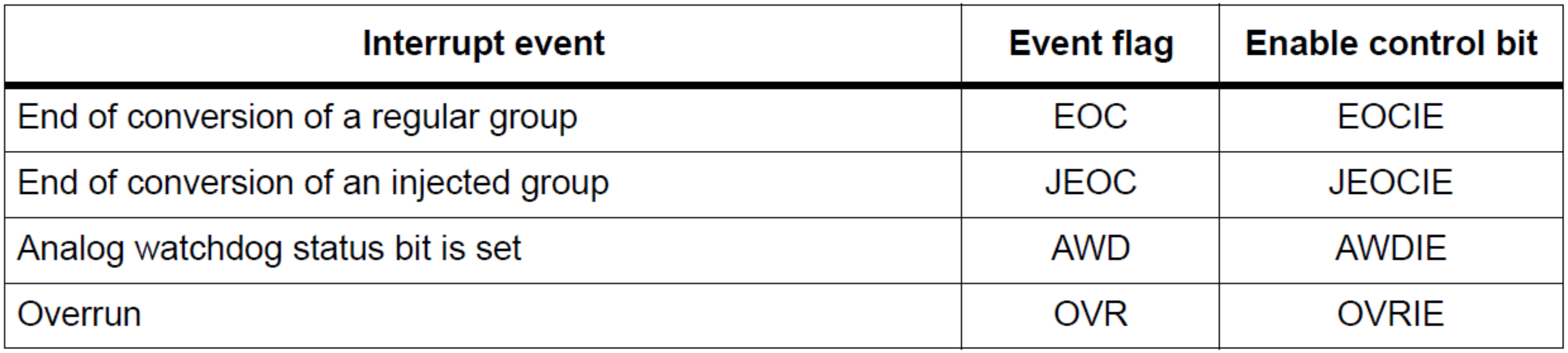
Table 73. Internal reference voltage calibration values

Symbol	Parameter	Memory address
V_{REFIN_CAL}	Raw data acquired at temperature of 30 °C, V_{DDA}=3.3 V	0x1FFF 7A2A - 0x1FFF 7A2B

ADC interrupts

- ❖ An interrupt can be produced on the end of conversion for regular and injected groups, when the analog watchdog status bit is set and when the overrun status bit is set. Separate interrupt enable bits are available for flexibility.
- ❖ Two other flags are present in the ADC_SR register, but there is no interrupt associated with them:
  - ➢ • JSTRT (Start of conversion for channels of an injected group)
  - ➢ • STRT (Start of conversion for channels of a regular group)

Table 70. ADC interrupts



Interrupt event	Event flag	Enable control bit
End of conversion of a regular group	EOC	EOCIE
End of conversion of an injected group	JEOC	JEOCIE
Analog watchdog status bit is set	AWD	AWDIE
Overrun	OVR	OVRIE

Register

82

List of abbreviations for registers

- **read/write (rw)** : Software can read and write to these bits.
- **read-only (r)** : Software can only read these bits.
- **write-only (w)**: Software can only write to this bit. Reading the bit returns the reset value.
- **read/clear (rc_w1)** : Software can read as well as clear this bit by writing 1. Writing ‘0’ has no effect on the bit value.
- **read/clear (rc_w0)** : Software can read as well as clear this bit by writing 0. Writing ‘1’ has no effect on the bit value.
- **read/clear by read (rc_r)** : Software can read this bit. Reading this bit automatically clears it to ‘0’. Writing ‘0’ has no effect on the bit value.
- **read/set (rs)** : Software can read as well as set this bit. Writing ‘0’ has no effect on the bit value.
- **read-only write trigger (rt_w)** : Software can read this bit. Writing ‘0’ or ‘1’ triggers an event but has no effect on the bit value.
- **toggle (t)** : Software can only toggle this bit by writing ‘1’. Writing ‘0’ has no effect.
- **Reserved (Res.)** : Reserved bit, must be kept at reset value.

Glossary

- • The CPU core integrates two debug ports:
  - – JTAG debug port (JTAG-DP) provides a 5-pin standard interface based on the Joint Test Action Group (JTAG) protocol.
  - – SWD debug port (SWD-DP) provides a 2-pin (clock and data) interface based on the Serial Wire Debug (SWD) protocol.
    - For both the JTAG and SWD protocols, please refer to the Cortex®-M4 with FPU Technical Reference Manual
- • Word: data/instruction of 32-bit length.
- • Half word: data/instruction of 16-bit length.
- • Byte: data of 8-bit length.
- • Double word: data of 64-bit length.
- • IAP (in-application programming): IAP is the ability to reprogram the Flash memory of a microcontroller while the user program is running.
- • ICP (in-circuit programming): ICP is the ability to program the Flash memory of a microcontroller using the JTAG protocol, the SWD protocol or the bootloader while the device is mounted on the user application board.
- • I-Code: this bus connects the Instruction bus of the CPU core to the Flash instruction interface. Prefetch is performed on this bus.
- • D-Code: this bus connects the D-Code bus (literal load and debug access) of the CPU to the Flash data interface.
- • Option bytes: product configuration bits stored in the Flash memory.
- • OBL: option byte loader.
- • AHB: advanced high-performance bus.
- • CPU: refers to the Cortex®-M4 with FPU core.

84

ADC register map

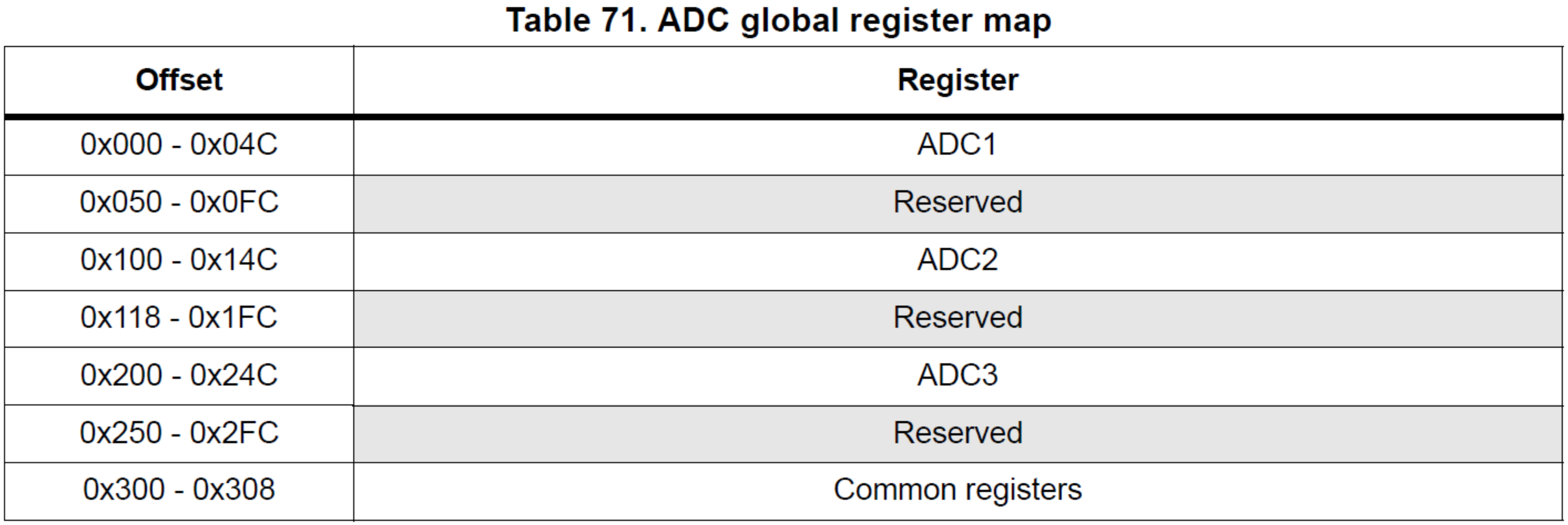
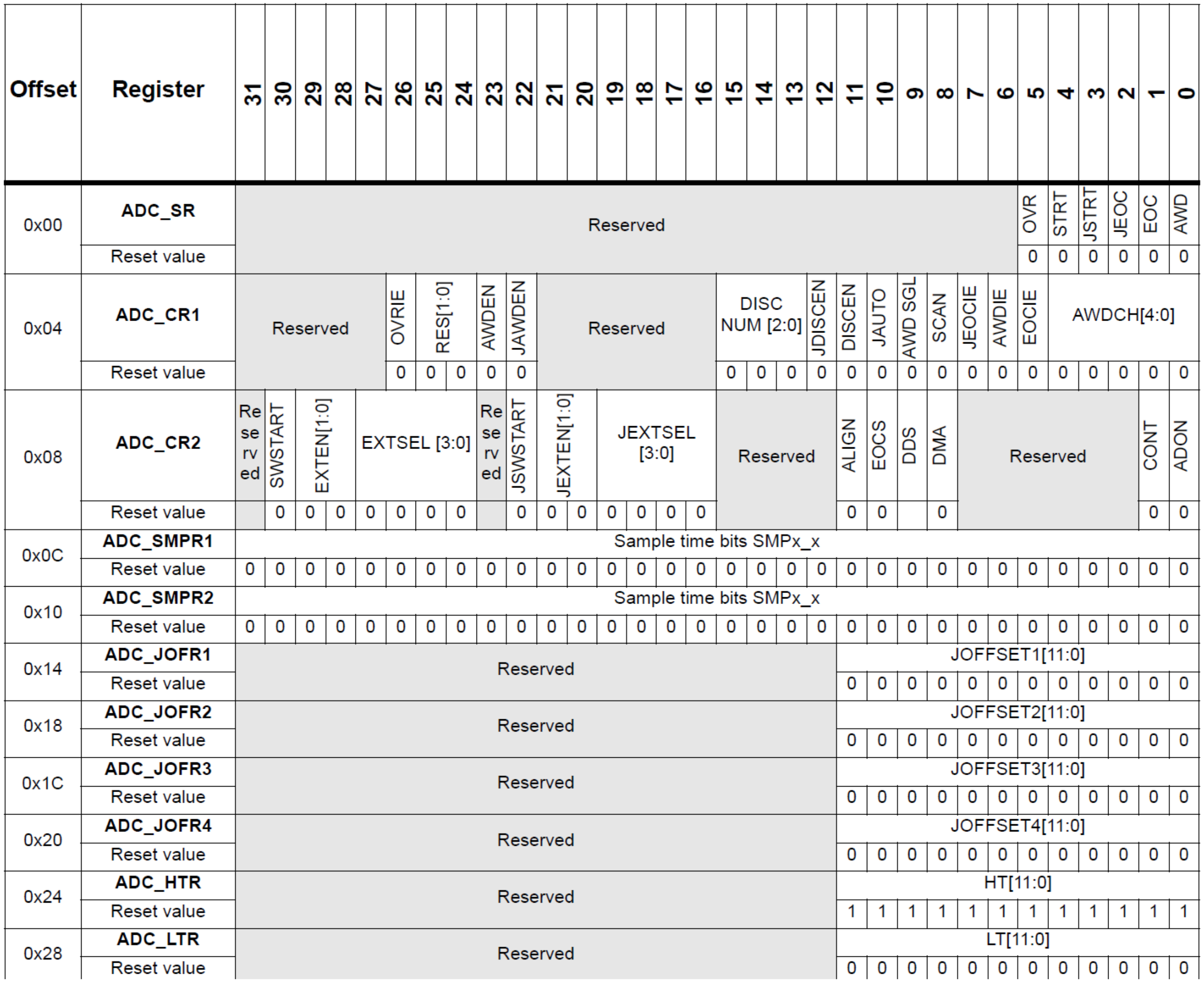


Table 71. ADC global register map

Offset	Register
0x000 - 0x04C	ADC1
0x050 - 0x0FC	Reserved
0x100 - 0x14C	ADC2
0x118 - 0x1FC	Reserved
0x200 - 0x24C	ADC3
0x250 - 0x2FC	Reserved
0x300 - 0x308	Common registers

Table 72. ADC register map and reset values for each ADC



Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	**ADC_SR**	Reserved																										OVR	STRT	JSTRT	JEOC	EOC	AWD
	Reset value																											0	0	0	0	0	0
0x04	**ADC_CR1**	Reserved					OVRIE	RES[1:0]		AWDEN	JAWDEN	Reserved						DISC NUM [2:0]			JDISCEN	DISCEN	JAUTO	AWD SGL	SCAN	JEOCIE	AWDIE	EOCIE	AWDCH[4:0]				
	Reset value						0	0	0	0	0							0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x08	**ADC_CR2**	Reserved	SWSTART	EXTEN[1:0]		EXTSEL [3:0]				Reserved	JSWSTART	JEXTEN[1:0]		JEXTSEL [3:0]				Reserved				ALIGN	EOCS	DDS	DMA	Reserved						CONT	ADON
	Reset value		0	0	0	0	0	0	0		0	0	0	0	0	0	0					0	0		0							0	0
0x0C	**ADC_SMPR1**	Sample time bits SMPx_x																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x10	**ADC_SMPR2**	Sample time bits SMPx_x																															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x14	**ADC_JOFR1**	Reserved																				JOFFSET1[11:0]											
	Reset value																					0	0	0	0	0	0	0	0	0	0	0	0
0x18	**ADC_JOFR2**	Reserved																				JOFFSET2[11:0]											
	Reset value																					0	0	0	0	0	0	0	0	0	0	0	0
0x1C	**ADC_JOFR3**	Reserved																				JOFFSET3[11:0]											
	Reset value																					0	0	0	0	0	0	0	0	0	0	0	0
0x20	**ADC_JOFR4**	Reserved																				JOFFSET4[11:0]											
	Reset value																					0	0	0	0	0	0	0	0	0	0	0	0
0x24	**ADC_HTR**	Reserved																				HT[11:0]											
	Reset value																					1	1	1	1	1	1	1	1	1	1	1	1
0x28	**ADC_LTR**	Reserved																				LT[11:0]											
	Reset value																					0	0	0	0	0	0	0	0	0	0	0	0

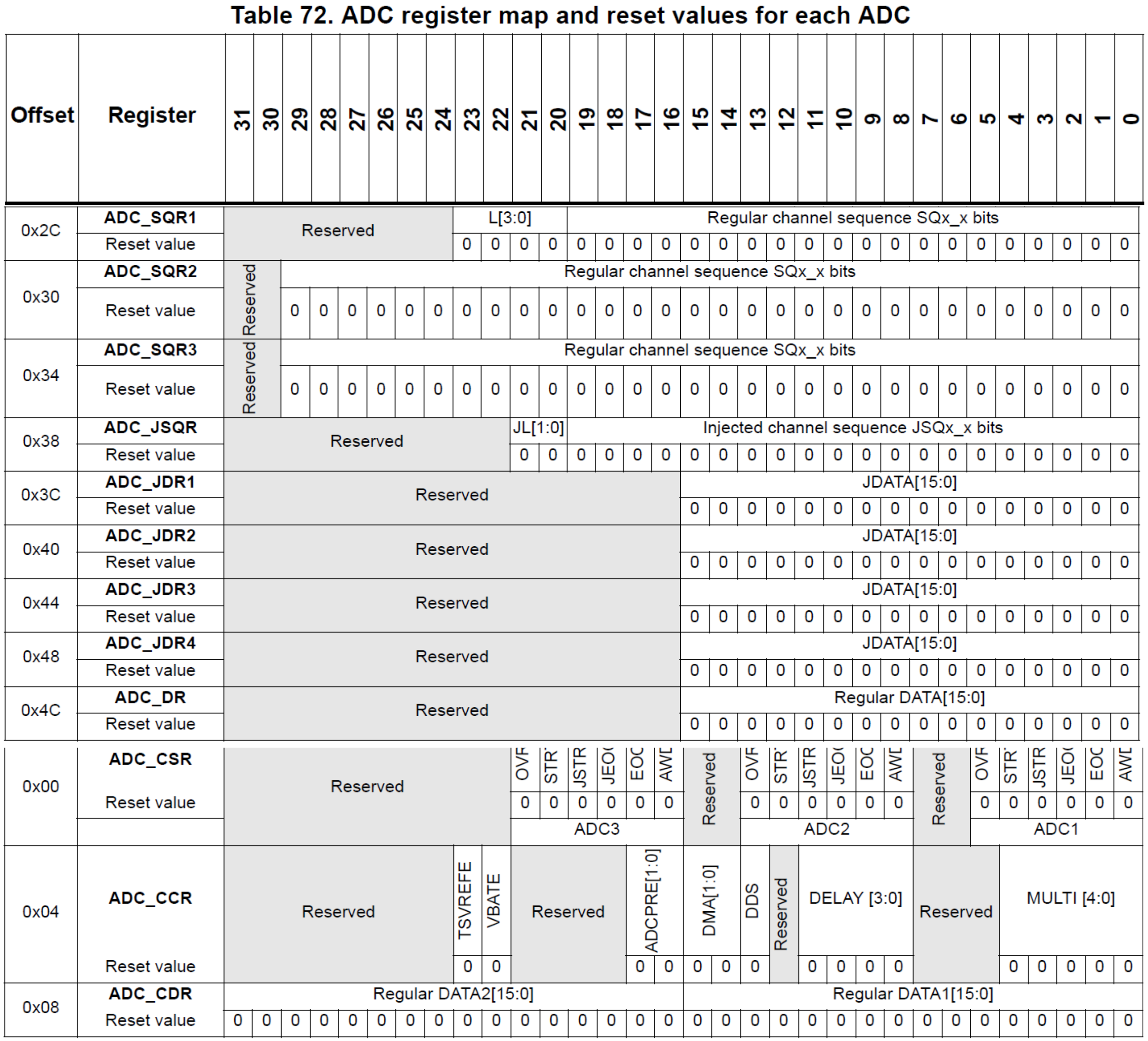


Table 72. ADC register map and reset values for each ADC

Offset	Register	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x2C	ADC_SQR1	Reserved								L[3:0]				Regular channel sequence SQx_x bits																			
	Reset value									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x30	ADC_SQR2	Reserved		Regular channel sequence SQx_x bits																													
	Reset value			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x34	ADC_SQR3	Reserved		Regular channel sequence SQx_x bits																													
	Reset value			0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x38	ADC_JSQR	Reserved										JL[1:0]		Injected channel sequence JSQx_x bits																			
	Reset value											0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x3C	ADC_JDR1	Reserved																JDATA[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x40	ADC_JDR2	Reserved																JDATA[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x44	ADC_JDR3	Reserved																JDATA[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x48	ADC_JDR4	Reserved																JDATA[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x4C	ADC_DR	Reserved																Regular DATA[15:0]															
	Reset value																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x00	ADC_CSR	Reserved										OVF	STR	JSTR	JEO	EOC	AWD	Reserved		OVF	STR	JSTR	JEO	EOC	AWD	Reserved		OVF	STR	JSTR	JEO	EOC	AWD
	Reset value											0	0	0	0	0	0			0	0	0	0	0	0			0	0	0	0	0	0
												ADC3								ADC2								ADC1					
0x04	ADC_CCR	Reserved								TSVREFE	VBATE	Reserved				ADCPRE[1:0]		DMA[1:0]		DDS	Reserved	DELAY [3:0]				Reserved			MULTI [4:0]				
	Reset value									0	0					0	0	0	0	0		0	0	0	0				0	0	0	0	0
0x08	ADC_CDR	Regular DATA2[15:0]																Regular DATA1[15:0]															
	Reset value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

13.13.1 ADC status register (ADC_SR)

Address offset: 0x00

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										OVR	STRT	JSTRT	JEOC	EOC	AWD
										rc_w0	rc_w0	rc_w0	rc_w0	rc_w0	rc_w0

13.13.2 ADC control register 1 (ADC_CR1)

Address offset: 0x04

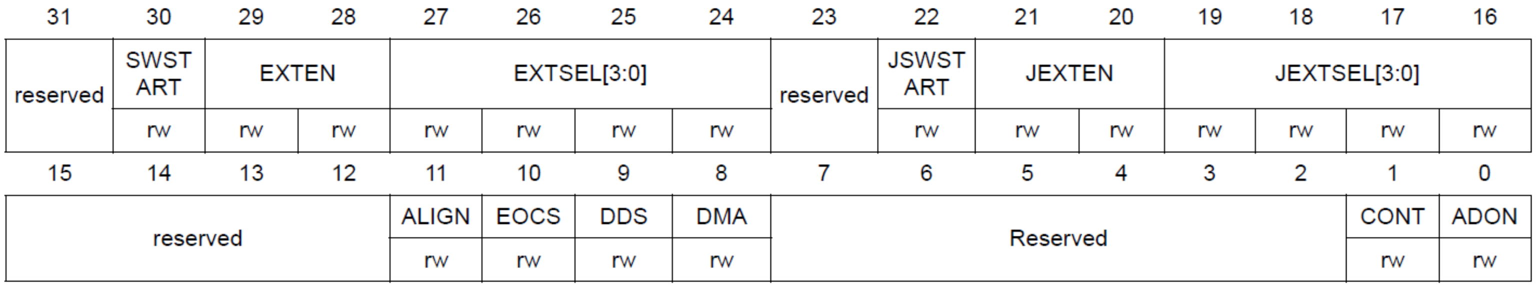
Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					OVRIE	RES		AWDEN	JAWDEN	Reserved					
					rw	rw	rw	rw	rw						
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DISCNUM[2:0]			JDISCEN	DISCEN	JAUTO	AWDSGL	SCAN	JEOCIE	AWDIE	EOCIE	AWDCH[4:0]				
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

13.13.3 ADC control register 2 (ADC_CR2)

Address offset: 0x08

Reset value: 0x0000 0000



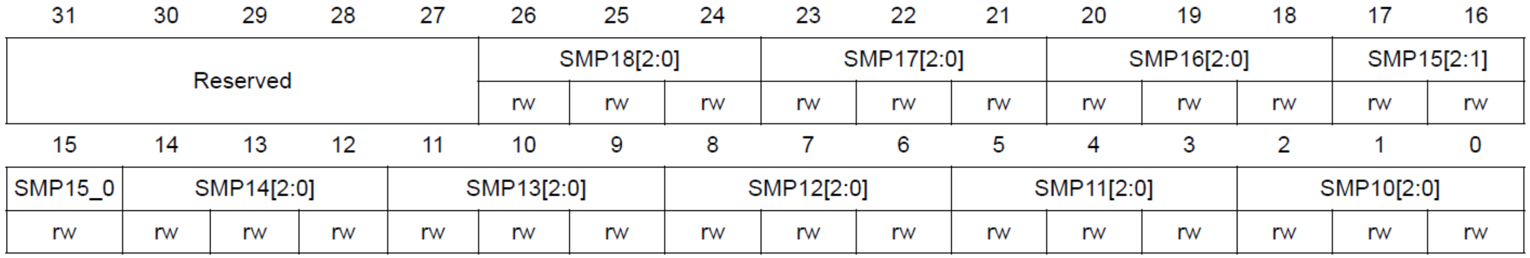
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
reserved	SWSTART	EXTEN		EXTSEL[3:0]				reserved	JSWSTART	JEXTEN		JEXTSEL[3:0]			
	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved				ALIGN	EOCS	DDS	DMA	Reserved						CONT	ADON
				rw	rw	rw	rw							rw	rw

13.13.4 ADC sample time register 1 (ADC_SMPR1)

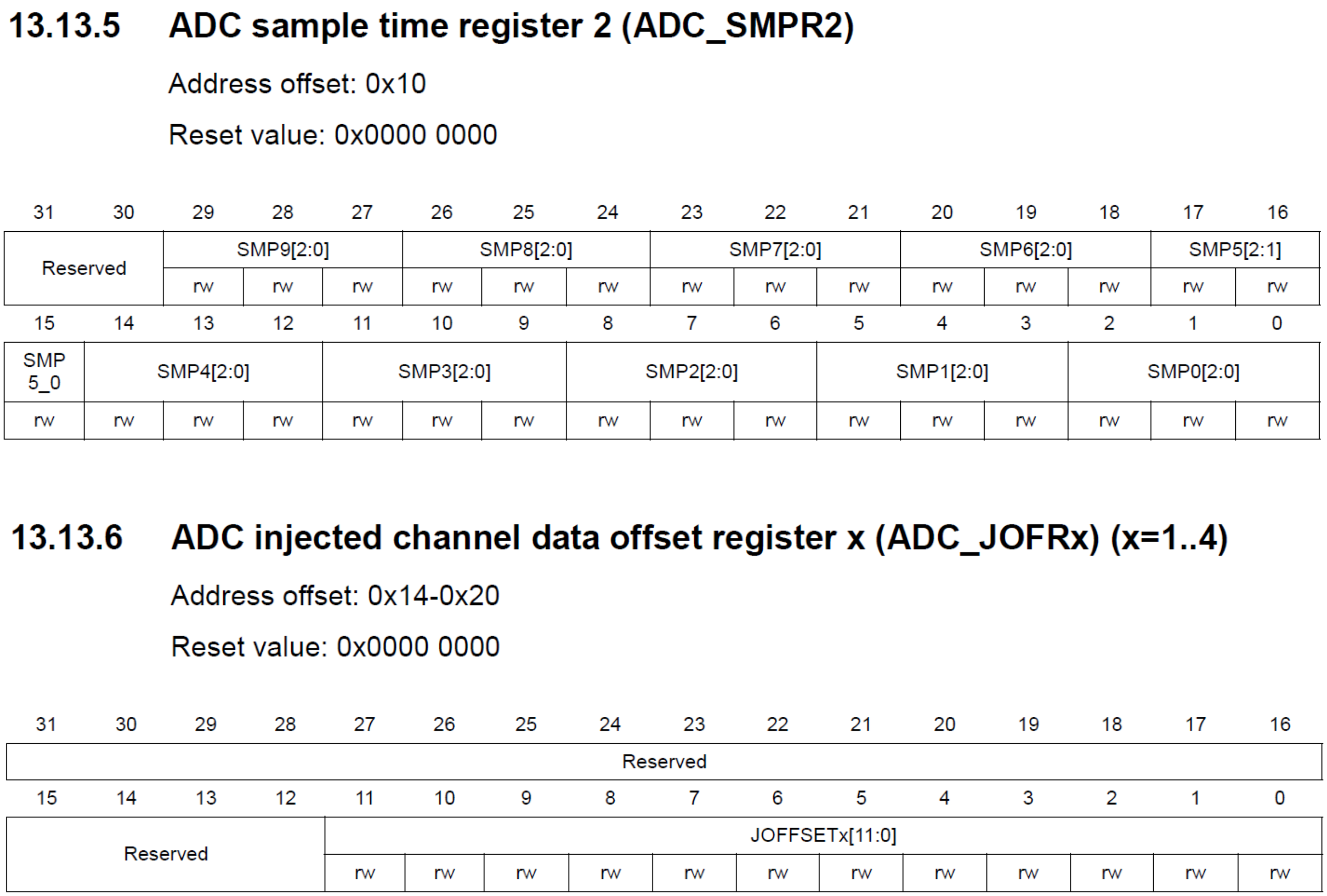
Address offset: 0x0C

Reset value: 0x0000 0000



31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved					SMP18[2:0]			SMP17[2:0]			SMP16[2:0]			SMP15[2:1]	
					rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMP15_0	SMP14[2:0]			SMP13[2:0]			SMP12[2:0]			SMP11[2:0]			SMP10[2:0]		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw



13.13.5 ADC sample time register 2 (ADC_SMPR2)

Address offset: 0x10

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		SMP9[2:0]			SMP8[2:0]			SMP7[2:0]			SMP6[2:0]			SMP5[2:1]	
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SMP 5_0	SMP4[2:0]			SMP3[2:0]			SMP2[2:0]			SMP1[2:0]			SMP0[2:0]		
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

13.13.6 ADC injected channel data offset register x (ADC_JOFRx) (x=1..4)

Address offset: 0x14-0x20

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				JOFFSETx[11:0]											
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

13.13.7 ADC watchdog higher threshold register (ADC_HTR)

Address offset: 0x24

Reset value: 0x0000 0FFF

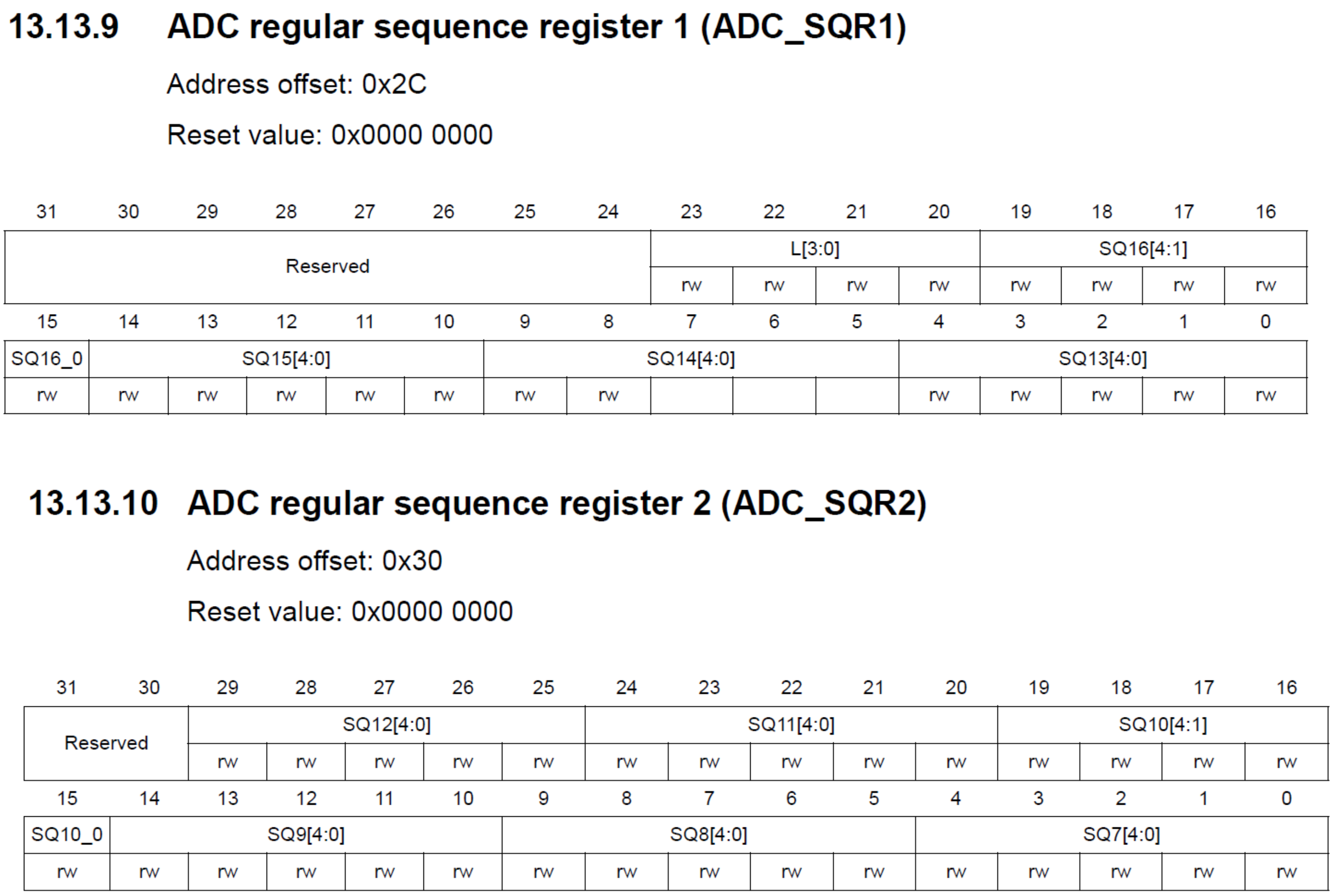
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				HT[11:0]											
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

13.13.8 ADC watchdog lower threshold register (ADC_LTR)

Address offset: 0x28

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				LT[11:0]											
				rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw



13.13.9 ADC regular sequence register 1 (ADC_SQR1)

Address offset: 0x2C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								L[3:0]				SQ16[4:1]			
								rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SQ16_0	SQ15[4:0]					SQ14[4:0]					SQ13[4:0]				
rw	rw	rw	rw	rw	rw	rw	rw				rw	rw	rw	rw	rw

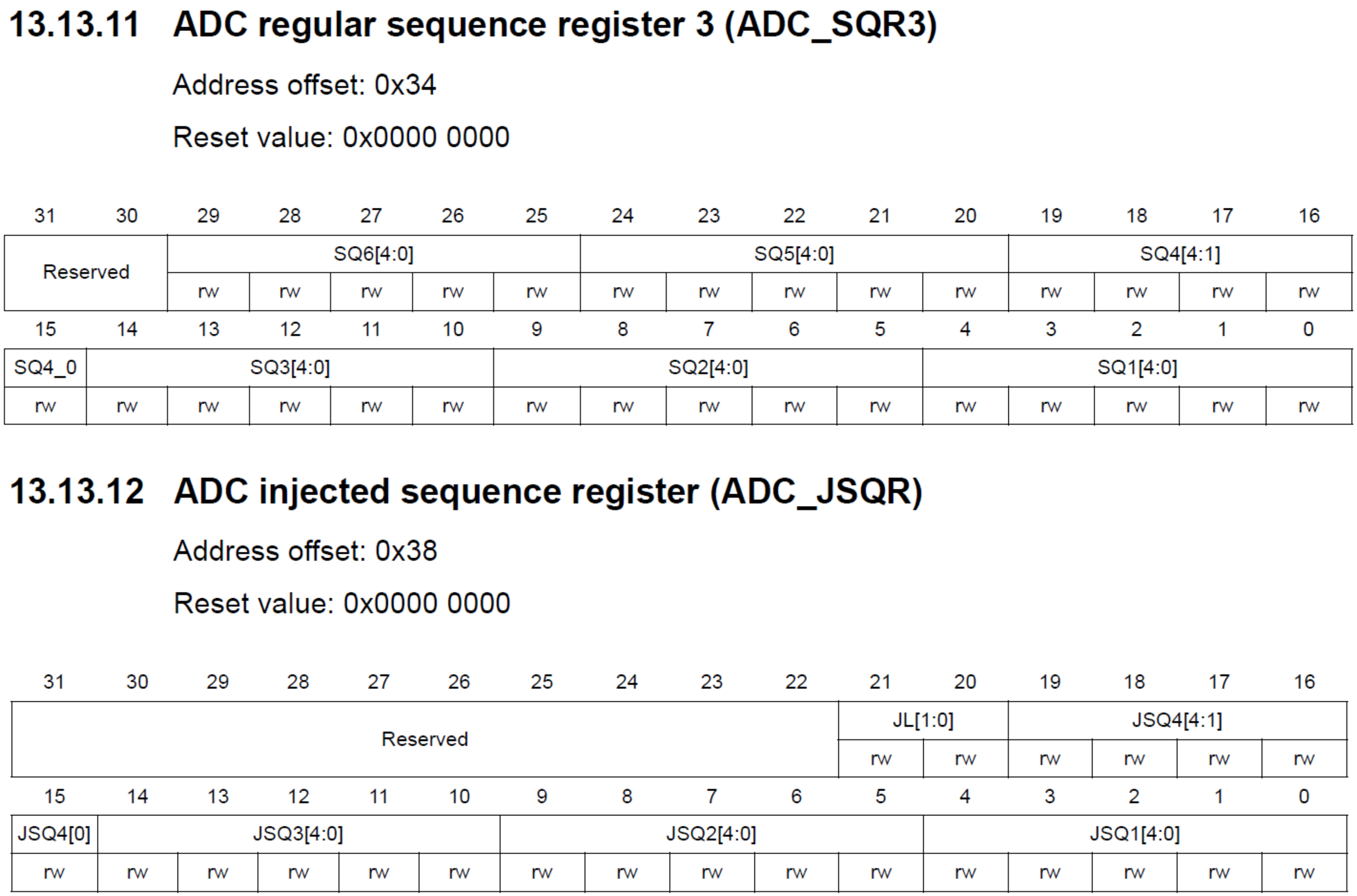
13.13.10 ADC regular sequence register 2 (ADC_SQR2)

Address offset: 0x30

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		SQ12[4:0]					SQ11[4:0]					SQ10[4:1]			
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SQ10_0	SQ9[4:0]					SQ8[4:0]					SQ7[4:0]				
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw



13.13.11 ADC regular sequence register 3 (ADC_SQR3)

Address offset: 0x34

Reset value: 0x0000 0000

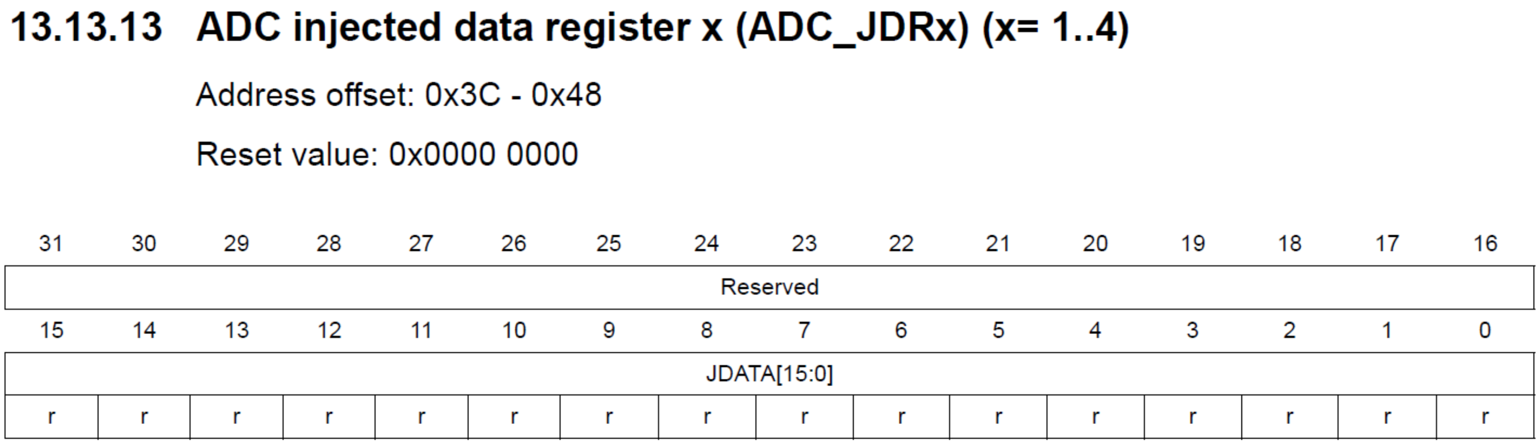
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		SQ6[4:0]					SQ5[4:0]					SQ4[4:1]			
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
SQ4_0	SQ3[4:0]					SQ2[4:0]					SQ1[4:0]				
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

13.13.12 ADC injected sequence register (ADC_JSQR)

Address offset: 0x38

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										JL[1:0]		JSQ4[4:1]			
										rw	rw	rw	rw	rw	rw
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
JSQ4[0]	JSQ3[4:0]					JSQ2[4:0]					JSQ1[4:0]				
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

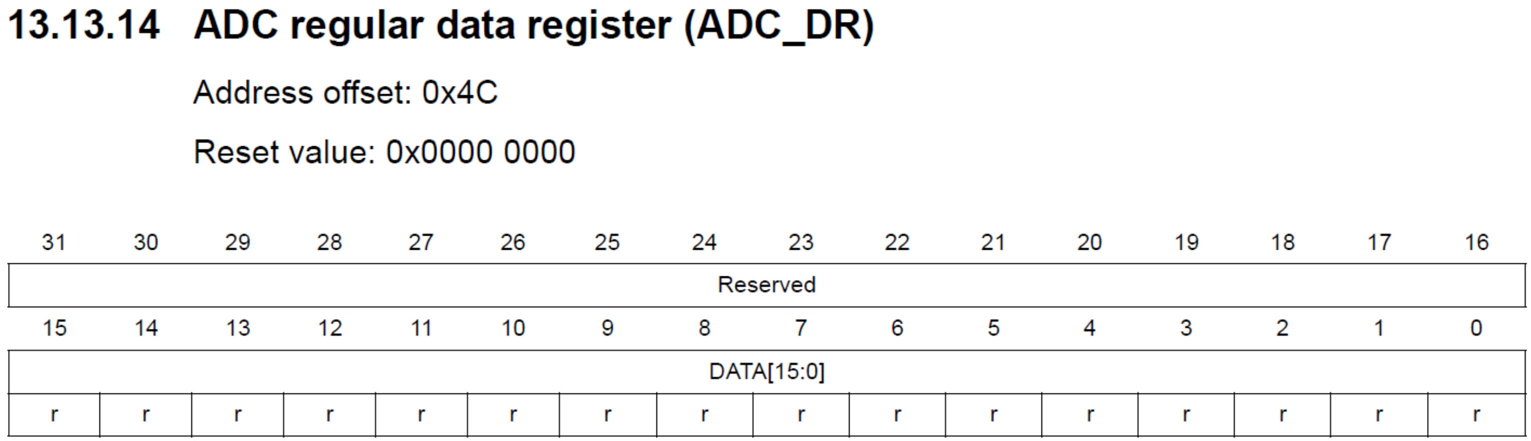


13.13.13 ADC injected data register x (ADC_JDRx) (x= 1..4)

Address offset: 0x3C - 0x48

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
JDATA[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r



13.13.14 ADC regular data register (ADC_DR)

Address offset: 0x4C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
DATA[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

13.13.15 ADC Common status register (ADC_CSR)

Address offset: 0x00 (this offset address is relative to ADC1 base address + 0x300)

Reset value: 0x0000 0000

This register provides an image of the status bits of the different ADCs. Nevertheless it is read-only and does not allow to clear the different status bits. Instead each status bit must be cleared by writing it to 0 in the corresponding ADC_SR register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										OVR3	STRT3	JSTRT3	JEOC 3	EOC3	AWD3
										ADC3					
										r	r	r	r	r	r
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		OVR2	STRT2	JSTRT 2	JEOC2	EOC2	AWD2	Reserved		OVR1	STRT1	JSTRT1	JEOC 1	EOC1	AWD1
		ADC2								ADC1					
		r	r	r	r	r	r			r	r	r	r	r	r

13.13.16 ADC common control register (ADC_CCR)

Address offset: 0x04 (this offset address is relative to ADC1 base address + 0x300)

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								TSVREFE	VBATE	Reserved				ADCPRE	
								rw	rw					rw	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DMA[1:0]		DDS	Res.	DELAY[3:0]				Reserved			MULTI[4:0]				
rw	rw	rw		rw	rw	rw	rw				rw	rw	rw	rw	rw

13.13.17 ADC common regular data register for dual and triple modes (ADC_CDR)

Address offset: 0x08 (this offset address is relative to ADC1 base address + 0x300)

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA2[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA1[15:0]															
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r