CT Übungsaufgaben
ADC

1. Offset errors

a. Draw the transfer function of an ideal 3-Bit ADC. V_{REF} is set to 2V.
b. That ADC has an offset error of -1.5 LSB.
    I. Draw the new transfer function
    II. Convert the LSB in volts
    III. Convert the offset error in Volts
    IV. Calculate the offset error in % of FSR

2. Programming the ADC

Consider the ADC registers and addresses given below. Assume that the ADCs are properly initialized and started for regular channel conversion.

Write C code (including correct register addresses) to do the following:

a. Wait until ADC1 conversion has completed.
b. Set an 8-bit variable (var2) to 0xFF if there was a loss of data on ADC2, otherwise reset that variable to 0.
c. Set ADC3 resolution to 10-bit.

13.13 ADC registers

Refer to *Section 1.1 on page 57* for a list of abbreviations used in register descriptions.

The peripheral registers must be written at word level (32 bits). Read accesses can be done by bytes (8 bits), half-words (16 bits) or words (32 bits).

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

Bits 31:6 Reserved, must be kept at reset value.

Bit 5 **OVR:** Overrun

This bit is set by hardware when data are lost (either in single mode or in dual/triple mode). It is cleared by software. Overrun detection is enabled only when DMA = 1 or EOCS = 1.

0: No overrun occurred

1: Overrun has occurred

Bit 4 **STRT:** Regular channel start flag

This bit is set by hardware when regular channel conversion starts. It is cleared by software.

0: No regular channel conversion started

1: Regular channel conversion has started

Bit 3 **JSTRT:** Injected channel start flag

This bit is set by hardware when injected group conversion starts. It is cleared by software.

0: No injected group conversion started

1: Injected group conversion has started

Bit 2 **JEOC:** Injected channel end of conversion

This bit is set by hardware at the end of the conversion of all injected channels in the group. It is cleared by software.

0: Conversion is not complete

1: Conversion complete

Bit 1 **EOC:** Regular channel end of conversion

This bit is set by hardware at the end of the conversion of a regular group of channels. It is cleared by software or by reading the ADC_DR register.

0: Conversion not complete (EOCS=0), or sequence of conversions not complete (EOCS=1)

1: Conversion complete (EOCS=0), or sequence of conversions complete (EOCS=1)

Bit 0 **AWD:** Analog watchdog flag

This bit is set by hardware when the converted voltage crosses the values programmed in the ADC_LTR and ADC_HTR registers. It is cleared by software.

0: No analog watchdog event occurred

1: Analog watchdog event occurred

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

Bits 31:27 Reserved, must be kept at reset value.

Bit 26 **OVRIE:** Overrun interrupt enable

This bit is set and cleared by software to enable/disable the Overrun interrupt.
0: Overrun interrupt disabled
1: Overrun interrupt enabled. An interrupt is generated when the OVR bit is set.

Bits 25:24 **RES[1:0]:** Resolution

These bits are written by software to select the resolution of the conversion.
00: 12-bit (15 ADCCLK cycles)
01: 10-bit (13 ADCCLK cycles)
10: 8-bit (11 ADCCLK cycles)
11: 6-bit (9 ADCCLK cycles)

Bit 23 **AWDEN:** Analog watchdog enable on regular channels

This bit is set and cleared by software.
0: Analog watchdog disabled on regular channels
1: Analog watchdog enabled on regular channels

Bit 22 **JAWDEN:** Analog watchdog enable on injected channels

This bit is set and cleared by software.
0: Analog watchdog disabled on injected channels
1: Analog watchdog enabled on injected channels

Bits 21:16 Reserved, must be kept at reset value.

Bits 15:13 **DISCNUM[2:0]:** Discontinuous mode channel count

These bits are written by software to define the number of regular channels to be converted in discontinuous mode, after receiving an external trigger.
000: 1 channel
001: 2 channels
...
111: 8 channels

Bit 12 **JDISCEN:** Discontinuous mode on injected channels

This bit is set and cleared by software to enable/disable discontinuous mode on the injected channels of a group.
0: Discontinuous mode on injected channels disabled
1: Discontinuous mode on injected channels enabled

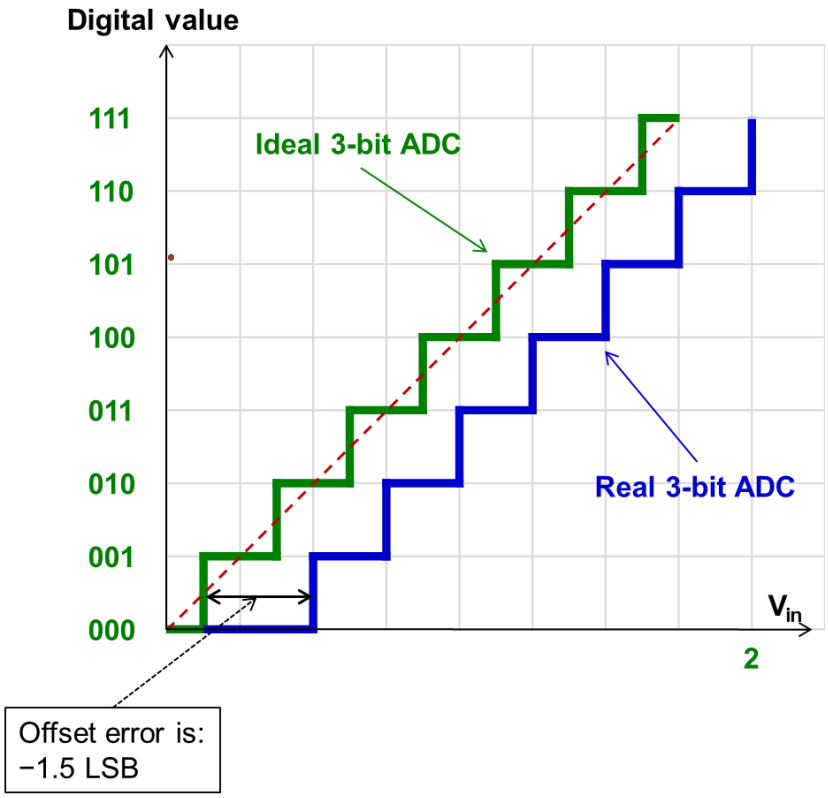
ADC region		offset		Address of register
0x4001 2000 - 0x4001 23FF	ADC1	0x000 - 0x04C	Specific registers	0x40012000 + 0x000 + register offset
			Reserved	
	ADC2	0x100 - 0x14C	Specific registers	0x40012000 + 0x100 + register offset
			Reserved	
	ADC3	0x200 - 0x24C	Specific registers	0x40012000 + 0x200 + register offset
			Reserved	
	Common	0x300 - 0x308	Common registers	0x40012000 + 0x300 + register offset

Solution

1. Offset errors

Solution 1.a (as below, but only the green curve)

Solution for 1.b (only the blue curve)



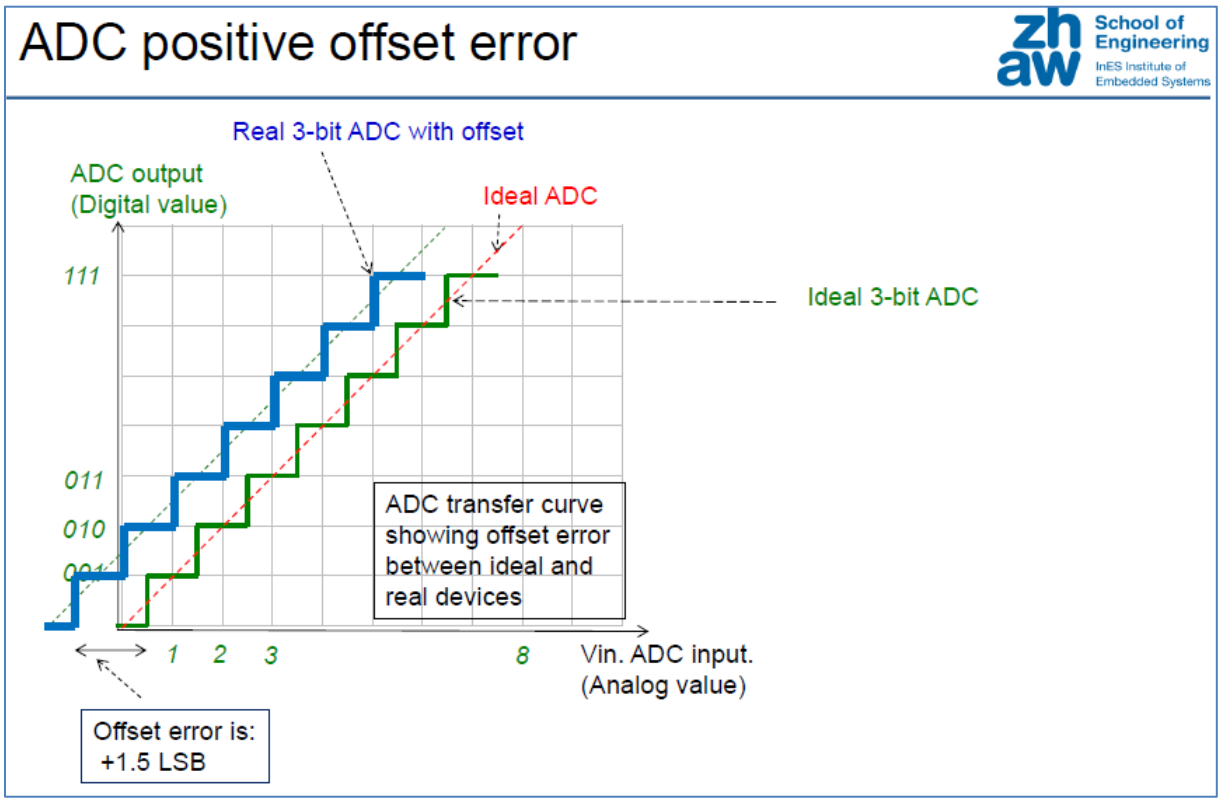
V_{REF} = 2 V → 1LSB = $2/2^3$ = 2/8 = 0.25V

Offset error of -1.5 LSB corresponds to -1.5 * 0.25V = 0.375V

FSR = V_{REF} – 1LSB = 2 - 0.25V = 1.75V

Offset Error (%FSR) = Offset Error (V) * 100 / FSR = 0.375V * 100 / 1.75V = 21.43%

Additional information:
In case of a positive offset error of +1.5 LSB, the graph will look like below.



2. Programming the ADC

```
a.
// Check Bit1 (EOC) of ADC1 status register.
// Address of status register of ADC1 register is:
//0x4001'2000 + base offset of ADC1 + status reg offset = 0x40012000

#define MY_STATUS_REG_ADC1 (*((volatile uint8_t *)(0x40012000)))
while (!(MY_STATUS_REG_ADC1 & 0x02)){
}

b.
// If bit5 of status register is 1, then an overrun has occurred.
// Address of status register of ADC1 register is:
//0x4001'2000 + base offset of ADC2 + status reg offset = 0x40012100

#define MY_STATUS_REG_ADC2 (*((volatile uint8_t *)(0x40012100)))

if (MY_STATUS_REG_ADC2 & 0x20){
    var2 = 0xFF;
} else {
    var2 = 0x00;
}

c.
// Write [bit25 bit24] of ADC3 control register1 to [01]
// Address of that register is:
//0x4001'2000 + base offset ADC3 + Control reg1 offset = 0x40012204
```

```
#define MY_control_REG1_ADC3 (*((volatile uint32_t *)(0x40012204)))

MY_control_REG1_ADC3 |= 0x01000000; // force bit24 to 1
MY_control_REG1_ADC3 &= 0xFDFFFFFF; // force bit25 to 0
```

Could also be written as:

```
MY_control_REG1_ADC3 &= 0xFCFFFFFF; // force [bit25 bit24] to [00]
MY_control_REG1_ADC3 |= 0x01000000; // force [bit25 bit24] to [01]
```