ADC Polled A/D Conversion

The ADC0 (ADC zero) module on the KL05 is a 12-bit analog-to-digital (A/D)converter. It converts an analog voltage to a *n*-bit digital number, where *n* can be selected from 8 to 12 depending on the specific configuration of the module. The analog voltage can be a single input voltage relative to the module's low reference voltage, (i.e., single-ended). The digital number is determined based on the relationship of the input voltage in reference to the range of the module's high and low reference voltages.

Using the ADC in a program for polled single-ended A/D conversion of an input relative to VDDA, (3.3 V), requires first selecting the ADC module clock source and frequency, power configuration, output channel, speed and sample time configuration, and reference voltage. Next, the module's self-calibration function should be started. After the self-calibration finishes, the gain compensation values must be computed. Finally, the module can be configured for to perform a single conversion on trigger. Afterward, conversion must be initiated by writing to the status and control register for the desired channel, and the digital value may be obtained through polling.

Hardware configuration

In lab, the ADC0 is used to convert the analog voltage output DAC0_OUT from DAC0 to a 10-bit digital value. On the KL05, DAC0_OUT connects to ADC0_SE5 on the ADC0 module. No additional external wiring is required to use the ADC0 for this configuration.

Program initialization

The following initializations are required prior to ADC0 self-calibration.

- ADC0 module clock enabled in SIM_SCGC6: ADC0 = 1
- ADC0 low-power mode enabled, input clock divide by four selected for ADCK, long sample time selected, 10-bit conversion selected, and bus clock divided by two selected for input clock in ADC0_CFG1, (see analysis that follows):
  ADLPC = 1; ADIV = 10_2; ADLSMP = 1; MODE = 10_2; ADICLK = 01_2
- ADC0 output channel A selected, asynchronous clock output disabled, normal conversion speed selected and default longest sample time selected in ADC0_CFG2:
  MUXSEL = 0; ADACKEN = 0; ADHSC = 0; ADLSTS = 00_2.
- ADC0 software trigger selected, compare function disabled, DMA disabled,and VDDA as reference voltage in ADC0_SC2, (see analysis that follows):
  ADTRG = 0; ACFE = 0; DMAEN = 0; REFSEL = 01_2.

Next, the self-calibration function is initiated by setting the CAL bit, selecting one set of conversions, enabling hardware averaging, and selecting the maximum number of samples in ADC0_SC3.

CAL = 1; ADCO = 0; AVGE = 1; AVGS = 11_2

When the self-calibration function finishes, the CAL bit is cleared, so CAL must be polled for being clear before proceeding. In addition, any error in calibration will cause the CALF bit to be set, so after self-calibration completes, (i.e., CAL = 0), it should next be verified that CALF = 0.

After self-calibration has successfully completed, (i.e., CAL = 0 and CALF = 0), the gain compensation value must be computed using 16-bit values and operations, and then must be stored for automatic gain compensation by the ADC, as indicated by the pseudocode below, where the | operator indicates bitwise or.

```
ADC0_PG = ((ADC0_CLP0 + ADC0_CLP1 + ADC0_CLP2
                ADC0_CLP3 + ADC0_CLP4 + ADC0_CLPS) >> 1) | 0x8000
```

The final initialization step after self-calibration has completed is to select normal polled A/D conversion in ADC0_SC3.

CAL = 0; ADCO = 0; AVGE = 0

Analysis. For the self-calibration function, the frequency of ADCK needs to be 4 MHz or less. With a KL05 core clock frequency of 48 MHz, the bus clock has a frequency of 24 MHz, (i.e., half of the core clock). If bus clock divided by two is selected as the ADC0 module input clock, then the ADC0 input clock has a frequency of 12 MHz. The ADIV field of ADC0_CFG1 determines the input clock divisor (2^{ADIV}) for ADCK. Thus, the value of ADIV for the fastest ADCK frequency usable for self-calibration is determined as follows.

$$ADIV = \left\lceil \log_2 \frac{f_{ADCInputClock}}{4MHz} \right\rceil = \left\lceil \log_2 \frac{12MHz}{4MHz} \right\rceil = \lceil 1.6 \rceil = 2$$

The REFSEL field of ADC0_SC2 selects the reference voltages. The KL05 configuration provides two possible reference pairs: VREFH/VREFL and VDDA/VSSA. REFSEL = 1 selects VDDA and VSSA, (i.e., +3.3 V and GND), as the reference voltage pair.

Assembly language code. The following assembly language code initializes and calibrates ADC0 for polled A/D conversion of DAC0_OUT. All symbols not defined in this code are from the RIT CMPE-250 include file MKL05Z4.s.

```
;ADC0_CFG1 symbols
ADC0_CFG1_ADIV_BY2              EQU  2
ADC0_CFG1_MODE_SGL10            EQU  2
ADC0_CFG1_ADICLK_BUSCLK_DIV2    EQU  1
ADC0_CFG1_LP_LONG_SGL10_3MHZ    EQU  (ADC_CFG1_ADLPC_MASK :OR: \
             (ADC0_CFG1_ADIV_BY2 << ADC_CFG1_ADIV_SHIFT) :OR: \
             ADC_CFG1_ADLSMP_MASK :OR: \
             (ADC0_CFG1_MODE_SGL10 << ADC_CFG1_MODE_SHIFT) :OR: \
             ADC0_CFG1_ADICLK_BUSCLK_DIV2)
;ADC0_CFG2 symbol
ADC0_CFG2_CHAN_A_NORMAL_LONG    EQU  0x00
;ADC0_SC2 symbols
ADC0_SC2_REFSEL_VDDA            EQU  0x01u
ADC0_SC2_SWTRIG_VDDA            EQU  ADC0_SC2_REFSEL_VDDA
;ADC0_SC3 symbols
ADC0_SC3_CAL                    EQU  (ADC_SC3_CAL_MASK :OR: \
                                      ADC_SC3_AVGE_MASK :OR: \
                                      ADC_SC3_AVGS_MASK)
ADC0_SC3_SINGLE                 EQU  0x00

;Enable ADC0 module clock
            LDR   Ri,=SIM_SCGC6
            LDR   Rj,=SIM_SCGC6_ADC0_MASK
            LDR   Rk,[Ri,#0]        ;current SIM_SCGC6 value
            ORRS  Rk,Rk,Rj          ;only ADC0 bit changed (set)
            STR   Rk,[Ri,#0]        ;update SIM_SCGC6
;Set ADC0 power and timing
            LDR   Ri,=ADC0_BASE
            MOVS  Rj,#ADC0_CFG1_LP_LONG_SGL10_3MHZ
            STR   Rj,[Ri,#ADC_CFG1_OFFSET]
;Select channel A and set timing
            MOVS  Rj,#ADC0_CFG2_CHAN_A_NORMAL_LONG
            STR   Rj,[Ri,#ADC_CFG2_OFFSET]
;Select SW trigger and VDDA reference
            MOVS  Rj,#ADC0_SC2_SWTRIG_VDDA
            STR   Rj,[Ri,#ADC_SC2_OFFSET]
```

```
;Start calibration
START_CAL
            MOVS   Rj,#ADC0_SC3_CAL
            STR    Rj,[Ri,#ADC_SC3_OFFSET]
;Wait for calibration to complete
            MOVS   Rj,#ADC_SC3_CAL_MASK
WAIT_CAL    LDR    Rk,[Ri,#ADC_SC3_OFFSET]
            TST    Rk,Rj
            BNE    WAIT_CAL
;Check for calibration failure
            MOVS   Rj,#ADC_SC3_CALF_MASK
            LDR    Rk,[Ri,#ADC_SC3_OFFSET]
            TST    Rk,Rj
            BNE    START_CAL
;Compute and store plus-side calibration value using all 16-bit operations
            LDR    Rj,=0xFFFF
            LDR    Rk,=0x8000
            LDR    Rm,[Ri,#ADC_CLP0_OFFSET]
            LDR    Rn,[Ri,#ADC_CLP1_OFFSET]
            ADDS   Rm,Rm,Rn
            ANDS   Rm,Rm,Rj
            LDR    Rn,[Ri,#ADC_CLP2_OFFSET]
            ADDS   Rm,Rm,Rn
            ANDS   Rm,Rm,Rj
            LDR    Rn,[Ri,#ADC_CLP3_OFFSET]
            ADDS   Rm,Rm,Rn
            ANDS   Rm,Rm,Rj
            LDR    Rn,[Ri,#ADC_CLP4_OFFSET]
            ADDS   Rm,Rm,Rn
            ANDS   Rm,Rm,Rj
            LDR    Rn,[Ri,#ADC_CLPS_OFFSET]
            ADDS   Rm,Rm,Rn
            ANDS   Rm,Rm,Rj
            LSRS   Rm,Rm,#1
            ORRS   Rm,Rm,Rk
            STR    Rm,[Ri,#ADC_PG_OFFSET]
;Select single conversion
            MOVS   Rj,#ADC0_SC3_SINGLE
            STR    Rj,[Ri,#ADC_SC1A_OFFSET]
```

C code. The following C code initializes and calibrates ADC0 for polled A/D conversion of DAC0_OUT. All symbols not defined in this code are from the Keil Freescale include file MKL05Z4.h.

```
/* ADC0_CFG1 symbols */
#define ADC0_CFG1_ADIV_BY2 (2u)
#define ADC0_CFG1_MODE_SGL10 (2u)
#define ADC0_CFG1_ADICLK_BUSCLK_DIV2 (1u)
#define ADC0_CFG1_LP_LONG_SGL10_3MHZ \
               (ADC_CFG1_ADLPC_MASK | \
                (ADC0_CFG1_ADIV_BY2 << ADC_CFG1_ADIV_SHIFT) | \
                ADC_CFG1_ADLSMP_MASK | \
                (ADC0_CFG1_MODE_SGL10 << ADC_CFG1_MODE_SHIFT) | \
                ADC0_CFG1_ADICLK_BUSCLK_DIV2)
/* ADC0_CFG2 symbol */
#define ADC0_CFG2_CHAN_A_NORMAL_LONG (0x00u)
/* ADC0_SC2 symbols */
#define ADC0_SC2_REFSEL_VDDA (0x01u)
#define ADC0_SC2_SWTRIG_VDDA (ADC0_SC2_REFSEL_VDDA)
/* ADC0_SC3 symbols */
#define ADC0_SC3_CAL (ADC_SC3_CAL_MASK | \
                      ADC_SC3_AVGE_MASK | \
                      ADC_SC3_AVGS_MASK)
#define ADC0_SC3_SINGLE (0x00u)

/* Enable ADC0 module clock */
SIM->SCGC6 |= SIM_SCGC6_ADC0_MASK;
/* Set ADC0 power and timing */
ADC0->CFG1 = ADC0_CFG1_LP_LONG_SGL10_3MHZ;
/* Select channel A and set timing */
ADC0->CFG2 = ADC0_CFG2_CHAN_A_NORMAL_LONG;
/* Select SW trigger and VDDA reference */
ADC0->SC2 = ADC0_SC2_SWTRIG_VDDA;
/* Start calibration */
do {
     ADC0->SC3 = ADC0_SC3_CAL;
     /* Wait for calibration to complete */
     while (ADC0->SC3 & ADC_SC3_CAL_MASK);
     /* Check for calibration failure */
} while (ADC0->SC3 & ADC_SC3_CALF_MASK)
/* Compute and store plus-side calibration value */
ADC0->PG = (((Int16) ADC0->CLPS + (Int16) ADC0->CLP4 +
             (Int16) ADC0->CLP3 + (Int16) ADC0->CLP2 +
             (Int16) ADC0->CLP1 + (Int16) ADC0->CLP0
            ) >> 1) | (Int16) 0x8000;
/* Select single conversion */
ADC0->SC3 = ADC0_SC3_SINGLE;
```

ADC polled A/D conversions

Once a program has initialized the ADC, the ADC may be used for polled A/D conversions via two steps. First, the conversion must be initiated by writing to ADC0_SC1A to specify polled single-ended or differential mode conversion and to specify the input channel. For lab, polled single-ended conversion of input channel AD5, (which connects to DAC0_OUT), is specified.

AIEN = 0; ADCH = 00101_2

After the A/D conversion from AD5 (DAC0_OUT) is begun by writing to ADC0_SC1A, the COCO bit of ADC0_SC1A is polled for the bit to be set, (COCO = 1). When COCO is set, the converted digital value from AD5 may be read from ADC0_RA

Assembly language code. The following assembly language code performs a polled input of the A/D conversion of DAC0_OUT. All symbols not defined in this code are from the RIT CMPE-250 include file MKL05Z4.s.

```
;ADC0_SC1A symbols
ADC0_SC1_ADCH_AD5               EQU   5
ADC0_SC1_SGL_DAC0               EQU   ADC0_SC1_ADCH_AD5

;Start conversion
            LDR   Ri,=ADC0_BASE
            MOVS  Rj,#ADC0_SC1_SGL_DAC0
            STR   Rj,[Ri,#ADC0_SC1A_OFFSET]
;Wait for conversion to complete
            MOVS  Rj,#ADC_SC1_COCO_MASK
Wait        LDR   Rk,[Ri,#ADC0_SC1A_OFFSET]
            TST   Rk,Rj
            BEQ   Wait
;Read digital value
            LDR   Rj,[Ri,#ADC_RA_OFFSET]
```

C code. The following C code performs a polled input of the A/D conversion of DAC0_OUT. All symbols not defined in this code are from the Keil Freescale include file MKL05Z4.h.

```
/* ADC0_SC1A symbols */
#define ADC0_SC1_ADCH_AD5 (5u)
#define ADC0_SC1_SGL_DAC0 (ADC0_SC1_ADCH_AD5)

;Start conversion
ADC0->SC1[0] = ADC0_SC1_SGL_DAC0;
;Wait for conversion to complete
while (!(ADC0->SC1[0] & ADC_SC1_COCO_MASK));
;Read digital value
UInt16Variable = ADC0->R[0];
```