EDDI
Electronic Design
Development Institute

에디로봇아카데미

임베디드 마스터 Lv2 과정

제 1기

2022. 02. 27

손표훈

CONTENTS

EDDI
Electronic Design
Development Institute

- TMS570LC43x의 ADC 특징
  - ADC의 conversion period 계산
  - ADC의 digital value 계산식
- ADC의 H/W 구성
- ADC의 주요 레지스터 설정
- Halcogen ADC 설정
- ADC get data 함수

TMS570LC43x의 ADC 특징

1. SAR타입의 ADC이다.

2. 10 ~ 12bit 분해능을 선택 할 수 있다.

3. 3개의 변환 그룹으로 나뉜다
(1) Event group : H/W trigger만 가능
(2) group1 : H/W & S/W trigger 둘 다 가능
(2) group2 : H/W & S/W trigger 둘 다 가능

4. ADC peripheral은 2개의 SAR타입의 ADC로 구성된다.
- 여기서 channel은 ADC 핀 개수이다-
(1) ADC1 : 32 channel
(2) ADC2 : 25 channel(여기서 16 channel은 ADC1과 공유된다)

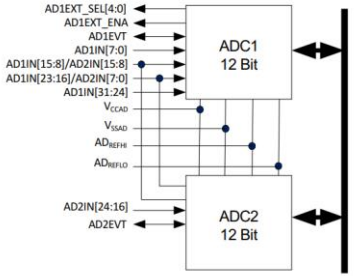
5. ADC 변환모드는 두개가 있다
(1) single conversion
(2) continuos conversion

6. H/W trigger는 8개가 있다

This microcontrollers implements up to two instances of the ADC module. The main features of the ADC module are:

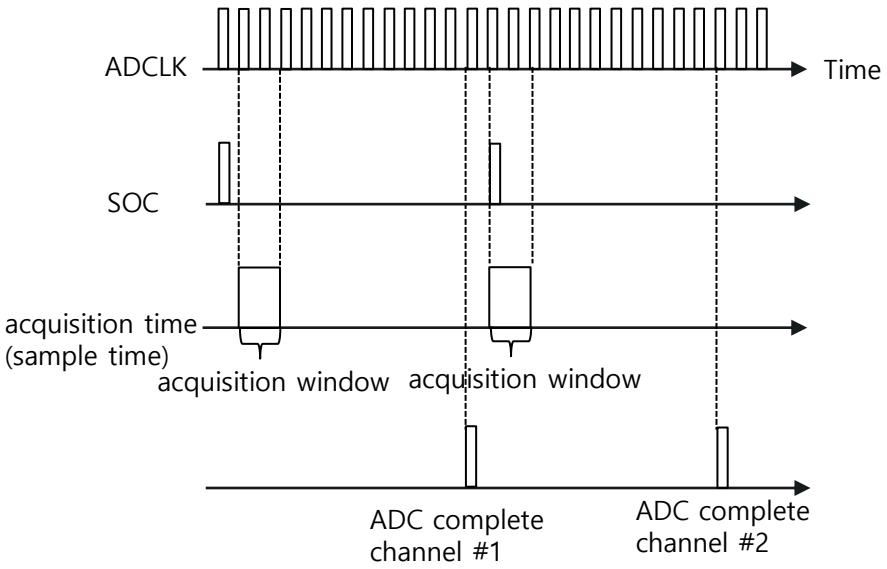
- Selectable 10-bit or 12-bit resolution
- Successive-approximation-register architecture
- Three conversion groups – Group1, Group2, and Event Group
- All three conversion groups can be configured to be hardware-triggered; group1 and group2 can also be triggered by software
- Conversion results are stored in a 64-word memory (SRAM)
  - These 64 words are divided between the three conversion groups and are configurable by software
  - Accesses to the conversion result RAM are protected by parity
- Flexible options for generating DMA requests for transferring conversion results
- Selectable channel conversion order
  - Sequential conversions in ascending order of channel number, OR
  - User-defined channel conversion order with the Enhanced Channel Selection Mode
    - The Enhanced Channel Selection Mode is only available to ADC1.
- Single or continuous conversion modes
- Embedded self-test logic for input channel failure detection (open / short to power / short to ground)
- Embedded calibration logic for offset error correction
- Enhanced Power-down mode
- External event pin (ADEVT) to trigger conversions
  - ADEVT is also programmable as general-purpose I/O
- Eight hardware events to trigger conversions

Figure 22-1. Channel Assignments of Two ADC Cores



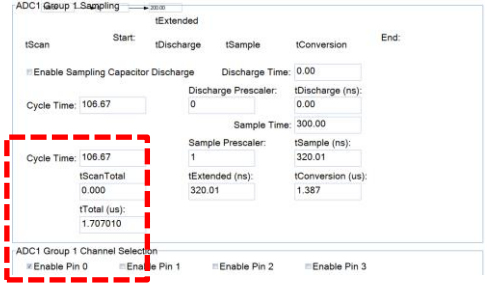
TMS570LC43x의 ADC 특징

➢ ADC의 conversion period 계산



- Conversion Period:
  - The conversion period starts one full ADCLK after the falling edge of START.
  - One bit of the conversion result is output on each rising edge of ADCLK in the conversion period, starting with the most-significant bit first.
  - The conversion period is 12 ADCLK cycles in case of a 12-bit ADC, and is 10 ADCLK cycles in case of a 10-bit ADC.
  - The ADC core generates an End-Of-Conversion (EOC) signal to the sequencer at the end of the conversion period. At this time the complete 12-, or 10-bit conversion result is available.
  - The sequencer captures the ADC core conversion result output as soon as EOC is driven High.

→ 12bit 모드는 ADCLK을 12cycle
→ 10bit 모드는 ADCLK을 10cycle 가져 간다



* Halcogen 1ch ADC 변환 총 시간

→ ADCLK이 VCLK3를 8분주하여 106.67ns라 하고, acquisition time이 ADxSAMP 설정으로 3 ADCLK을 가져간다면
→ 12bit 모드라면 변환 결과를 얻는데 총 15clock이 소모된다 : 채널당 대략 1.6us가 소요된다

TMS570LC43x의 ADC 특징

EDDI
Electronic Design Development Institute

➢ ADC의 digital value 계산식

$$DigitalResult = \frac{1024 \times (InputVoltage - AD_{REFLO})}{AD_{REFHI} - AD_{REFLO}} - 0.5$$

10bit 분해능의 디지털 변환식

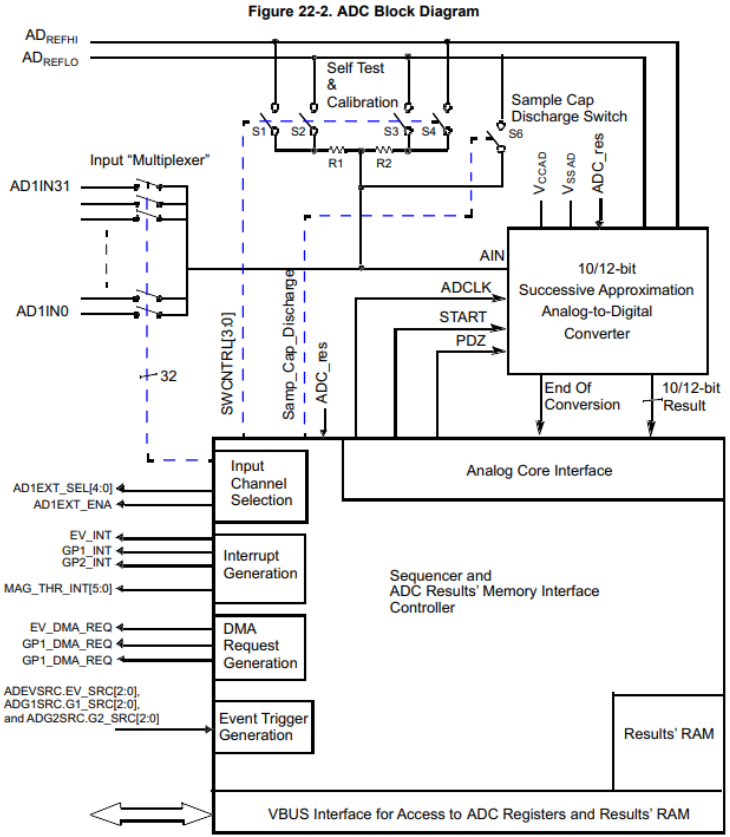
$$DigitalResult = \frac{4096 \times (InputVoltage - AD_{REFLO})}{(AD_{REFHI} - AD_{REFLO})} - 0.5$$

12bit 분해능의 디지털 변환식

→ ADREFHI는 ADREFHI핀에 입력되는 ADC의 +기준전압

→ ADREFLO는 ADREFLO핀에 입력되는 ADC의 -기준전압

ADC의 H/W 구성



1. ADC input channel을 선택할 수 있는 MUX
2. 인터럽트 생성 기능
3. DMA(Direct Memory Access)모듈의 요청 수신 기능
4. SOC(Start Of Conversion) 신호를 선택 할 수 있는 기능
5. SAR타입의 ADC와 SAR타입의 Logic Control 모듈과 메모리 인터페이스+64word크기의 RAM
6. Sample & Hold의 Cap 방전시간을 설정할 수 있는 기능

총 6개의 sub 모듈로 구성되어 있다

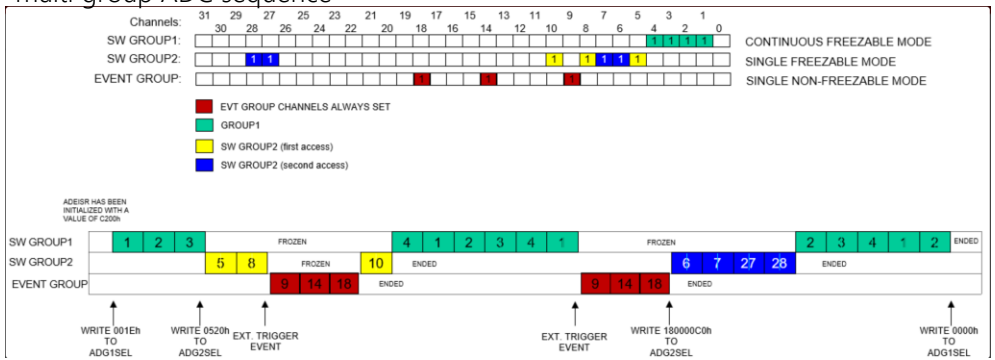
ADC의 주요 레지스터 설정

EDDI
Electronic Design Development Institute

→ 우선 TMS570LC43x의 ADC의 기본 동작을 위해 어떤 레지스터 설정이 필요한지 다음을 살펴보자

1. 분해능 선택
   → TMS570LC43x의 ADC에는 10bit, 12bit의 분해능을 설정할 수 있다
2. ADC 클럭을 설정한다
   → ADC의 Sampling 주파수를 설정하는 부분이다
   → GCM모듈로 부터 분주된 VCLK3의 주파수를 입력 받아 분주비 설정을 통해 주파수를 설정한다
3. ADC의 acquisition time을 설정한다
   → acquisition time은 sample을 얻는 시간을 의미하며, sample & hold시간을 의미한다
   → 디지털 결과까지 얻는 시간을 의미하는 것은 아니다
4. 변환 모드를 설정한다
   → ADC는 2가지 변환모드가 있다
   (1) single conversion : SOC trigger 한번에 한번의 ADC가 실행되고 다시 trigger 신호가 발생해야 ADC를 시작한다
   이 때 다중 그룹의 ADC를 사용하는 경우 그룹 우선순위에 따라 순서대로 ADC 실행
   (2) continuous conversion : SOC trigger가 한번만 발생하면 trigger의 재발생 여부와 상관없이 계속 ADC를 실행한다
   이 때 group의 우선순위에 따라 자동으로 ADC가 순서대로 실행된다

* multi group ADC sequence

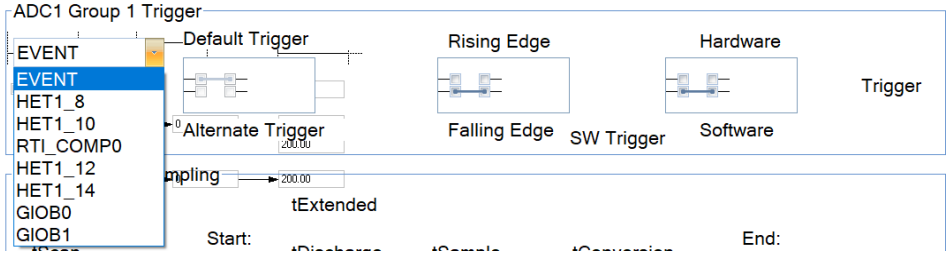


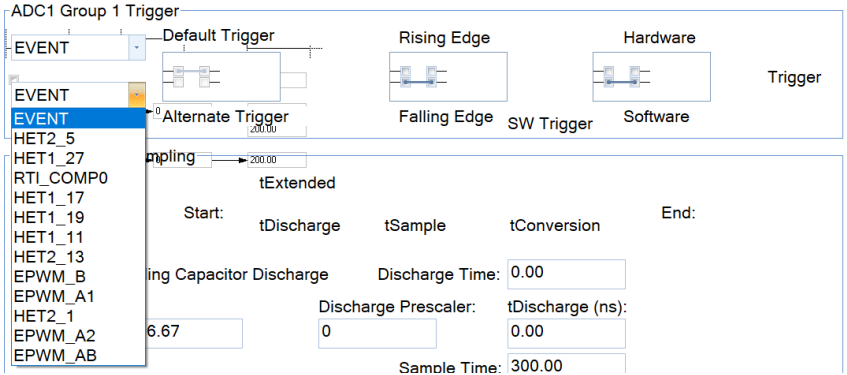
출처 : https://e2e.ti.com/support/microcontrollers/arm-based-microcontrollers-group/arm-based-microcontrollers/f/arm-based-microcontrollers-forum/1029658/tms570ls3137-adc-continuous-conversion-mode/3808084?tisearch=e2e-sitesearch&keymatch=single%2525252520conversion#3808084

ADC의 주요 레지스터 설정

→ 우선 TMS570LC43x의 ADC의 기본 동작을 위해 어떤 레지스터 설정이 필요한지 다음을 살펴보자

5. SOC 신호를 설정한다
   → Trigger source의 종류는 아래와 같다





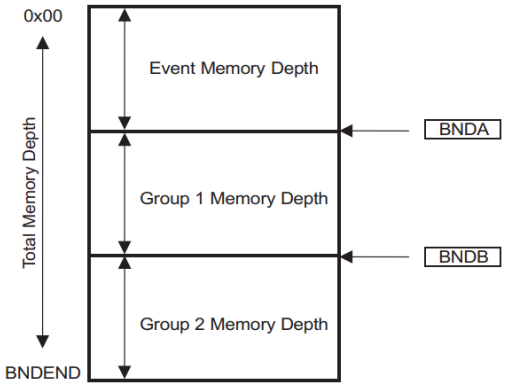
→ trigger source의 극성을 선택한다 : rising edge or falling edge

ADC의 주요 레지스터 설정

→ 우선 TMS570LC43x의 ADC의 기본 동작을 위해 어떤 레지스터 설정이 필요한지 다음을 살펴보자

6. 변환 완료가 되었는지 확인한다
   → ADEVSR, ADG1SR, ADG2SR status register를 통해 변환 완료를 확인한다
7. 변환 결과를 ADC모듈내의 SRAM으로 부터 읽는다
   → ADBNDCR의 BNDA, BNDB bit field를 설정하여 그룹별 메모리 주소의 boundary를 설정하여 사이즈를 설정한다
   → ADBNDEND를 이용하여 64word의 사이즈 중 얼만큼 사용할 지를 결정한다

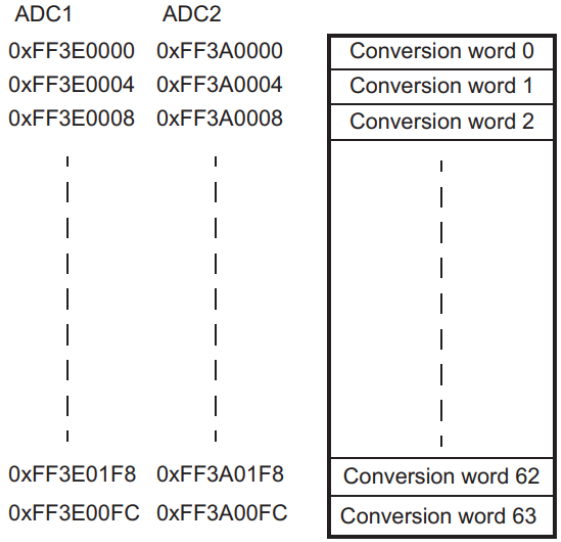
Figure 22-3. FIFO Implementation



- Number of buffers for Event Group = 2 × BNDA
- Number of buffers for Group1 = 2 × (BNDB – BNDA)
- Number of buffers for Group2 = Total number of buffers – 2 × BNDB

→ channel의 순번의 오름차순에 따라 메모리에 변환 결과가 저장된다

Figure 22-6. ADC Memory Mapping



ADC의 주요 레지스터 설정

EDDI
Electronic Design Development Institute

→ FIFO내 DATA와 Channel ID의 영역은 아래와 같다
→ 0~11(12bit)까지 변환결과가 위치하고 16~20(5bit)까지 Channel ID가 위치한다

Figure 22-4. Format of Conversion Result Read from FIFO, 12-bit ADC

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x90 to 0xAF ADEVBUFFER	EV_EMPTY	Reserved										EV_CHID				
	Reserved											EV_DR				
0xB0 to 0xCF ADG1BUFFER	G1_EMPTY	Reserved										G1_CHID				
	Reserved											G1_DR				
0xD0 to 0xEF ADG2BUFFER	G2_EMPTY	Reserved										G2_CHID				
	Reserved											G2_DR				

Figure 22-5. Format of Conversion Result Read from FIFO, 10-bit ADC

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x90 to 0xAF ADEVBUFFER	Reserved															
	EV_EMPTY	EV_CHID										EV_DR				
0xB0 to 0xCF ADG1BUFFER	Reserved															
	G1_EMPTY	G1_CHID										G1_DR				
0xD0 to 0xEF ADG2BUFFER	Reserved															
	G2_EMPTY	G2_CHID										G2_DR				

ADC의 주요 레지스터 설정

EDDI
Electronic Design Development Institute

→ ADC의 기본동작 설정 절차

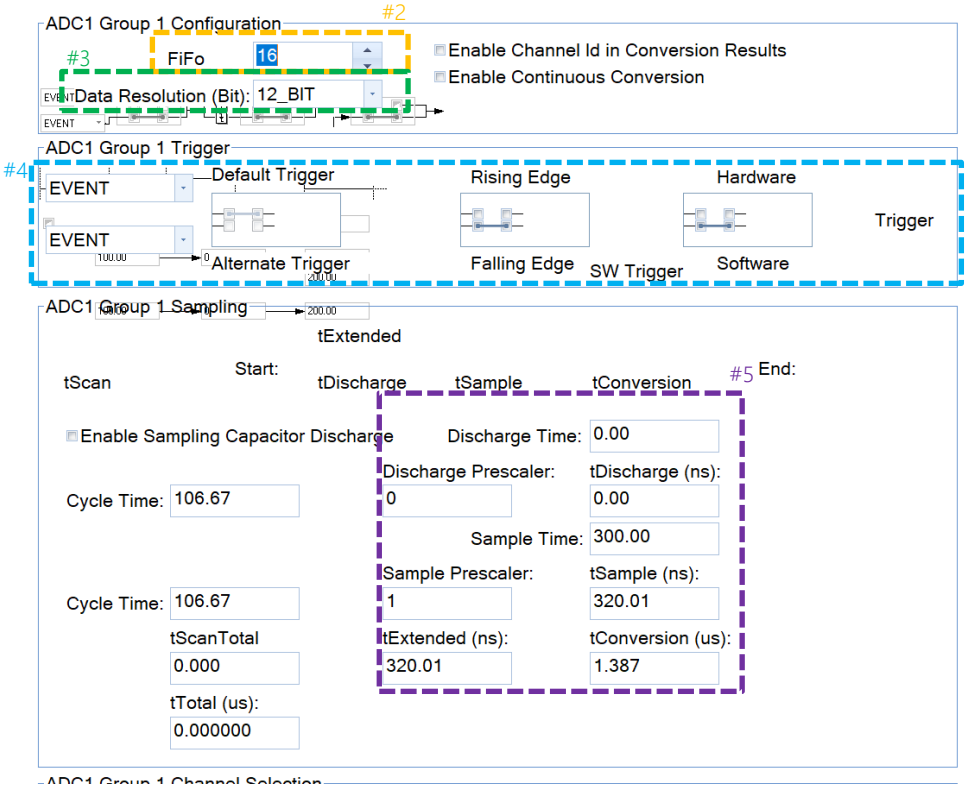
22.2.1.11 Example Sequence for Basic Configuration of ADC Module

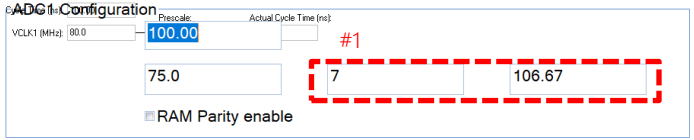
The following sequence is necessary to configure the ADC to convert channels 0, 2, 4, and 8 in single-conversion mode using Group1:

1. Write 0 to the Reset Control Register (ADRSTCR) to release the module from the reset state
2. Write 1 to the ADC_EN bit of the Operating Mode Control Register (ADOPMODECR) to enable the ADC state machine
3. Configure the ADCLK frequency by programming the desired divider into the Clock Control Register (ADCLOCKCR)
4. Configure the acquisition time for the group that is to be used. For example, configure the Group1 Sampling Time Control Register (ADG1SAMP) to set the acquisition time for Group1.
5. Select the channels that need to be converted in Group1 by writing to the Group1 Channel Select Register (ADG1SEL). In this example, a value of 0x115 needs to be written to ADG1SEL in order to select channels 0, 2, 4, and 8 for conversion in Group1.
   - The ADC sequencer will start the Group1 conversions as soon as the write to the ADG1SEL register is completed.
6. Wait for the GP1_END bit to be set in the Group1 Conversion Status Register (ADG1SR). This bit gets set when all the channels selected for conversion in Group1 are converted and the results are stored in the Group1 memory.
7. Read the conversion results by reading from the Group1 FIFO access location (ADG1BUFFER) or by reading directly from the Group1 results' memory.

Halcogen ADC 설정

→ ADC의 기본동작 설정 절차





#1. VCLK3를 분주비를 통해 ADCLK으로 만든다
#2. ADC결과를 저장하는 FIFO의 크기를 설정한다
#3. ADC의 분해능을 설정한다
#4. ADC의 SOC trigger source와 극성, H/W & S/W trigger 방식을 설정한다
#5. ADC sample & hold capacitor의 방전주기를 설정한다

ADC get data 함수

EDDI
Electronic Design Development Institute

→ 실제 data를 얻는 부분에 집중해보면 buf변수를 통해 아래와 같은 adc의 buf레지스터(32bit)를 저장한다(12bit 분해능 기준)

Figure 22-4. Format of Conversion Result Read from FIFO, 12-bit ADC

Offset Address Register	31 15	30 14	29 13	28 12	27 11	26 10	25 9	24 8	23 7	22 6	21 5	20 4	19 3	18 2	17 1	16 0
0x90 to 0xAF ADEVBUFFER	EV_EMPTY	Reserved										EV_CHID				
	Reserved	EV_DR														
0xB0 to 0xCF ADG1BUFFER	G1_EMPTY	Reserved										G1_CHID				
	Reserved	G1_DR														
0xD0 to 0xEF ADG2BUFFER	G2_EMPTY	Reserved										G2_CHID				
	Reserved	G2_DR														

→ 사용자 입력 파라메터인 adcData_t 형 구조체 포인터를 같은 데이터형의 ptr에 저장

→ ptr을 통해 value값에 buf의 하위 12bit를 마스킹(&0xFFF)하여 ADC변환 결과를 얻는다

→ ptr을 통해 id값에 buf를 16bit 우 쉬프트 하고 하위5bit를 마스킹(&0x1F)하여 ADC의 channel ID 결과를 얻는다

```
adcData_t *ptr = data;

/* USER CODE BEGIN (16) */
/* USER CODE END */

    mode = (adc->OPMODECR & ADC_12_BIT_MODE);

    if(mode == ADC_12_BIT_MODE)
      {
        /** -  Get conversion data and channel/pin id */
        for (i = 0U; i < count; i++)
        {
          buf        = adc->GxBUF[group].BUF0;
                /*SAFETYMCUSW 45 D MR:21.1 <APPROVED> "Valid non NULL input parameters are only allowed in this driver" */
          ptr->value = (uint16)(buf & 0xFFFU);
          ptr->id    = (uint32)((buf >> 16U) & 0x1FU);
          /*SAFETYMCUSW 567 S MR:17.1,17.4 <APPROVED> "Pointer increment needed" */
                ptr++;
        }
      }
```

```
typedef struct adcData
{
    uint32 id;     /**< Channel/Pin Id        */
    uint16 value;  /**< Conversion data value */
} adcData_t;
```

→ 위 동작은 설정한 ADC 메모리의 group사이즈(예제코드는 16word)만큼 반복한다

→ 다중 채널 ADC의 경우 사용자 입력 파라메터인 data를 adcData_t구조체 배열 형식으로 선언하여 사용하여 채널별 ADC변환 결과를 얻을 수 있다