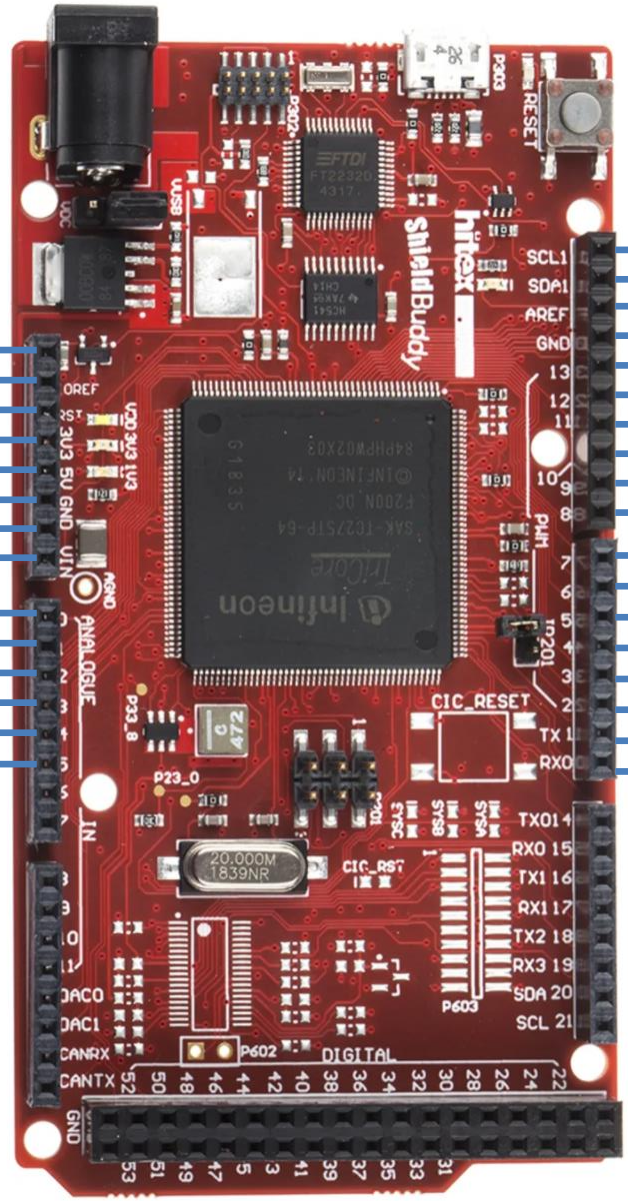
Infineon TC275 ADC (Analog-to-Digital Converter)

Architecture and Compiler for Embedded System LAB.
School of Electronics Engineering, KNU, KOREA
2021-05-11

Hitex ShieldBuddy TC275



ShieldBuddy TC275 Pin Layout	Arduino Uno Pin Layout
NC	NC
IOREF/VDD/5V	IOREF
RESET	Reset
+3.3V	+3V
VDD VIA RES521	+5V
GND	GND
GND	GND
VIN 9..12V	VIN
SAR4_7 AD0	A0|D14
SAR4_6 AD1	A1|D15
SAR4_5 AD2	A2|D16
SAR4_4 AD3	A3|D17
SAR3_1 AD4	A4|D18
SAR3_0 AD5	A5|D19
D19	P13_1 SCL1
D18	P13_2 SDA1
AREF	AREF0
GND	GND
D13	P10_2 SPCK
D12	P10_1 MISO
~D11	P10_3 MOSI
~D10	P10_5 SSO/PWM10
~D9	P02_7 PWM9
D8	P02_6 PWM8
D7	P02_5 PWM7
~D6	P02_4 PWM6
~D5	P02_3 PWM5
D4	P10_4 SS1/PWM4
~D3	P02_1 PWM3
D2	P02_0 PWM2
D1/TX	P15_2 TX
D0/RX	P15_3 RX

ACE Lab.

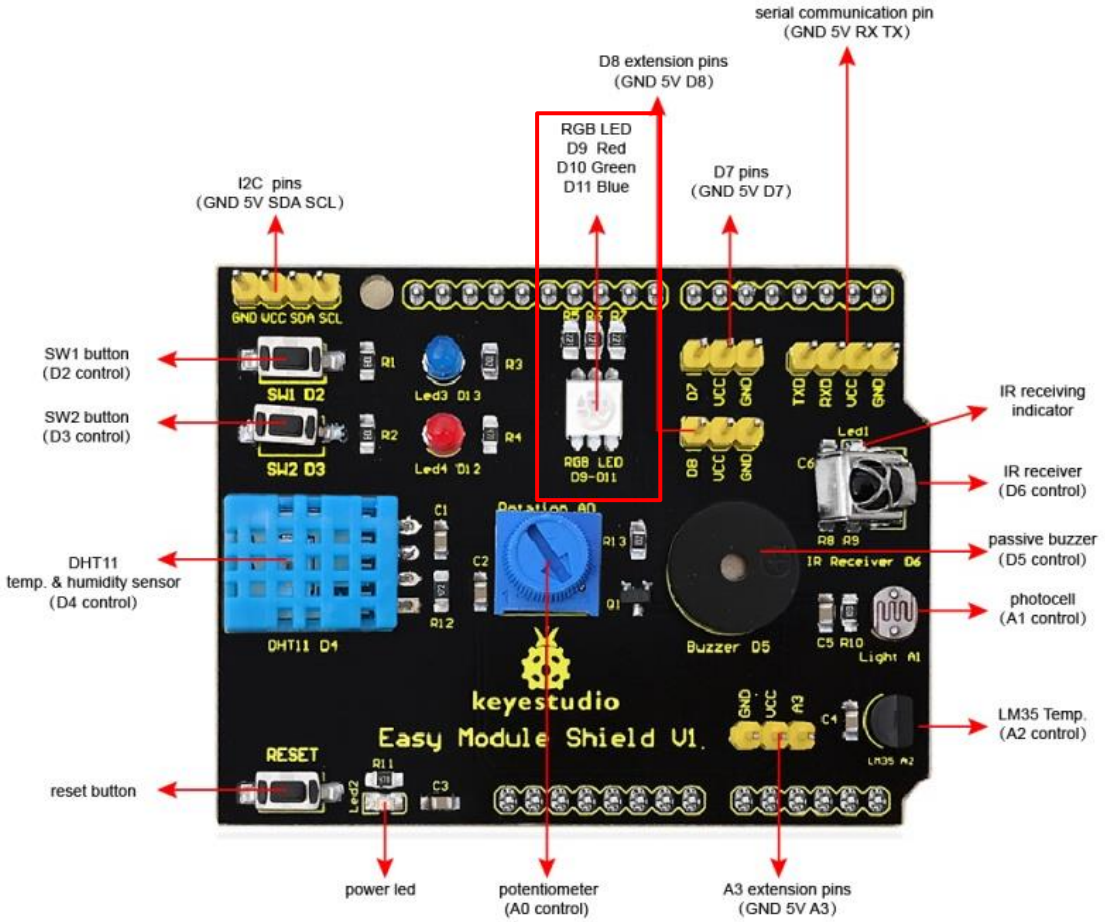
RGB LED Example

- GPIO를 통한 LED 제어
  1. 새로운 예제를 위한 프로젝트를 생성한다.
  2. 원하는 동작을 위해 레지스터와 메모리에 직접 접근해서 값을 써야한다.
  3. LED 사용을 위해 Board Schematic과 Datasheet에서 LED 연결 정보를 파악한다.
  4. LED가 연결된 PORT의 메모리 맵을 분석한다.
  5. 분석 결과를 활용해 임베디드 프로그래밍을 한다.

RGB LED Example

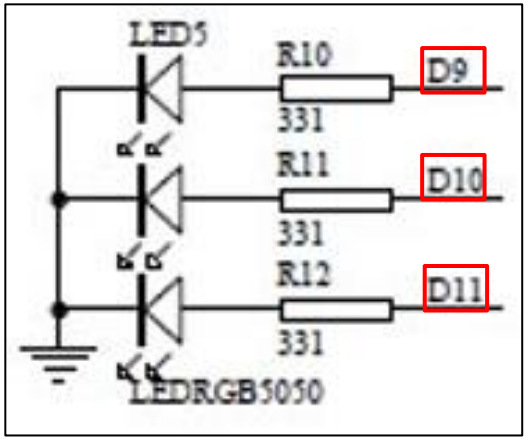
1. RGB LED 연결 정보 파악

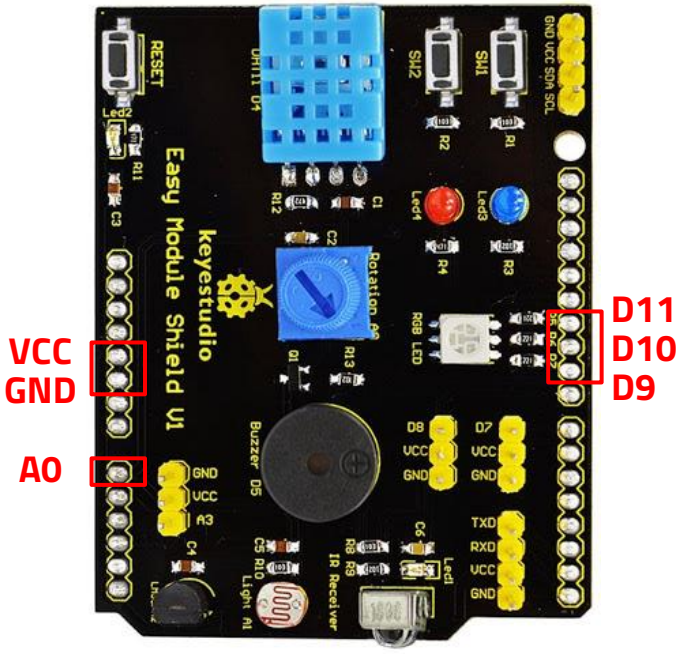
   ✓ **Easy Module Shield V1 확장 보드**의 RGB LED를 사용한다.



RGB LED Example

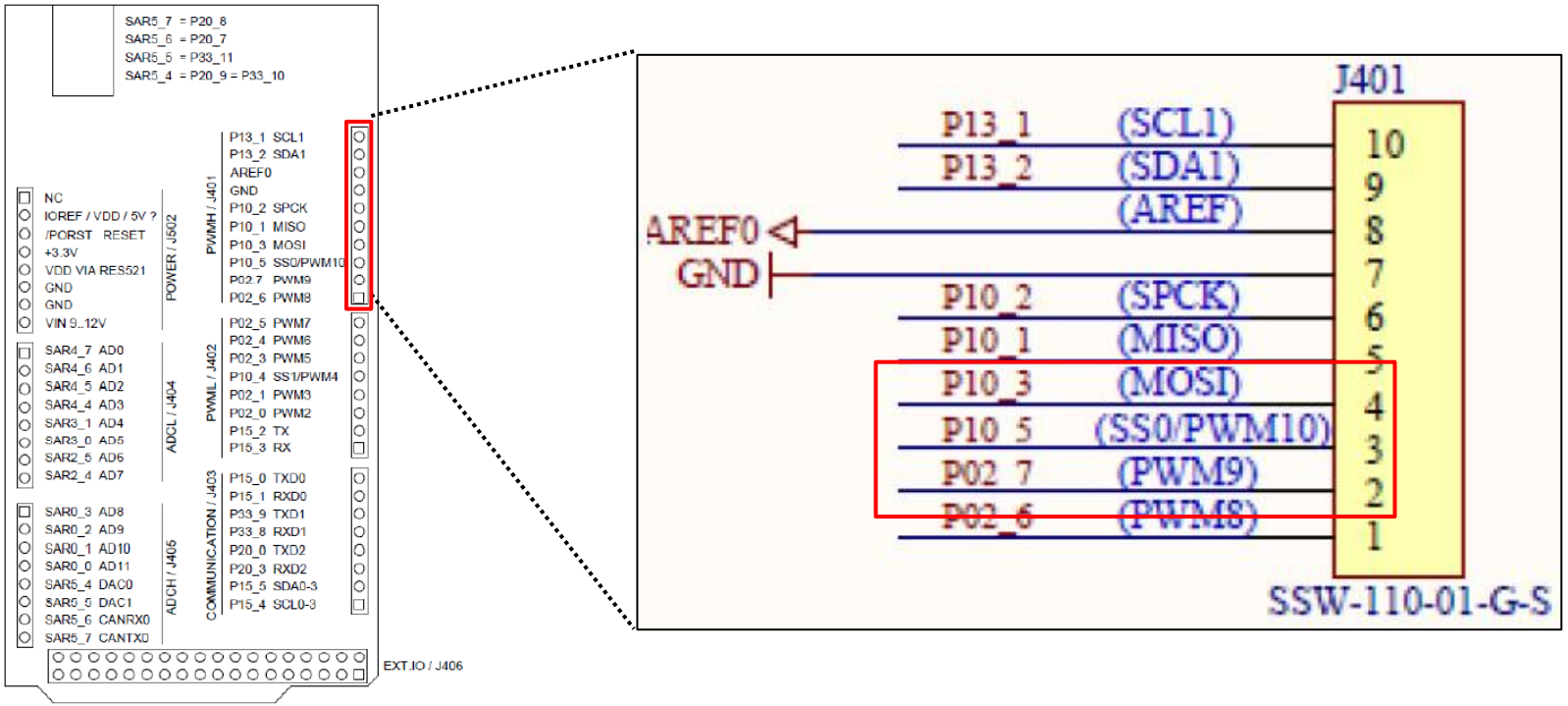
1. RGB LED 연결 정보 파악
   - ✓ RGB LED는 Easy Module Shield V1 확장 보드의 **Pin D9/D10/D11**과 연결되어 있다.
   - ✓ 타겟 보드는 Easy Module Shield V1 확장 보드의 Pin D9/D10/D11을 통해 RGB LED 출력을 보낼 수 있다.





RGB LED Example

1. RGB LED 연결 정보 파악
   - ✓ TC275 보드의 Schematic과 Datasheet를 확인했을 때, Easy Module Shield V1 확장 보드의 **Pin D9/D10/D11**과 연결되는 IO는 **PORT2의 Pin 7**과 **PORT10의 Pin 5/Pin3**다.
   - ✓ 해당 Pin의 출력이 High-level 일 때 LED는 켜지고, Low-level 일 때 LED는 꺼진다.



RGB LED Example

2. Data sheet 분석 : IO 설정
   - ✓ RGB LED RED를 사용하기 위해 연결된 Pin의 IO 설정이 필요하다.
   - ✓ RGB LED RED 제어를 위한 출력 신호를 내보내기 위해 해당 Pin을 **General-purpose output**으로 설정해야 한다.

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P02.7	I	General-purpose input	P02_IN.P7	P02_IOCR4. PC7	$0XXXX_B$
		GTM input	TIN7		
		QSPI3 input	SCLK3A		
		PSI5 input	PSIRX2B		
		SENT input	SENT1C		
		CCU60 input	CC61INC		
		CCU60 input	CCPOS1A		
		CCU61 input	T13HRB		
		GPT120 input	T3EUDA		
		DSADC input	DSCIN3B		
		CIF input	CIFD7		
		DSADC input	DSITR4E		
	O	General-purpose output	P02_OUT.P7		$1X000_B$
		GTM output	TOUT7		$1X001_B$
		Reserved	–		$1X010_B$
		QSPI3 output	SCLK3		$1X011_B$
		DSADC output	DSCOUT3		$1X100_B$
		VADC output	VADCEMUX01		$1X101_B$
		SENT output	SPC1		$1X110_B$
		CCU60 output	CC61		$1X111_B$

RGB LED Example

2. Data sheet 분석 : IO 설정
   - ✓ RGB LED GREEN를 사용하기 위해 연결된 Pin의 IO 설정이 필요하다.
   - ✓ RGB LED GREEN 제어를 위한 출력 신호를 내보내기 위해 해당 Pin을 **General-purpose output**으로 설정해야 한다.

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P10.5	I	General-purpose input	P10_IN.P5	P10_IOCR4. PC5	$0XXXX_B$
		GTM input	TIN107		
		SCU input	HWCFG4		
		MSC0 input	INJ01		
	O	General-purpose output	P10_OUT.P5		$1X000_B$
		GTM output	TOUT107		$1X001_B$
		ASCLIN2 output	ATX2		$1X010_B$
		QSPI3 output	SLSO38		$1X011_B$
		QSPI1 output	SLSO19		$1X100_B$
		GPT120 output	T6OUT		$1X101_B$
		ASCLIN2 output	ASLSO2		$1X110_B$
		Reserved	–		$1X111_B$

RGB LED Example

2. Data sheet 분석 : IO 설정
   - ✓ RGB LED BLUE를 사용하기 위해 연결된 Pin의 IO 설정이 필요하다.
   - ✓ RGB LED BLUE제어를 위한 출력 신호를 내보내기 위해 해당 Pin을 **General-purpose output**으로 설정해야 한다.

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P10.3	I	General-purpose input	P10_IN.P3	P10_IOCR0. PC3	$0XXXX_B$
		GTM input	TIN105		
		QSPI1 input	MTSR1A		
		SCU input	REQ3		
		GPT120 input	T5INB		
	O	General-purpose output	P10_OUT.P3		$1X000_B$
		GTM output	TOUT105		$1X001_B$
		VADC output	VADCG6BFL3		$1X010_B$
		QSPI1 output	MTSR1		$1X011_B$
		MSC0 output	EN00		$1X100_B$
		MSC0 output	END02		$1X101_B$
		CAN node 2 output	TXDCAN2		$1X110_B$
		Reserved	–		$1X111_B$

RGB LED Example

2. Data sheet 분석 : PORT 설정 (1)
   - ✓ P02_IOCR Register는 PORT02의 Input/Output을 설정한다.
   - ✓ LED RED 가 PORT02의 Pin 7에 연결되어 있기 때문에 **P02_IOCR4 Register의 PC7 bits**를 설정한다.

Table 13-3 Registers Address Space

Module	Base Address	End Address	Note
P00	$F003\ A000_H$	$F003\ A0FF_H$	13 pins
P01	$F003\ A100_H$	$F003\ A1FF_H$	5 pins
P02	$F003\ A200_H$	$F003\ A2FF_H$	12 pins
P10	$F003\ B000_H$	$F003\ B0FF_H$	9 pins
P11	$F003\ B100_H$	$F003\ B1FF_H$	16 pins

P02_IOCR4 Register 주소: F003_A214h (F003A200h + 14h)

P02_IOCR4 Register 구조:

P0n_IOCR4 (n=0-2)
Port 0n Input/Output Control Register 4
($F003\ A014_H + n*100_H$) Reset Value: $1010\ 1010_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PC7					0			PC6					0		
rw					r			rw					r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PC5					0			PC4					0		
rw					r			rw					r		

Field	Bits	Type	Description
PC4, PC5, PC6, PC7	[7:3], [15:11], [23:19], [31:27]	rw	**Port Control for Port n Pin 4 to 7** This bit field determines the Port n line x functionality (x = 4-7) according to the coding table (see Table 13-5).
0	[2:0], [10:8], [18:16], [26:24]	r	**Reserved** Read as 0; should be written with 0.

RGB LED Example

2. Data sheet 분석 : PORT 설정 (1)
   - ✓ P10_IOCR Register는 PORT10의 Input/Output을 설정한다.
   - ✓ LED GREEN이 PORT10의 Pin 5에 연결되어 있기 때문에 **P10_IOCR4 Register의 PC5 bits**를 설정한다.

Table 13-3 Registers Address Space

Module	Base Address	End Address	Note
P00	$F003\ A000_H$	$F003\ A0FF_H$	13 pins
P01	$F003\ A100_H$	$F003\ A1FF_H$	5 pins
P02	$F003\ A200_H$	$F003\ A2FF_H$	12 pins
P10	$F003\ B000_H$	$F003\ B0FF_H$	9 pins
P11	$F003\ B100_H$	$F003\ B1FF_H$	16 pins

P10_IOCR4 Register 주소: F003_B014h (F003B000h + 14h)

P10_IOCR4 Register 구조:

Pn_IOCR4 (n=10-11)
Port n Input/Output Control Register 4
($F003\ A614_H + n*100_H$) **Reset Value: $1010\ 1010_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PC7					0			PC6					0		
rw					r			rw					r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PC5					0			PC4					0		
rw					r			rw					r		

Field	Bits	Type	Description
PC4, PC5, PC6, PC7	[7:3], [15:11], [23:19], [31:27]	rw	**Port Control for Port n Pin 4 to 7** This bit field determines the Port n line x functionality (x = 4-7) according to the coding table (see Table 13-5).
0	[2:0], [10:8], [18:16], [26:24]	r	**Reserved** Read as 0; should be written with 0.

RGB LED Example

2. Data sheet 분석 : PORT 설정 (1)
   - ✓ P10_IOCR Register는 PORT10의 Input/Output을 설정한다.
   - ✓ LED BLUE가 PORT10의 Pin 3에 연결되어 있기 때문에 **P10_IOCR0 Register의 PC3 bits**를 설정한다.

Table 13-3 Registers Address Space

Module	Base Address	End Address	Note
P00	$F003\ A000_H$	$F003\ A0FF_H$	13 pins
P01	$F003\ A100_H$	$F003\ A1FF_H$	5 pins
P02	$F003\ A200_H$	$F003\ A2FF_H$	12 pins
P10	$F003\ B000_H$	$F003\ B0FF_H$	9 pins
P11	$F003\ B100_H$	$F003\ B1FF_H$	16 pins

P10_IOCR0 Register 주소: F003_B010h (F003B000h + 10h)

P10_IOCR0 Register 구조:

Pn_IOCR0 (n=10-11)
Port n Input/Output Control Register 0
($F003\ A610_H + n^*100_H$) **Reset Value: $1010\ 1010_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PC3					0			PC2					0		
rw					r			rw					r		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PC1					0			PC0					0		
rw					r			rw					r		

Field	Bits	Type	Description
PC0, PC1, PC2, PC3	[7:3], [15:11], [23:19], [31:27]	rw	**Port Control for Port n Pin 0 to 3** This bit field determines the Port n line x functionality (x = 0-3) according to the coding table (see Table 13-5).
0	[2:0], [10:8], [18:16], [26:24]	r	**Reserved** Read as 0; should be written with 0.

RGB LED Example

2. Data sheet 분석 : PORT 설정 (2)

✓ PORT02의 Pin7과 PORT10의 Pin5/Pin3를 General-purpose output (push-pull)으로 설정하기 위해 각 IOCR Register의 **PC7, PC5, PC3 bits**를 **10000b**로 설정한다.

Table 13-5 PCx Coding

PCx[4:0]	I/O	Characteristics	Selected Pull-up / Pull-down / Selected Output Function
10000_B	Output	Push-pull	General-purpose output
10001_B			Alternate output function 1
10010_B			Alternate output function 2
10011_B			Alternate output function 3
10100_B			Alternate output function 4
10101_B			Alternate output function 5
10110_B			Alternate output function 6
10111_B			Alternate output function 7
11000_B		Open-drain	General-purpose output
11001_B			Alternate output function 1
11010_B			Alternate output function 2
11011_B			Alternate output function 3
11100_B			Alternate output function 4
11101_B			Alternate output function 5
11110_B			Alternate output function 6
11111_B			Alternate output function 7

RGB LED Example

2. Data sheet 분석 : PORT 출력 설정

   - ✓ P02_OMR Register는 PORT02의 출력을 설정한다.
   - ✓ PORT02의 Pin 7 출력을 설정하기 위해 **P02_OMR Register의 PCL7 bit와 PS7 bit**를 설정한다.
     - ✓ PCL7 bit만 Set 하면 P02.7 출력이 '0 (Low-level)'으로 Clear 된다.
     - ✓ PS7 bit만 Set 하면 P02.7 출력이 '1 (High-level)'로 Set 된다.
     - ✓ PCL7 bit와 PS7 bit를 동시에 Set 하면 P02.7 출력이 Toggle 된다.

Table 13-3 Registers Address Space

Module	Base Address	End Address	Note
P00	F003 A000$_H$	F003 A0FF$_H$	13 pins
P01	F003 A100$_H$	F003 A1FF$_H$	5 pins
P02	F003 A200$_H$	F003 A2FF$_H$	12 pins
P10	F003 B000$_H$	F003 B0FF$_H$	9 pins
P11	F003 B100$_H$	F003 B1FF$_H$	16 pins
P12	F003 B200$_H$	F003 B2FF$_H$	2 pins
P13	F003 B300$_H$	F003 B3FF$_H$	4 pins
P14	F003 B400$_H$	F003 B4FF$_H$	11 pins
P15	F003 B500$_H$	F003 B5FF$_H$	9 pins

P10_OMR Register 주소: F003_A204h (F003A200h + 4h)

P10_OMR Register 구조:

P0n_OMR (n=0-2)
Port 0n Output Modification Register (F003 A004$_H$ + n*100$_H$) **Reset Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PS 15	PS 14	PS 13	PS 12	PS 11	PS 10	PS 9	PS 8	PS 7	PS 6	PS 5	PS 4	PS 3	PS 2	PS 1	PS 0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Table 13-9 Function of the Bits PCLx and PSx

PCLx	PSx	Function
0	0	Bit Pn_OUT.Px is not changed.
0	1	Bit Pn_OUT.Px is set.
1	0	Bit Pn_OUT.Px is reset.
1	1	Bit Pn_OUT.Px is toggled.

RGB LED Example

2. Data sheet 분석 : PORT 출력 설정

   - ✓ P10_OMR Register는 PORT10의 출력을 설정한다.
   - ✓ PORT10의 Pin 5/3 출력을 설정하기 위해 **P10_OMR Register의 PCL5/3 bit와 PS5/3 bit**를 설정한다.
     - ✓ PCL5, PCL3 bit만 Set 하면 P10.5, P10.3 출력이 '0 (Low-level)'으로 Clear 된다.
     - ✓ PS5, PS3 bit만 Set 하면 P10.5, P10.3 출력이 '1 (High-level)'로 Set 된다.
     - ✓ PCL5, PCL3 bit와 PS5, PS3 bit를 동시에 Set 하면 P10.5, P10.3 출력이 Toggle 된다.

Table 13-3 Registers Address Space

Module	Base Address	End Address	Note
P00	F003 A000$_H$	F003 A0FF$_H$	13 pins
P01	F003 A100$_H$	F003 A1FF$_H$	5 pins
P02	F003 A200$_H$	F003 A2FF$_H$	12 pins
P10	F003 B000$_H$	F003 B0FF$_H$	9 pins
P11	F003 B100$_H$	F003 B1FF$_H$	16 pins
P12	F003 B200$_H$	F003 B2FF$_H$	2 pins
P13	F003 B300$_H$	F003 B3FF$_H$	4 pins
P14	F003 B400$_H$	F003 B4FF$_H$	11 pins
P15	F003 B500$_H$	F003 B5FF$_H$	9 pins

P10_OMR Register 주소: F003_B004h (F003B000h + 4h)

P10_OMR Register 구조:

Pn_OMR (n=10-15)
Port n Output Modification Register (F003 A604$_H$ + n*100$_H$) **Reset Value: 0000 0000$_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PS 15	PS 14	PS 13	PS 12	PS 11	PS 10	PS 9	PS 8	PS 7	PS 6	PS 5	PS 4	PS 3	PS 2	PS 1	PS 0
w	w	w	w	w	w	w	w	w	w	w	w	w	w	w	w

Table 13-9 Function of the Bits PCLx and PSx

PCLx	PSx	Function
0	0	Bit Pn_OUT.Px is not changed.
0	1	Bit Pn_OUT.Px is set.
1	0	Bit Pn_OUT.Px is reset.
1	1	Bit Pn_OUT.Px is toggled.

RGB LED Example

3. 프로그래밍

1) RGB LED가 연결된 PORG02 Pin7과 PORT10 Pin 5/3에 대한 IO 설정을 한다.

```
70  /* Define PORT02/10 Registers for RGB LED */
71  #define PORT02_BASE     (0xF003A200)
72  #define PORT02_IOCR4    (*(volatile unsigned int*)(PORT02_BASE + 0x14))
73  #define PORT02_OMR      (*(volatile unsigned int*)(PORT02_BASE + 0x04))
74
75  #define PC7             27
76  #define PCL7            23
77  #define PS7             7
78
79  #define PORT10_BASE     (0xF003B000)
80  #define PORT10_IOCR4    (*(volatile unsigned int*)(PORT10_BASE + 0x14))
81  #define PORT10_IOCR0    (*(volatile unsigned int*)(PORT10_BASE + 0x10))
82  #define PORT10_OMR      (*(volatile unsigned int*)(PORT10_BASE + 0x04))
83
84  #define PC5             11
85  #define PC3             27
86  #define PCL5            21
87  #define PCL3            19
88  #define PS5             5
89  #define PS3             3
```

PORT IO 설정관련 레지스터 주소 및 비트 필드 정의

```
152 /* Initialize RGB LED */
153 void init_RGBLED(void)
154 {
155     /* Reset IOCR0 bits */
156     PORT02_IOCR4 &= ~((0x1F) << PC7);
157     PORT10_IOCR4 &= ~((0x1F) << PC5);
158     PORT10_IOCR0 &= ~((0x1F) << PC3);
159
160     /* Set PC bits in IOCR0 with push-pull(2b10000) */
161     PORT02_IOCR4 |= ((0x10) << PC7);
162     PORT10_IOCR4 |= ((0x10) << PC5);
163     PORT10_IOCR0 |= ((0x10) << PC3);
164 }
```

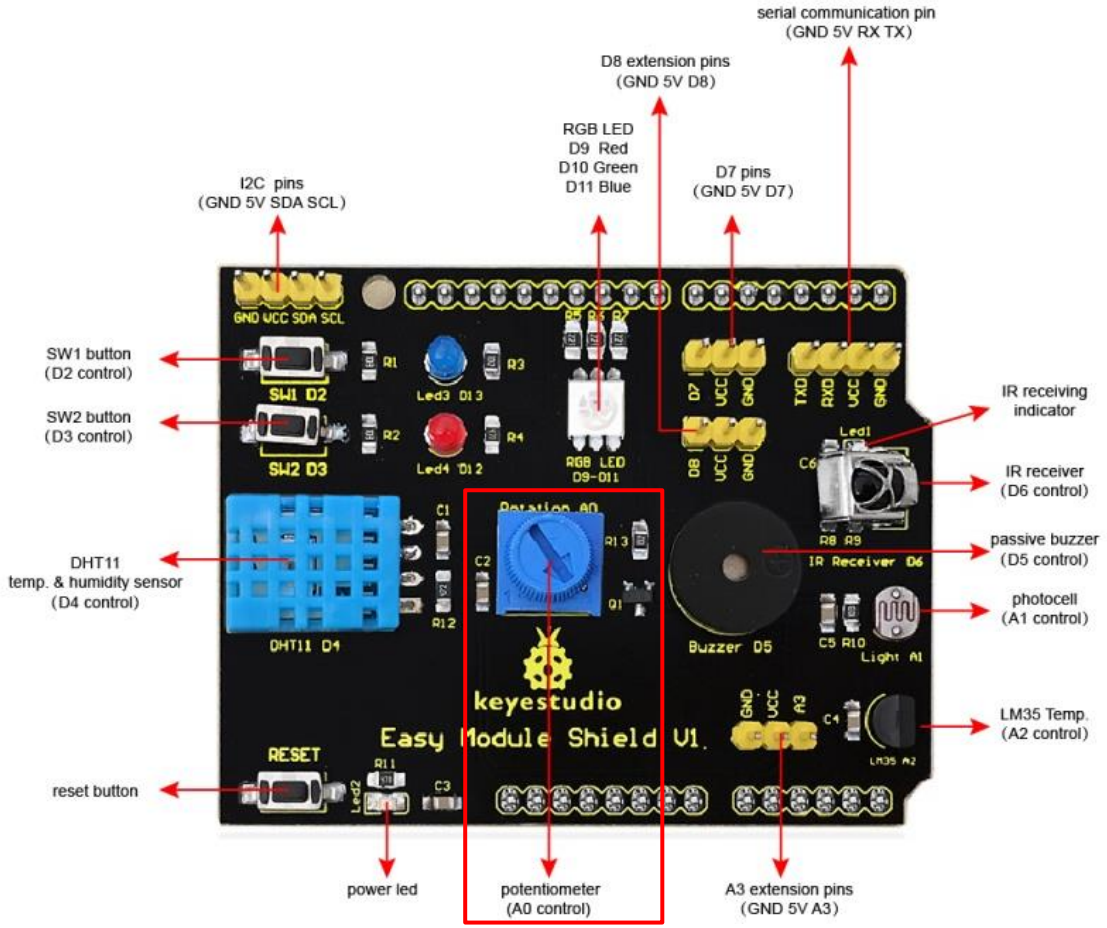
PORT IO 설정 함수

ADC Example

- ADC로 읽은 전압 값의 범위에 따른 RGB LED 출력 제어
  1. 새로운 예제를 위한 프로젝트를 생성한다.
  2. 원하는 동작을 위해 레지스터와 메모리에 직접 접근해서 값을 써야한다.
  3. Potentiometer 사용을 위해 Board Schematic과 Datasheet에서 Potentiometer 연결 정보를 파악한다.
  4. ADC 모듈의 동작 원리를 파악하고 메모리 맵을 분석한다.
  5. 분석 결과를 활용해 임베디드 프로그래밍을 한다.

ADC Example

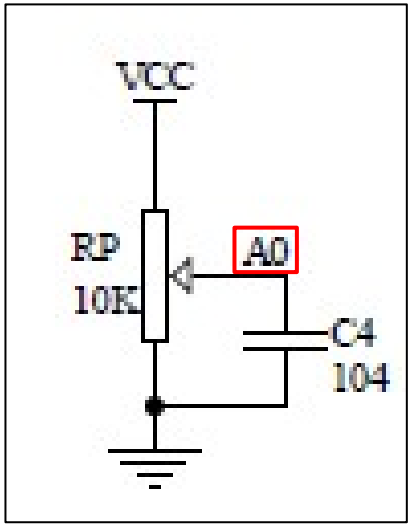
1. Potentiometer 연결 정보 파악
   - ✓ 타겟 보드인 Shield Buddy TC275에는 사용 가능한 Potentiometer가 없기 때문에 **Easy Module Shield V1 확장 보드**의 Potentiometer를 사용한다.

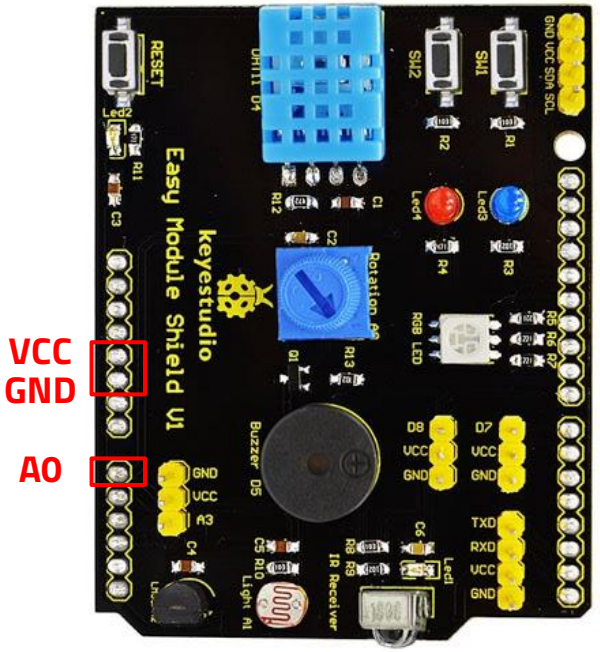


ADC Example

1. Potentiometer 연결 정보 파악
   - ✓ Potentiometer는 Easy Module Shield V1 확장 보드의 **Pin A0**과 연결되어 있다.
   - ✓ Potentiometer에 따라 Pin A0의 전압 값이 달라진다.
   - ✓ 타겟 보드는 Easy Module Shield V1 확장 보드의 Pin A0을 통해 아날로그 전압 값을 입력 받을 수 있다.

     (정상적인 Potentiometer 동작을 위해 VCC 및 GND도 연결해야 한다.)





ADC Example

1. Potentiometer 연결 정보 파악

✓ TC275 보드의 Schematic과 Datasheet를 확인했을 때, Easy Module Shield V1 확장 보드의 **Pin A0**와 연결되는 IO는 **SAR4의 Pin 7**이다.

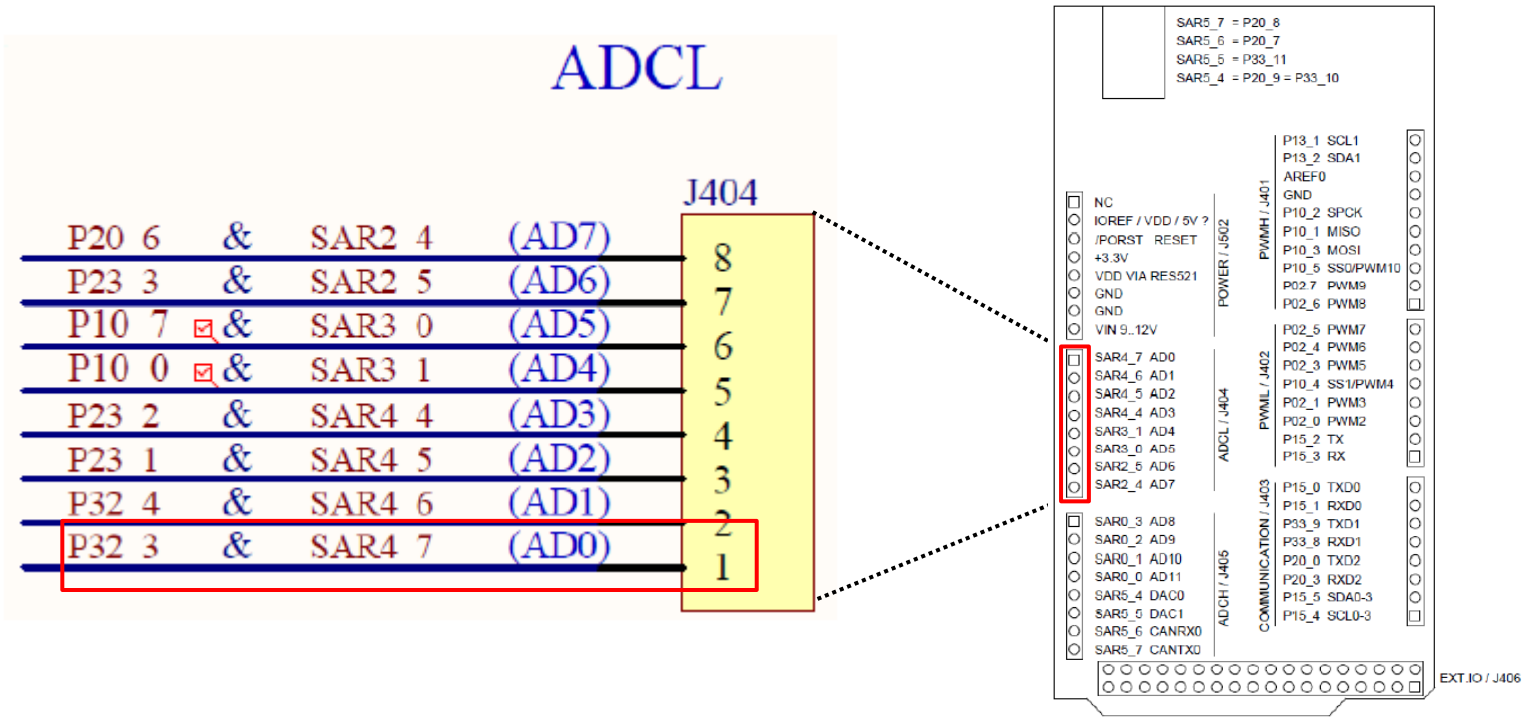


Table 2-15 Analog Inputs (cont'd)

Pin	Symbol	Ctrl	Type	Function
35	AN36	I	**S / HighZ / VDDM**	Analog input 34
	VADCG4.4			VADC analog input channel 4 of group 4
	DS3PA			DSADC: positive analog input of channel of DSADC 3, pin A
	SENT6A			SENT input channel 6, pin A
34	AN37	I	**S / HighZ / VDDM**	Analog input 37
	VADCG4.5			VADC analog input channel 5 of group 4
	DS3NA			DSADC: negative analog input of channel of DSADC 3, pin A
	SENT7A			SENT input channel 7, pin A
33	AN38	I	**S / HighZ / VDDM**	Analog input 38
	VADCG4.6			VADC analog input channel 6 of group 4
	DS3PB			DSADC: positive analog input of channel of DSADC 3, pin B
	SENT8A			SENT input channel 8, pin A
32	AN39	I	**S / HighZ / VDDM**	Analog input 39
	VADCG4.7			VADC analog input channel 7 of group 4
	DS3NB			DSADC: negative analog input of channel of DSADC 3, pin B
	SENT9A			SENT input channel 9, pin A
31	AN44	I	**D / HighZ / VDDM**	Analog input 44
	VADCG5.4			VADC analog input channel 4 of group 5
	DS3PC			DSADC: positive analog input of channel of DSADC 3, pin C
30	AN45	I	**D / HighZ / VDDM**	Analog input 45
	VADCG5.5			VADC analog input channel 5 of group 5
	DS3NC			DSADC: negative analog input of channel of DSADC 3, pin C
29	AN46	I	**D / HighZ / VDDM**	Analog input 46
	VADCG5.6			VADC analog input channel 6 of group 5
	DS3PD			DSADC: positive analog input of channel of DSADC 3, pin D
28	AN47	I	**D / HighZ / VDDM**	Analog input 47
	VADCG5.7			VADC analog input channel 7 of group 5
	DS3ND			DSADC: negative analog input of channel of DSADC 3, pin D

Table 28-12 Analog Connections in the TC27x (cont'd)

Signal	Dir.	Source/Destin.[1)]	Description
G1CH4	I	AN12	analog input channel 4 of group 1
G1CH5	I	AN13	analog input channel 5 of group 1
G1CH6	I	AN14	analog input channel 6 of group 1
G1CH7	I	AN15	analog input channel 7 of group 1
G2CH0 (AltRef)	I	AN16	analog input channel 0 of group 2
G2CH1 (MD)	I	AN17	analog input channel 1 of group 2
G2CH2 (MD)	I	AN18	analog input channel 2 of group 2
G2CH3 (PDD)	I	AN19	analog input channel 3 of group 2
G2CH4	I	AN20 (X)	analog input channel 4 of group 2
G2CH5	I	AN21 (X)	analog input channel 5 of group 2
G2CH6	I	AN22	analog input channel 6 of group 2
G2CH7	I	AN23	analog input channel 7 of group 2
G3CH0 (AltRef)	I	AN24, P40.0 (X)	analog input channel 0 of group 3
G3CH1 (MD)	I	AN25, P40.1 (X)	analog input channel 1 of group 3
G3CH2 (MD)	I	AN26, P40.2	analog input channel 2 of group 3
G3CH3 (PDD)	I	AN27, P40.3	analog input channel 3 of group 3
G3CH4	I	AN28	analog input channel 4 of group 3
G3CH5	I	AN29	analog input channel 5 of group 3
G3CH6	I	AN30	analog input channel 6 of group 3
G3CH7	I	AN31	analog input channel 7 of group 3
G4CH0 (AltRef)	I	AN32, P40.4	analog input channel 0 of group 4
G4CH1 (MD)	I	AN33, P40.5	analog input channel 1 of group 4
G4CH2 (MD)	I	AN34	analog input channel 2 of group 4
G4CH3 (PDD, noAltref)	I	AN35	analog input channel 3 of group 4
G4CH4	I	AN36, P40.6 (X)	analog input channel 4 of group 4
G4CH5	I	AN37, P40.7 (X)	analog input channel 5 of group 4
G4CH6	I	AN38, P40.8 (X)	analog input channel 6 of group 4
G4CH7	I	AN39, P40.9 (X)	analog input channel 7 of group 4
G5CH0 (AltRef)	I	AN40	analog input channel 0 of group 5
G5CH1 (MD)	I	AN41	analog input channel 1 of group 5

ADC Example

2. Data sheet 분석 : 아날로그 입력 Pin

- ✓ Potentiometer에 의한 아날로그 전압 값을 입력 받기 위해서는 아날로그 입력 Pin이 필요하다.
- ✓ Potentiometer가 연결된 Pin AN0은 아날로그 전압 값을 측정하여 디지털 값으로 변환하는 **VADC (Versatile Analog-to-Digital Converter)의 Channel 7 of Group 4**과 연결되어 있는 **아날로그 입력 Pin**이다.
- ✓ 해당 Pin은 아날로그 입력 전용 Pin이기 때문에 추가적인 IO 설정이 필요하지 않다.

ADC Example

2. Data sheet 분석 : VADC 구조 분석 (1)

- ✓ VADC는 8 개의 **Group(0~7)**으로 구성되며 각 Group은 **ADC Kernel**로 구성된다.
- ✓ Group은 Request Control에 의한 Conversion Request에 따라 동작을 수행한다.
- ✓ Conversion Request는 여러 Analog Input Channels 중 하나를 선택하며 AD Converter는 해당 Channel을 디지털 값으로 변환한다.
- ✓ 변환된 값은 Result Handling으로 전달되며 정해진 위치에 저장된다.

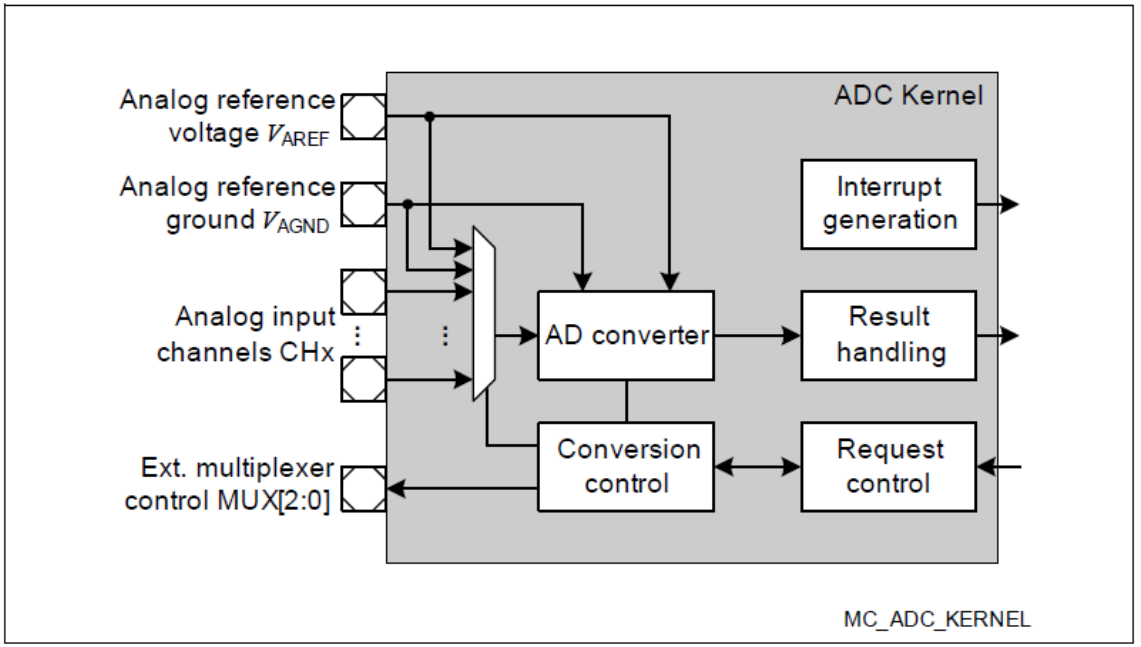


Figure 28-2 ADC Kernel Block Diagram

ADC Example

2. Data sheet 분석 : VADC 구조 분석 (2)

- ✓ **Request Control**은 Conversion Request를 생성하는 2개의 **Request Source**와 각 Request Source에서 생성된 Conversion Request를 관리하는 **Request Source Arbiter**로 구성된다.
- ✓ Request Source는 Timer Unit / External Request / Software에 따라 Conversion Request를 생성한다.
- ✓ 각 Request Source는 필요에 따라 Enable / Disable 할 수 있다.

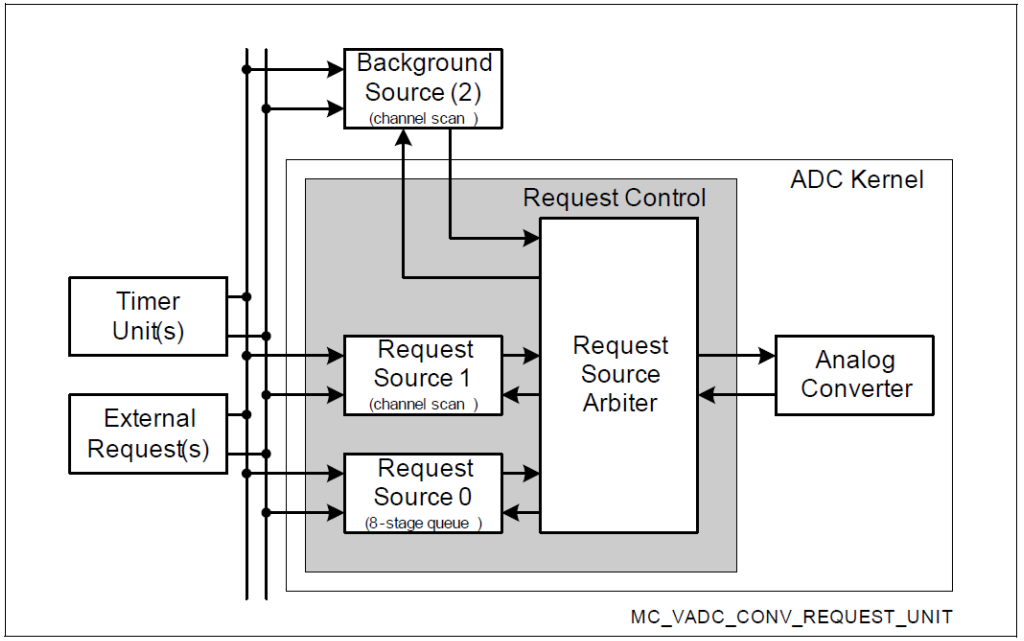


Figure 28-3 Conversion Request Unit

ADC Example

2. Data sheet 분석 : VADC 구조 분석 (3)
   - ✓ **Result Handling**은 변환된 결과인 디지털 값을 설정된 위치에 저장한다.
   - ✓ 디지털 값은 추가적으로 압축 및 필터링 될 수 있다.
   - ✓ 디지털 값은 **Global Result Register** 또는 **Group Result Register**에 저장될 수 있으며 각 Result Register는 디지털 값 뿐만 아니라 새로운 결과 값이 저장되었음을 나타내는 Valid Flag도 포함한다.
   - ✓ Valid Flag는 새로운 디지털 값이 저장되면 Set 되고, 해당 Result Register가 읽히면 Clear 된다.

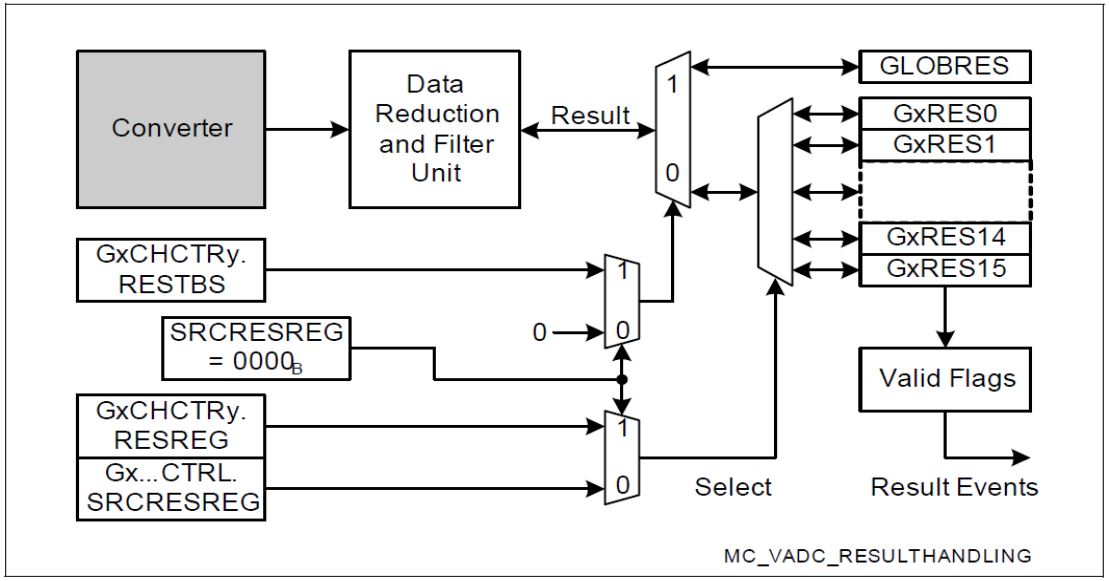
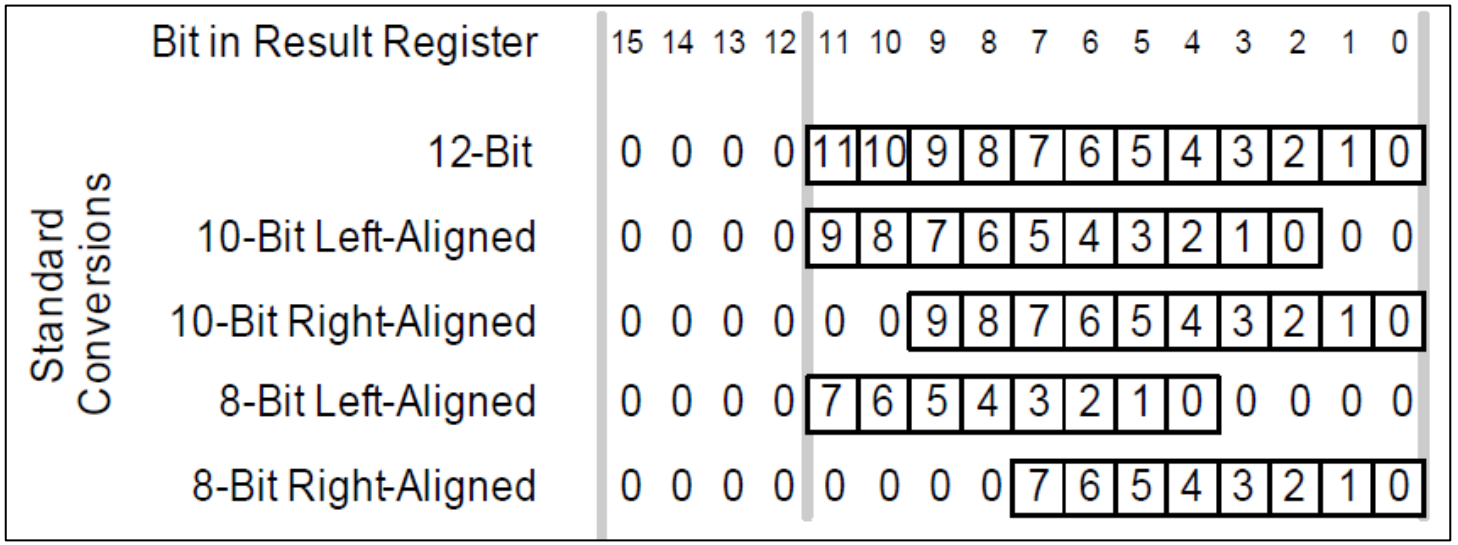


Figure 28-18 Conversion Result Storage

ADC Example

2. Data sheet 분석 : VADC 구조 분석 (4)
   - ✓ VADC의 변환된 결과인 디지털 값은 **8-bit에서 12-bit의 크기**를 가진다.
   - ✓ 설정에 따라 디지털 값은 Result Register 내에 **Right-Aligned** 또는 **Left-Aligned**로 저장된다.
   - ✓ 따라서, 설정에 따라 결과값을 적절하게 읽어 처리해야 한다.



ADC Example

2. Data sheet 분석 : VADC Enable 설정
   - ✓ VADC_CLC Register는 VADC 모듈의 Enable 설정을 한다.
   - ✓ VADC 모듈을 Enable 하기 위해 **DISR bit**를 **0**으로 설정한다.
   - ✓ VADC 모듈이 Enable 되어 있는지 확인하기 위해 **DISS bit**가 **0**인지 확인한다.

VADC_CLC Register 주소: F002_0000h (F0020000h + 0h)

VADC_CLC Register 구조:

Table 28-10 Registers Address Space

Module	Base Address	End Address	Note
VADC	$F002\ 0000_H$	$F002\ 3FFF_H$	

CLC
Clock Control Register **(0000_H)** **Reset Value: $0000\ 0003_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	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
0	0	0	0	0	0	0	0	0	0	0	0	EDIS	0	DISS	DISR
r	r	r	r	r	r	r	r	r	r	r	r	rw	r	r	rw

Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable/disable control of the module. Also the analog section is disabled by clearing ANONS. 0_B On request: enable the module clock 1_B Off request: stop the module clock
DISS	1	r	**Module Disable Status Bit** 0_B Module clock is enabled 1_B Off: module is not clocked
0	2	r	**Reserved, write 0, read as 0**
EDIS	3	rw	**Sleep Mode Enable Control** Used to control module's reaction to sleep mode. 0_B Sleep mode request is enabled and functional 1_B Module disregards the sleep mode control signal
0	[31:4]	r	**Reserved, write 0, read as 0**

ADC Example

2. Data sheet 분석 : System Critical Register 설정 (1)
   - ✓ 설정해야 하는 VADC_CLC Register는 System Critical Register이기 때문에 Write Protected (System ENDINIT, End-of-Initialization) 되어 있다.
   - ✓ 해당 Register를 수정하기 위해서는 System ENDINIT을 해제해야 한다.
   - ✓ SCU_WDTCPU0CON0 Register는 **System Critical Register**의 **System ENDINIT을 설정/해제**한다.

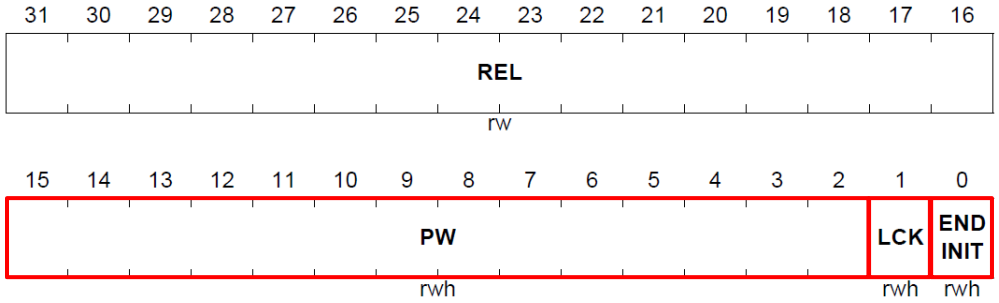
SCU_WDTCPU0CON0 Register 주소: F003_6100h
(F0036000h + 100h)

SCU_WDTCPU0CON0 Register 구조:

Table 7-27 Registers Address Spaces - SCU Kernel Registers

Module	Base Address	End Address	Note
SCU	$F003\ 6000_H$	$F003\ 63FF_H$	-

WDTCPU0CON0
CPU0 WDT Control Register 0 **(100_H)** **Reset Value: $FFFC\ 000E_H$**



ADC Example

2. Data sheet 분석 : System Critical Register 설정 (2)
   - ✓ **ENDINIT bit**는 System ENDINIT의 설정 상태를 나타내며 Modify Access를 통해서만 수정이 가능하다.
   - ✓ **LCK bit**는 SCU_WDTCPU0CON0 Register의 Lock 상태를 나타내며 해당 Register의 Lock 상태는 Password Access를 통해 Unlock 되고, Modify Access를 통해 Lock 된다.
   - ✓ **PW bits**는 SCU_WDTCPU0CON0 Register에 접근하기 위한 Password를 저장하며 해당 값을

Field	Bits	Type	Description
ENDINIT	0	rwh	**End-of-Initialization Control Bit** 0_B Access to Endinit-protected registers is permitted. 1_B Access to Endinit-protected registers is not permitted. This bit must be written with a '1' during a Password Access or Check Access (although this write is only used for the password-protection mechanism and is not stored). This bit must be written with the required ENDINIT update value during a Modify Access.
LCK	1	rwh	**Lock Bit to Control Access to WDTxCON0** 0_B Register WDTxCON0 is unlocked 1_B Register WDTxCON0 is locked (default after ApplicationReset) The current value of LCK is controlled by hardware. It is cleared after a valid Password Access to WDTxCON0 when WDTxSR.US is 0 (or when WDTxSR.US is 1 and the SMU is in RUN mode), and it is automatically set again after a valid Modify Access to WDTxCON0. During a write to WDTxCON0, the value written to this bit is only used for the password-protection mechanism and is not stored. This bit must be cleared during a Password Access to WDTxCON0, and set during a Modify Access to WDTxCON0. A Check Access does not clear LCK.

PW	[15:2]	rwh	**User-Definable Password Field for Access to WDTxCON0** This bit field is written with an initial password value during a Modify Access. A read from this bitfield returns this initial password, but bits [7:2] are inverted (toggled) to ensure that a simple read/write is not sufficient to service the WDT. If corresponding WDTxSR.PAS = 0 then this bit field must be written with its current contents during a Password Access or Check Access. If corresponding WDTxSR.PAS = 1 then this bit field must be written with the next password in the LFSR sequence during a Password Access or Check Access The default password after Application Reset is 00000000111100_B A-step silicon: Bits [7:2] must be written with 111100_B during Password Access and Modify Access. Read returns 000011_B for these bits.

ADC Example

2. Data sheet 분석 : System Critical Register 설정 (3)
   - ✓ SCU_WDTCPU0CON0 Register에 적절한 값을 Write하여 **Password Access**를 수행한다.
   - ✓ **Password Access**는 **SCU_WDTCPU0CON0 Register의 Lock 상태를 해제**하며 과정은 다음과 같다.
     1. SCU_WDTCPU0CON0 Register의 값을 읽어 REL bits, PW bits를 파악한다.
     2. Bits[7:2] (PW bits의 일부)가 반전되어 읽히기 때문에 이를 반전시켜 정확한 PW bits를 얻는다.
     3. Write 할 값의 bits[31:16]은 읽혀진 REL bits 값으로 설정하고 bit[15:2]는 앞서 구한 정확한 PW bits 값으로 설정한다.
     4. Write 할 값의 bit[1]은 0으로 설정하고, bit[0]은 1로 설정한다.
     5. 설정된 값을 SCU_WDTCPU0CON0 Register에 한번에 쓴다.
     6. SCU_WDTCPU0CON0 Register의 LCK bit를 확인하여 Lock 상태가 해제되었는지 파악한다. (Password Access가 정상적으로 수행되면 Lock 상태가 해제되며 LCK bit가 0으로 설정된다.)
   - ✓ Password Access를 통해 SCU_WDTCPU0CON0 Register의 Lock 상태가 해제되면 Modify Access를 통해 System ENDINIT을 설정/해제할 수 있다.

ADC Example

2. Data sheet 분석 : System Critical Register 설정 (4)

   ✓ SCU_WDTCPU0CON0 Register에 적절한 값을 Write하여 **Modify Access**를 수행한다.

   ✓ **Modify Access**는 **System ENDINIT을 설정/해제**하며 과정은 다음과 같다.

   1. SCU_WDTCPU0CON0 Register의 값을 읽어 REL bits, PW bits를 파악한다.
   2. Bits[7:2] (PW bits의 일부)가 반전되어 읽히기 때문에 이를 반전시켜 정확한 PW bits를 얻는다.
   3. Write 할 값의 bits[31:16]은 읽혀진 REL bits 값으로 설정하고 bit[15:2]는 앞서 구한 정확한 PW bits 값으로 설정한다.
   4. Write 할 값의 bit[1]은 1로 설정하고, bit[0]은 적절한 값으로 설정한다.
      (System ENDINIT 설정: bit[0] = 1, System ENDINIT 해제 : bit[0] = 0)
   5. 설정된 값을 SCU_WDTCPU0CON0 Register에 한번에 쓴다.
   6. SCU_WDTCPU0CON0 Register의 LCK bit를 확인하여 Lock 상태가 다시 설정되었는지 파악한다.
      (Modify Access가 정상적으로 수행되면 Lock 상태가 설정되며 LCK bit가 1로 설정된다.)

   ✓ Modify Access를 통해 System ENDINIT을 해제하면 System Critical Register를 수정할 수 있으며,

ADC Example

2. Data sheet 분석 : Group 설정 (1)

- ✓ VADC_GxARBPR Register는 Group의 Request Source Arbiter에 대한 설정을 한다.
- ✓ VADC의 여러 Group 중, Potentiometer와 연결된 Pin AN0이 Group 4에 입력되기 때문에 **VADC_G4ARBPR Register**를 설정한다.
- ✓ Pin AN0에 대한 Conversion Request만 생성하면 되기 때문에 Request Source Arbiter는 **Request Source 0에 대한 설정 (PRIO0 bits / CSM0 bit / ASEN0 bit)**만 수행한다.

VADC_G4ARBPR Register 주소: F002_1484h (F0020000h + 1484h)

VADC_G4ARBPR Register 구조:

Table 28-10 Registers Address Space

Module	Base Address	End Address	Note
VADC	$F002\ 0000_H$	$F002\ 3FFF_H$	

GxARBPR (x = 0 - 7)
Arbitration Priority Register, Group x
($x * 0400_H + 0484_H$) **Reset Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	ASEN2	ASEN1	ASEN0	0	0	0	0	0	0	0	0
r	r	r	r	r	rw	rw	rw	r	r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	CSM2	0	PRIO2		CSM1	0	PRIO1		CSM0	0	PRIO0	
r	r	r	r	rw	r	rw		rw	r	rw		rw	r	rw	

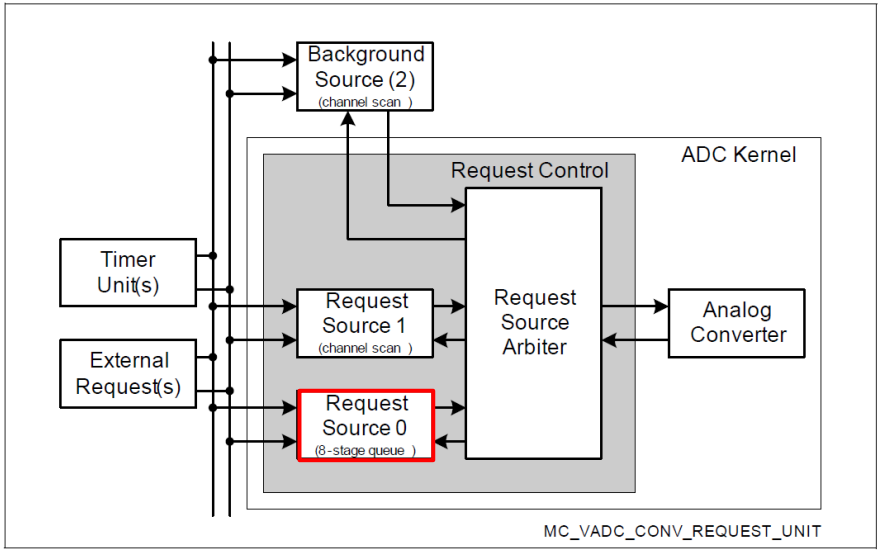


Figure 28-3 Conversion Request Unit

ADC Example

2. Data sheet 분석 : Group 설정 (2)

- ✓ Request Source 0의 우선 순위를 가장 높게 설정하기 위해 **PRIO0 bits**를 **11b**로 설정한다.
- ✓ Request Source 0의 Conversion Request가 현재 수행하고 있는 Conversion이 끝날 때까지 기다린 후에 실행되도록 설정하기 위해 **CSM0 bit**를 **0b**로 설정한다.
- ✓ Request Source 0을 Enable 하기 위해 **ASEN0 bit**를 **1b**로 설정한다.

GxARBPR (x = 0 - 7)
Arbitration Priority Register, Group x
($x * 0400_H + 0484_H$) **Reset Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	AS EN2	AS EN1	AS EN0	0	0	0	0	0	0	0	0
r	r	r	r	r	rw	rw	rw	r	r	r	r	r	r	r	r

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	CSM 2	0	PRIO 2		CSM 1	0	PRIO 1		CSM 0	0	PRIO 0	
r	r	r	r	rw	r	rw		rw	r	rw		rw	r	rw	

Field	Bits	Type	Description
PRIO0, PRIO1, PRIO2	[1:0], [5:4], [9:8]	rw	**Priority of Request Source x** Arbitration priority of request source x (in slot x) 00_B Lowest priority is selected. ... 11_B Highest priority is selected.
CSM0, CSM1, CSM2	3, 7, 11	rw	**Conversion Start Mode of Request Source x** 0_B Wait-for-start mode 1_B Cancel-inject-repeat mode, i.e. this source can cancel conversion of other sources.
0	2, 6, 10, [23:12]	r	**Reserved, write 0, read as 0**
ASENy (y = 0 - 2)	24 + y	rw	**Arbitration Slot y Enable** Enables the associated arbitration slot of an arbiter round. The request source bits are not modified by write actions to ASENR. 0_B The corresponding arbitration slot is disabled and considered as empty. Pending conversion requests from the associated request source are disregarded. 1_B The corresponding arbitration slot is enabled. Pending conversion requests from the associated request source are arbitrated.
0	[31:27]	r	**Reserved, write 0, read as 0**

ADC Example

2. Data sheet 분석 : Group 설정 (3)

- ✓ VADC_GxQMR Register는 Group의 Request Source에 대한 설정을 한다.
- ✓ Group 4의 Request Source 0을 사용하기 때문에 **VADC_G4QMR0 Register**를 설정한다.
- ✓ Software를 통해 Request Source 0의 Conversion Request 생성을 가능하게 하기 위해 **ENGT bit**를 **01b**로 설정한다.
- ✓ 초기화시, Request Source 0에 의한 Conversion Request를 Clear 하기위해 **FLUSH bit**를 **1**로 설정한다.

VADC_G4QMR0 Register 주소: F002_1504h
(F0020000h + 1504h)

VADC_G4QMR0 Register 구조:

Table 28-10 Registers Address Space

Module	Base Address	End Address	Note
VADC	$F002\ 0000_H$	$F002\ 3FFF_H$	

GxQMR0 (x = 0 - 7)
Queue 0 Mode Register, Group x
($x * 0400_H + 0504_H$) **Reset Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	RPT DIS
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	CEV	FLU SH	TR EV	CLR V	0	0	0	0	0	EN TR	ENGT	
r	r	r	r	w	w	w	w	r	r	r	r	r	rw	rw	

Field	Bits	Type	Description
ENGT	[1:0]	rw	**Enable Gate** Selects the gating functionality for source 0/2. 00_B No conversion requests are issued 01_B Conversion requests are issued if a valid conversion request is pending in the queue 0 register or in the backup register 10_B Conversion requests are issued if a valid conversion request is pending in the queue 0 register or in the backup register and REQGTx = 1 11_B Conversion requests are issued if a valid conversion request is pending in the queue 0 register or in the backup register and REQGTx = 0 *Note: REQGTx is the selected gating signal.*
FLUSH	10	w	**Flush Queue** 0_B No action 1_B Clear all queue entries (including backup stage) and the event flag EV. The queue contains no more valid entry.

ADC Example

2. Data sheet 분석 : Group 설정 (4)

   ✓ **TREV bit**는 Conversion Request를 생성하는 트리거 이벤트를 소프트웨어적으로 발생시킨다.

   ✓ 따라서, Conversion Request를 생성하고자 할 때 해당 bit를 **1**로 설정한다.

GxQMR0 (x = 0 - 7)
Queue 0 Mode Register, Group x

(x * 0400_H + 0504_H) **Reset Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	RPT DIS
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	rw

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	CEV	FLU SH	TR EV	CLR V	0	0	0	0	0	EN TR	ENGT	
r	r	r	r	w	w	w	w	r	r	r	r	r	rw	rw	

Field	Bits	Type	Description
CLRV	8	w	**Clear Valid Bit** 0_B No action 1_B The next pending valid queue entry in the sequence and the event flag EV are cleared. If there is a valid entry in the queue backup register (QBUR.V = 1), this entry is cleared, otherwise the entry in queue register 0 is cleared.
TREV	9	w	**Trigger Event** 0_B No action 1_B Generate a trigger event by software
FLUSH	10	w	**Flush Queue** 0_B No action 1_B Clear all queue entries (including backup stage) and the event flag EV. The queue contains no more valid entry.

ADC Example

2. Data sheet 분석 : Group 설정 (5)

- ✓ VADC_GxARBCFG Register는 Group의 AD Converter에 대한 설정을 한다.
- ✓ Group 4을 사용하기 때문에 **VADC_G4ARBCFG Register**를 설정한다.
- ✓ AD Converter를 Normal Operation Mode로 동작시키기 위해 **ANONC bits**를 **11b**로 설정한다.

VADC_G4ARBCFG Register 주소: F002_1480h
(F0020000h + 1480h)

VADC_G4ARBCFG Register 구조:

Table 28-10 Registers Address Space

Module	Base Address	End Address	Note
VADC	$F002\ 0000_H$	$F002\ 3FFF_H$	

GxARBCFG (x = 0 - 7)
Arbitration Configuration Register, Group x
($x * 0400_H + 0480_H$) **Reset Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SAMPLE	BUSY	CALS	CAL	0	0	SYNRUN	CHNR					CSRC		ANONS	
rh	rh	rh	rh	r	r	rh	rh					rh		rh	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	ARBM	0	ARBRND		0	0	ANONC	
r	r	r	r	r	r	r	r	rw	r	rw		r	r	rw	

- **ANONS = 11_B: Normal Operation**
  The converter is active, conversions are started immediately. Requires no wakeup time.
- **ANONS = 10_B or 01_B: Reserved**
- **ANONS = 00_B: Converter switched Off** (default after reset)

Field	Bits	Type	Description
ANONC	[1:0]	rw	**Analog Converter Control** Defines the value of bitfield ANONS in a stand-alone converter or a converter in master mode. Coding see ANONS or Section 28.4.1.
0	[3:2]	r	**Reserved, write 0, read as 0**
ARBRND	[5:4]	rw	**Arbitration Round Length** Defines the number of arbitration slots per arb. round (arbitration round length = t_{ARB}).[1)] 00_B 4 arbitration slots per round ($t_{ARB} = 4 / f_{ADCD}$) 01_B 8 arbitration slots per round ($t_{ARB} = 8 / f_{ADCD}$) 10_B 16 arbitration slots per round ($t_{ARB} = 16 / f_{ADCD}$) 11_B 20 arbitration slots per round ($t_{ARB} = 20 / f_{ADCD}$)
0	6	r	**Reserved, write 0, read as 0**

ADC Example

Table 28-4 Sample Time Coding

STCS / STCE	Additional Clock Cycles	Sample Time
$0\ 0000_B$	0	$2 / f_{ADCI}$
$0\ 0001_B$	1	$3 / f_{ADCI}$
…	…	…
$0\ 1111_B$	15	$17 / f_{ADCI}$
$1\ 0000_B$	16	$18 / f_{ADCI}$
$1\ 0001_B$	32	$34 / f_{ADCI}$
…	…	…
$1\ 1110_B$	240	$242 / f_{ADCI}$
$1\ 1111_B$	256	$258 / f_{ADCI}$

2. Data sheet 분석 : Group 설정 (6)

- ✓ VADC_GxICLASS Register는 Group의 Input Class에 대한 설정을 한다.
- ✓ Analog Input Channel은 미리 설정된 Input Class 중 하나에 속하게 되며 해당 Input Class의 설정이 반영된다.
- ✓ Group 4의 Input Class 0을 설정하기 위해 **VADC_G4ICLASS0 Register**를 설정한다.
- ✓ Sample Time과 Conversion Mode를 설정하기 위해 **STCS bits / CMS bits**를 설정한다.

VADC_G4ICLASS0 Register 주소: F002_14A0h
(F0020000h + 14A0h)

VADC_G4ICLASS0 Register 구조:

Table 28-10 Registers Address Space

Module	Base Address	End Address	Note
VADC	$F002\ 0000_H$	$F002\ 3FFF_H$	

GxICLASS0 (x = 0 - 7)
Input Class Register 0, Group x

($x * 0400_H + 04A0_H$) **Reset Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	CME			0	0	0	STCE				
r	r	r	r	r	rw			r	r	r	rw				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	CMS			0	0	0	STCS				
r	r	r	r	r	rw			r	r	r	rw				

Field	Bits	Type	Description
STCS	[4:0]	rw	**Sample Time Control for Standard Conversions** Number of additional clock cycles to be added to the minimum sample phase of 2 analog clock cycles: Coding and resulting sample time see Table 28-4. For conversions of external channels, the value from bitfield STCE can be used.
0	[7:5]	r	**Reserved, write 0, read as 0**
CMS	[10:8]	rw	**Conversion Mode for Standard Conversions** 000_B 12-bit conversion 001_B 10-bit conversion 010_B 8-bit conversion 011_B Reserved 100_B Reserved 101_B 10-bit fast compare mode 110_B Reserved 111_B Reserved
0	[15:11]	r	**Reserved, write 0, read as 0**

ADC Example

2. Data sheet 분석 : Channel 설정 (1)

   ✓ VADC_GxCHCTR Register는 Group의 Analog Input Channel에 대한 설정을 한다.

   ✓ Potentiometer가 연결된 Pin AN0이 Group 4의 Input Channel 7에 입력되기 때문에 **VADC_G4CHCTR7 Register**를 설정한다.

   ✓ 해당 Input Channel에 대한 Input Class / Result Register / Result Align을 설정하기 위해 **ICLSEL bits / RESREG bits / RESPOS bit**를 설정한다.

VADC_G4CHCTR7 Register 주소: F002_161Ch
(F0020000h + 161Ch)

VADC_G4CHCTR7 Register 구조:

Table 28-10 Registers Address Space

Module	Base Address	End Address	Note
VADC	$F002\ 0000_H$	$F002\ 3FFF_H$	

G0CHCTRy (y = 0 - 7)
Group 0, Channel y Ctrl. Reg. ($0600_H + y * 0004_H$) **Reset Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	BWD EN	BWD CH		0	0	0	0	0	0	RES POS	RES TBS	RESREG			
r	rw	rw		r	r	r	r	r	r	rw	rw	rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BNDSELX				REF SEL	SY NC	CHEV MODE		BNDSELU		BNDSELL		0	0	ICLSEL	
rw				rw	rw	rw		rw		rw		r	r	rw	

ADC Example

2. Data sheet 분석 : Channel 설정 (2)

- ✓ Analog Input Channel 7의 Input Class를 앞서 설정한 Group 4의 Input Class 0으로 설정하기 위해 **ICLSEL bits**를 **00b**로 설정한다.
- ✓ Analog Input Channel 7의 디지털 값을 Group 0의 Result Register 1에 저장하기 위해 **RESREG bits**를 **0001b**로 설정한다.
- ✓ Analog Input Channel 7의 디지털 값을 Right-Aligned로 저장하기 위해 **RESPOS bit**를 **1**로 설정한다.

G4CHCTRy (y = 0 - 7)
Group 4, Channel y Ctrl. Reg. (1600_H + y * 0004_H) **Reset Value: 0000 0000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	BWDEN	BWDCH		0	0	0	0	0	0	RESPOS	RESTBS	RESREG			
r	rw	rw		r	r	r	r	r	r	rw	rw	rw			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BNDSELX				REFSEL	SYNC	CHEVMODE		BNDSELU		BNDSELL		0	0	ICLSEL	
rw				rw	rw	rw		rw		rw		r	r	rw	

Field	Bits	Type	Description
ICLSEL	[1:0]	rw	**Input Class Select** 00_B Use group-specific class 0 01_B Use group-specific class 1 10_B Use global class 0 11_B Use global class 1
RESREG	[19:16]	rw	**Result Register** 0000_B Store result in group result register GxRES0 ... 1111_B Store result in group result register GxRES15

Field	Bits	Type	Description
RESPOS	21	rw	**Result Position** 0_B Store results left-aligned 1_B Store results right-aligned

ADC Example

2. Data sheet 분석 : Conversion Request 설정

- ✓ VADC_GxQINR Register는 Request Source의 Conversion Request에 대한 설정을 한다.
- ✓ Group 4의 Request Source 0을 사용하기 때문에 **VADC_G4QINR0 Register**를 설정한다.
- ✓ Analog Input Channel 7을 입력으로 설정하기 위해 **REQCHNR bits**를 **7**으로 설정한다.
- ✓ Single-shot Mode로 설정하기 위해 **RF bit**를 **0b**로 설정한다.

  (RF=1b로 설정하면 Conversion후 다시 Conversion Request가 발생하여 Continuous Mode로 동작)

VADC_G4QINR0 Register 주소: F002_1510h
(F0020000h + 1510h)

VADC_G4QINR0 Register 구조:

Table 28-10 Registers Address Space

Module	Base Address	End Address	Note
VADC	$F002\ 0000_H$	$F002\ 3FFF_H$	

GxQINR0 (x = 0 - 7)
Queue 0 Input Register, Group x
($x * 0400_H + 0510_H$) **Reset Value: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	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
0	0	0	0	0	0	0	0	EXTR	ENSI	RF	REQCHNR				
r	r	r	r	r	r	r	r	w	w	w	w				

Field	Bits	Type	Description
REQCHNR	[4:0]	w	**Request Channel Number** Defines the channel number to be converted
RF	5	w	**Refill** 0_B No refill: this queue entry is converted once and then invalidated 1_B Automatic refill: this queue entry is automatically reloaded into QINR0 when the related conversion is started
ENSI	6	w	**Enable Source Interrupt** 0_B No request source interrupt 1_B A request source event interrupt is generated upon a request source event (related conversion is finished)

ADC Example

2. Data sheet 분석 : Result Register 설정
   - ✓ VADC_GxRES Register는 변환된 디지털 값에 대한 정보를 저장한다.
   - ✓ Analog Input Channel 7의 디지털 값을 Result Register 1에 저장하도록 설정했기 때문에 **VADC_G4RES1 Register**를 확인한다.
   - ✓ 변환이 끝나 새로운 디지털 값이 저장되었는지 확인하기 위해 **VF bit**가 **1**인지 확인한다.
   - ✓ 변환된 디지털 값을 확인하기 위해 Align을 고려하여 **RESULT bits**를 확인한다.

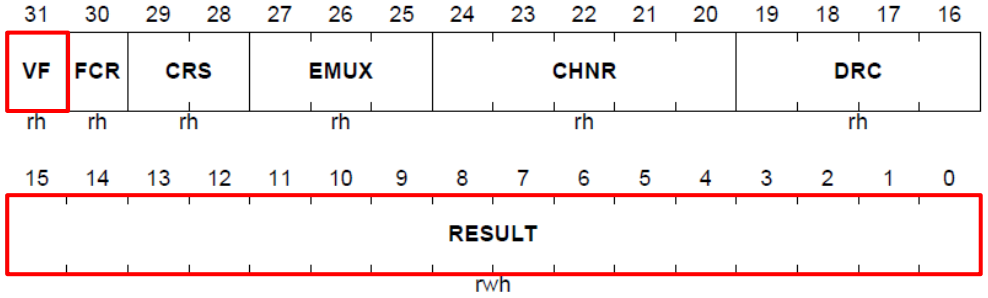
VADC_G4RES1 Register 주소: F002_1704h
(F0020000h + 1704h)

VADC_G4RES1 Register 구조:

Table 28-10 Registers Address Space

Module	Base Address	End Address	Note
VADC	$F002\ 0000_H$	$F002\ 3FFF_H$	

G0RESy (y = 0 - 15)
Group 0 Result Register y **($0700_H + y * 0004_H$)** **Reset Value: $0000\ 0000_H$**



Field	Bits	Type	Description
RESULT	[15:0]	rwh	**Result of Most Recent Conversion** The position of the result bits within this bitfield depends on the configured operating mode. Refer to Section 28.7.2.
VF	31	rh	**Valid Flag** Indicates a new result in bitfield RESULT or bit FCR. 0_B No new result available 1_B Bitfield RESULT has been updated with new result value and has not yet been read, or bit FCR has been updated

ADC Example

3. 프로그래밍

1) RGB LED가 연결된 PORT에 대한 설정을 수행하는 함수를 구현한다.

```
70  /* Define PORT02/10 Registers for RGB LED */
71  #define PORT02_BASE     (0xF003A200)
72  #define PORT02_IOCR4    (*(volatile unsigned int*)(PORT02_BASE + 0x14))
73  #define PORT02_OMR      (*(volatile unsigned int*)(PORT02_BASE + 0x04))
74
75  #define PC7             27
76  #define PCL7            23
77  #define PS7             7
78
79  #define PORT10_BASE     (0xF003B000)
80  #define PORT10_IOCR4    (*(volatile unsigned int*)(PORT10_BASE + 0x14))
81  #define PORT10_IOCR0    (*(volatile unsigned int*)(PORT10_BASE + 0x10))
82  #define PORT10_OMR      (*(volatile unsigned int*)(PORT10_BASE + 0x04))
83
84  #define PC5             11
85  #define PC3             27
86  #define PCL5            21
87  #define PCL3            19
88  #define PS5             5
89  #define PS3             3
```

PORT IO 설정관련 레지스터 주소 및 비트 필드 정의

```
152 /* Initialize RGB LED */
153 void init_RGBLED(void)
154 {
155     /* Reset IOCR0 bits */
156     PORT02_IOCR4 &= ~((0x1F) << PC7);
157     PORT10_IOCR4 &= ~((0x1F) << PC5);
158     PORT10_IOCR0 &= ~((0x1F) << PC3);
159
160     /* Set PC bits in IOCR0 with push-pull(2b10000) */
161     PORT02_IOCR4 |= ((0x10) << PC7);
162     PORT10_IOCR4 |= ((0x10) << PC5);
163     PORT10_IOCR0 |= ((0x10) << PC3);
164 }
```

PORT IO 설정 함수

ADC Example

3. 프로그래밍

2) VADC를 설정하기 위한 함수를 구현한다.

① SCU_WDTCPU0CON0 Register를 통해 Password/Modify Access를 수행하여 System ENDINIT을 해제한다.

② VADC_CLC Register를 통해 VADC 모듈을 Enable 한다.

③ SCU_WDTCPU0CON0 Register를 통해 Password/Modify Access를 수행하여 System ENDINIT을 설정한다.

④ VADC_G4ARBPR Register를 통해 Group 4의 Request Source 0에 대한 Request Source Arbiter 설정을 한다.

⑤ VADC_G4QMR0 Register를 통해 Request Source 0에 대한 설정을 한다.

⑥ VADC_G4ARBCFG Register를 통해 Analog Converter의 동작 모드를 Normal Operation으로 설정한다.

⑦ VADC_G4ICLASS0 Register를 통해 Input Class 0을 설정한다.

⑧ VADC_G4CHCTR7 Register를 통해 Analog Input Channel 7에 대한 설정을 한다.

ADC Example

3. 프로그래밍

2) VADC를 설정하기 위한 함수를 구현한다.

```
/* SCU Registers */
#define SCU_BASE            (0xF0036000)
#define SCU_WDT_CPU0CON0    (*(volatile unsigned int*)(SCU_BASE + 0x100))

#define LCK                 1
#define ENDINIT             0

/* VADC Registers */
#define VADC_BASE       (0xF0020000)
#define VADC_CLC        (*(volatile unsigned int*)(VADC_BASE + 0x000))
#define VADC_GLOBCFG    (*(volatile unsigned int*)(VADC_BASE + 0x080))
#define VADC_G4ARBCFG   (*(volatile unsigned int*)(VADC_BASE + 0x1480))
#define VADC_G4ARBPR    (*(volatile unsigned int*)(VADC_BASE + 0x1484))
#define VADC_G4ICLASS0  (*(volatile unsigned int*)(VADC_BASE + 0x14A0))
#define VADC_G4QMR0     (*(volatile unsigned int*)(VADC_BASE + 0x1504))
#define VADC_G4QINR0    (*(volatile unsigned int*)(VADC_BASE + 0x1510))
#define VADC_G4CHCTR7   (*(volatile unsigned int*)(VADC_BASE + 0x161C))
#define VADC_G4RES1     (*(volatile unsigned int*)(VADC_BASE + 0x1704))

#define DISS            1
#define DISR            0
#define ANONC           0
#define ASEN0           24
#define CSM0            3
#define PRIO0           0
#define CMS             8
#define STCS            0
#define FLUSH           10
#define TREV            9
#define ENGT            0
#define RF              5
#define REQCHNR         0
#define RESPOS          21
#define RESREG          16
#define ICLSEL          0
#define VF              31
#define RESULT          0
```

VADC 설정관련 레지스터 주소 및 비트 필드 정의

ADC Example

3. 프로그래밍

2) VADC를 설정하기 위한 함수를 구현한다.

```
165 void init_VADC(void)
166 {
167     /* VADC Enable */
168     /* Password Access to unlock WDTCPU0CON0 */
169 ①   SCU_WDT_CPU0CON0 = ((SCU_WDT_CPU0CON0 ^ 0xFC) & ~(1 << LCK)) | (1 << ENDINIT);
170     while((SCU_WDT_CPU0CON0 & (1 << LCK)) != 0);
171
172     /* Modify Access to clear ENDINIT bit */
173     SCU_WDT_CPU0CON0 = ((SCU_WDT_CPU0CON0 ^ 0xFC) | (1 << LCK)) & ~ (1 << ENDINIT);
174     while((SCU_WDT_CPU0CON0 & (1 << LCK)) == 0);
175
176 ②   VADC_CLC &= ~(1 << DISR);                // Enable VADC Module
177
178     /* Password Access to unlock WDTSCPU0CON0 */
179 ③   SCU_WDT_CPU0CON0 = ((SCU_WDT_CPU0CON0 ^ 0xFC) & ~(1 << LCK)) | (1 << ENDINIT);
180     while((SCU_WDT_CPU0CON0 & (1 << LCK)) != 0);
181
182     /* Modify Access to clear ENDINIT bit */
183     SCU_WDT_CPU0CON0 = ((SCU_WDT_CPU0CON0 ^ 0xFC) | (1 << LCK)) | (1 << ENDINIT);
184     while((SCU_WDT_CPU0CON0 & (1 << LCK)) == 0);
185
186     while((VADC_CLC & (1 << DISS)) != 0);     // Wait until module is enabled
187
188 ④   VADC_G4ARBPR |= ((0x3) << PRIO0);         // Highest Priority for Request Source 0
189     VADC_G4ARBPR &= ~(1 << CSM0);             // Conversion Start Mode : Wait-for-start mode
190     VADC_G4ARBPR |= (1 << ASEN0);             // Arbitration Source Input 0 Enable
```

ADC Example

3. 프로그래밍

2) VADC를 설정하기 위한 함수를 구현한다.

```
192 ⑤ VADC_G4QMR0  &= ~((0x3) << ENGT);          // Enable Conversion Requests
193   VADC_G4QMR0  |= ((0x1) << ENGT);
194
195   VADC_G4QMR0  |= (1 << FLUSH);              // Clear all Queue Entries
196
197 ⑥ VADC_G4ARBCFG |= ((0x3) << ANONC);         // Analog Converter : Normal Operation
198
199 ⑦ VADC_G4ICLASS0 &= ~((0x7) << CMS);         // Group-specific Class 0
200                                               // Conversion Mode : Standard Conversion (12-bit)
201
202   /* VADC Group 4 Channel 7 Setting */
203   VADC_G4CHCTR7 |= (1 << RESPOS);             // Read Results Right-aligned
204 ⑧ VADC_G4CHCTR7 &= ~((0xF) << RESREG);       // Store Result in Group Result Register G4RES1
205   VADC_G4CHCTR7 |= (1 << RESREG);
206   VADC_G4CHCTR7 &= ~((0x3) << ICLSEL);        // Use Group-specific Class 0
207 }
```

VADC 설정 함수

ADC Example

3. 프로그래밍

3) VADC의 Conversion Request를 생성하는 함수를 구현한다.

① VADC_G0QINR0 Register를 통해 Request Source 0에서 생성할 Conversion Request의 Analog Input Channel(Channel 7)을 설정한다.

② VADC_G0QINR0 Register를 통해 해당 Conversion Request가 Single-shot Mode로 동작하도록 설정한다.

③ VADC_G0QMR0 Register를 통해 Conversion Request를 생성하는 트리거 이벤트를 소프트웨어적으로 발생시킨다.

```
209 void VADC_startConversion(void)
210 {
211     /* No fill and Start Queue */
212 ①   VADC_G4QINR0 &= ~(0x1F);                  // Request Channel Number : 7
213     VADC_G4QINR0 |= (0x07);
214
215 ②   VADC_G4QINR0 &= ~(1 << RF);               // No fill : it is converted once
216
217 ③   VADC_G4QMR0 |= (1 << TREV);               // Generate a Trigger Event
218 }
```

ADC Example

3. 프로그래밍

4) VADC의 Result Register를 읽어오는 함수를 구현한다.

① VADC_G0RES1 Register의 VF bit를 통해 변환이 끝나 새로운 결과 값이 저장되기를 기다린다.

② VADC_G0RES1 Register를 Masking 하여 변환된 디지털 값만 읽어온다.

③ 해당 디지털 값을 반환한다.

```
220 unsigned int VADC_readResult(void)
221 {
222     unsigned int result;
223
224 ① while((VADC_G4RES1 & (1 << VF)) == 0);          // Wait until New Result Available
225
226 ② result = (VADC_G4RES1 & ((0xFFFF) << RESULT));  // Read Result
227
228 ③ return result;
229 }
```

ADC Example

3. 프로그래밍

5) 동작에 따라 'main' 함수를 구현한다. (필요한 레지스터 / 비트 필드 및 함수 프로토타입을

```
/* SCU Registers */
#define SCU_BASE            (0xF0036000)
#define SCU_WDT_CPU0CON0    (*(volatile unsigned int*)(SCU_BASE + 0x100))

#define LCK                 1
#define ENDINIT             0

/* VADC Registers */
#define VADC_BASE       (0xF0020000)
#define VADC_CLC        (*(volatile unsigned int*)(VADC_BASE + 0x000))
#define VADC_GLOBCFG    (*(volatile unsigned int*)(VADC_BASE + 0x080))
#define VADC_G4ARBCFG   (*(volatile unsigned int*)(VADC_BASE + 0x1480))
#define VADC_G4ARBPR    (*(volatile unsigned int*)(VADC_BASE + 0x1484))
#define VADC_G4ICLASS0  (*(volatile unsigned int*)(VADC_BASE + 0x14A0))
#define VADC_G4QMR0     (*(volatile unsigned int*)(VADC_BASE + 0x1504))
#define VADC_G4QINR0    (*(volatile unsigned int*)(VADC_BASE + 0x1510))
#define VADC_G4CHCTR7   (*(volatile unsigned int*)(VADC_BASE + 0x161C))
#define VADC_G4RES1     (*(volatile unsigned int*)(VADC_BASE + 0x1704))

#define DISS            1
#define DISR            0
#define ANONC           0
#define ASEN0           24
#define CSM0            3
#define PRIO0           0
#define CMS             8
#define STCS            0
#define FLUSH           10
#define TREV            9
#define ENGT            0
#define RF              5
#define REQCHNR         0
#define RESPOS          21
#define RESREG          16
#define ICLSEL          0
#define VF              31
#define RESULT          0
```

```
/* Define PORT02/10 Registers for RGB LED */
#define PORT02_BASE     (0xF003A200)
#define PORT02_IOCR4    (*(volatile unsigned int*)(PORT02_BASE + 0x14))
#define PORT02_OMR      (*(volatile unsigned int*)(PORT02_BASE + 0x04))

#define PC7             27
#define PCL7            23
#define PS7             7

#define PORT10_BASE     (0xF003B000)
#define PORT10_IOCR4    (*(volatile unsigned int*)(PORT10_BASE + 0x14))
#define PORT10_IOCR0    (*(volatile unsigned int*)(PORT10_BASE + 0x10))
#define PORT10_OMR      (*(volatile unsigned int*)(PORT10_BASE + 0x04))

#define PC5             11
#define PC3             27
#define PCL5            21
#define PCL3            19
#define PS5             5
#define PS3             3

void init_RGBLED(void);
void init_VADC(void);
void VADC_startConversion(void);
unsigned int VADC_readResult(void);
```

ADC Example

3. 프로그래밍

5) 동작에 따라 'main' 함수를 구현한다. (앞서 구현한 함수들을 호출한다.)

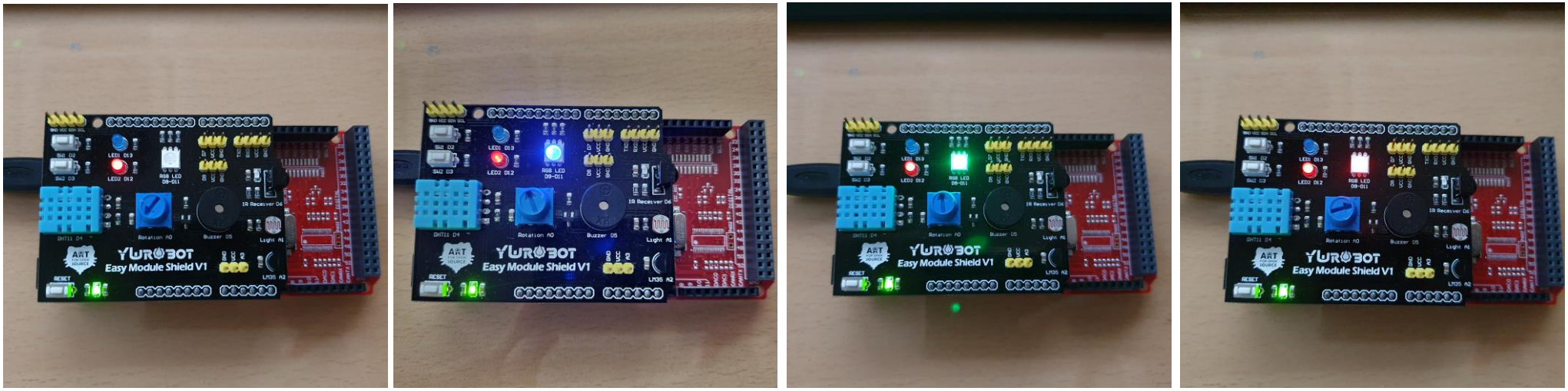
```
97  int core0_main(void)
98  {
99      IfxCpu_enableInterrupts();
100
101     /* !!WATCHDOG0 AND SAFETY WATCHDOG ARE DISABLED HERE!!
102      * Enable the watchdogs and service them periodically if it is required
103      */
104     IfxScuWdt_disableCpuWatchdog(IfxScuWdt_getCpuWatchdogPassword());
105     IfxScuWdt_disableSafetyWatchdog(IfxScuWdt_getSafetyWatchdogPassword());
106
107     /* Wait for CPU sync event */
108     IfxCpu_emitEvent(&g_cpuSyncEvent);
109     IfxCpu_waitEvent(&g_cpuSyncEvent, 1);
110
111     unsigned int adcResult;
112
113     /* Initialization */
114     init_RGBLED();              // Initialize PORT
115     init_VADC();                // Initialize VADC
116
```

```
118     while(1)
119     {
120         VADC_startConversion();
121         adcResult = VADC_readResult();
122
123         if(adcResult >= 3096)
124         {
125             PORT02_OMR |= (1<<PS7);             // Set LED RED
126             PORT10_OMR |= (1<<PCL5);            // Clear LED GREEN
127             PORT10_OMR |= (1<<PCL3);            // Clear LED BLUE
128         }
129         else if(adcResult >= 2048)
130         {
131             PORT02_OMR |= (1<<PCL7);             // Clear LED RED
132             PORT10_OMR |= (1<<PS5);              // Set LED GREEN
133             PORT10_OMR |= (1<<PCL3);             // Clear LED BLUE
134         }
135         else if(adcResult >= 1024)
136         {
137             PORT02_OMR |= (1<<PCL7);             // Clear LED RED
138             PORT10_OMR |= (1<<PCL5);             // Clear LED GREEN
139             PORT10_OMR |= (1<<PS3);              // Set LED BLUE
140         }
141         else
142         {
143             PORT02_OMR |= (1<<PCL7);             // Clear LED RED
144             PORT10_OMR |= (1<<PCL5);             // Clear LED GREEN
145             PORT10_OMR |= (1<<PCL3);             // Clear LED BLUE
146         }
147     }
148     return (1);
149 }
```

ADC Example

4. 동작 확인

   ✓ Build 및 Debug 후 (‘Resume’ 버튼 클릭), Potentiometer를 돌려보며 RGB-LED가 켜지는 것을 확인한다.



실습

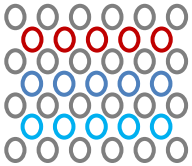
1. Reference Code 수행
2. Light Sensor(Group4, Channel 6) ADC 입력 측정
   - ✓ ADC 입력값에 따라 3색 LED 구동 → 입력값 기준 자유
   - ✓ 입력값 8번 입력받아서 평균값으로 구현
   - ✓ 결과 값을 Printf로 출력(1초 이상)

```
/* SCU Registers */
#define SCU_BASE            (0xF0036000)
#define SCU_WDT_CPU0CON0    (*(volatile unsigned int*)(SCU_BASE + 0x100))

#define LCK                 1
#define ENDINIT             0

/* VADC Registers */
#define VADC_BASE           (0xF0020000)
#define VADC_CLC            (*(volatile unsigned int*)(VADC_BASE + 0x000))
#define VADC_GLOBCFG        (*(volatile unsigned int*)(VADC_BASE + 0x080))
#define VADC_G4ARBCFG       (*(volatile unsigned int*)(VADC_BASE + 0x1480))
#define VADC_G4ARBPR        (*(volatile unsigned int*)(VADC_BASE + 0x1484))
#define VADC_G4ICLASS0      (*(volatile unsigned int*)(VADC_BASE + 0x14A0))
#define VADC_G4QMR0         (*(volatile unsigned int*)(VADC_BASE + 0x1504))
#define VADC_G4QINR0        (*(volatile unsigned int*)(VADC_BASE + 0x1510))
#define VADC_G4CHCTR6       (*(volatile unsigned int*)(VADC_BASE + 0x1618))
#define VADC_G4RES1         (*(volatile unsigned int*)(VADC_BASE + 0x1704))

#define DISS                1
#define DISR                0
#define ANONC               0
#define ASEN0               24
#define CSM0                3
#define PRIO0               0
#define CMS                 8
#define STCS                0
#define FLUSH               10
#define TREV                9
#define ENGT                0
#define RF                  5
#define REQCHNR             0
#define RESPOS              21
#define RESREG              16
#define ICLSEL              0
#define VF                  31
#define RESULT              0
#define CHNR                20
```

```
void init_VADC(void)
{
    /* VADC Enable */
    /* Password Access to unlock WDTCPU0CON0 */
    SCU_WDT_CPU0CON0 = ((SCU_WDT_CPU0CON0 ^ 0xFC) & ~(1 << LCK)) | (1 << ENDINIT);
    while((SCU_WDT_CPU0CON0 & (1 << LCK)) != 0);

    /* Modify Access to clear ENDINIT bit */
    SCU_WDT_CPU0CON0 = ((SCU_WDT_CPU0CON0 ^ 0xFC) | (1 << LCK)) & ~ (1 << ENDINIT);
    while((SCU_WDT_CPU0CON0 & (1 << LCK)) == 0);

    VADC_CLC &= ~(1 << DISR);                   // Enable VADC Module

    while((VADC_CLC & (1 << DISS)) != 0);       // Wait until module is enabled

    //VADC_GLOBCFG |= ((1<<31) | (1<<15) | 0x9);

    VADC_G4ARBPR |= ((0x3) << PRIO0);           // Highest Priority for Request Source 0
    VADC_G4ARBPR &= ~(1 << CSM0);               // Conversion Start Mode : Wait-for-start mode
    VADC_G4ARBPR |= (1 << (ASEN0));             // Arbitration Source Input 0 Enable

    VADC_G4QMR0  &= ~((0x3) << ENGT);           // Enable Conversion Requests
    VADC_G4QMR0  |= ((0x1) << ENGT);
    VADC_G4QMR0  |= (1 << FLUSH);               // Clear all Queue Entries

    VADC_G4ARBCFG |= ((0x3) << ANONC);          // Analog Converter : Normal Operation

    VADC_G4ICLASS0 &= ~((0x7) << CMS);          // Group-specific Class 0
    VADC_G4ICLASS0 &= ~((0x1F) << STCS);        // Conversion Mode : Standard Conversion (12-bit)
                                                // Additional Sample Time for Standard Conversion : 0
    /* VADC Group 4 Channel 6 Setting */
    VADC_G4CHCTR6 |= (1 << RESPOS);             // Read Results Right-aligned
    VADC_G4CHCTR6 &= ~((0xF) << RESREG);        // Store Result in Group Result Register G4RES1
    VADC_G4CHCTR6 |= (1 << RESREG);
    VADC_G4CHCTR6 &= ~((0x3) << ICLSEL);        // Use Group-specific Class 0
}

void VADC_startConversion(void)
{
    /* No fill and Start Queue */
    VADC_G4QINR0 &= ~(0x1F);                    // Request Channel Number : 6
    VADC_G4QINR0 |= (0x06);

    VADC_G4QINR0 &= ~(1 << RF);                 // No fill : it is converted once

    VADC_G4QMR0 |= (1 << TREV);                 // Generate a Trigger Event
}
```

Q & A

Thank you for your attention



Architecture and
Compiler
for Embedded Systems Lab.

School of Electronics Engineering, KNU

ACE Lab (hn02301@gmail.com)