Infineon TC275 Motor

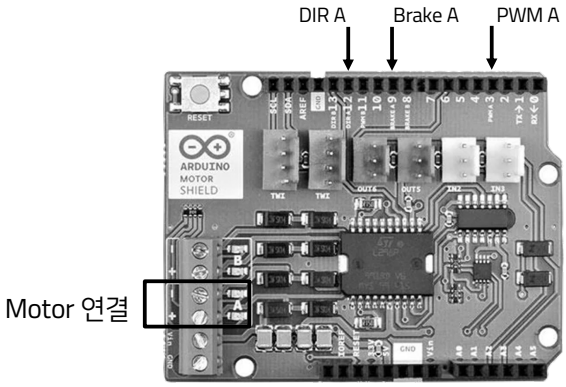
Jehong Jeon
Architecture and Compiler for Embedded system Lab.
School of Electronics Engineering, KNU, KOREA
2023-06-30

Motor Example

- PWM Duty Ratio에 따른 Motor Dimming
  1. 새로운 예제를 위한 프로젝트를 생성한다.
  2. 원하는 동작을 위해 레지스터와 메모리에 직접 접근해서 값을 써야한다.
  3. Board Schematic과 Datasheet를 통해 PWM 신호 출력에 대한 정보를 파악한다.
  4. PWM 신호 생성을 위해 사용할 GTM 모듈의 동작 원리를 파악하고 메모리 맵을 분석한다.
  5. 분석 결과를 활용해 임베디드 프로그래밍을 한다.

Motor Example

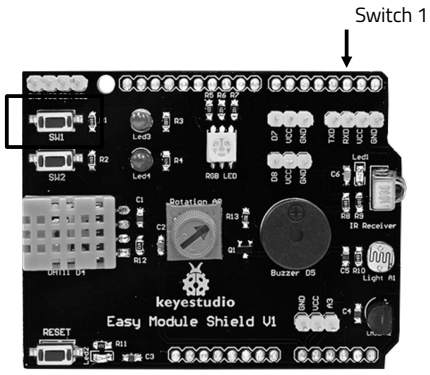
1. Motor 연결 정보 파악
   - ✓ Motor를 사용하기 위해 **Arduino Motor Shield**를 사용한다.



   - ✓ PWM A는 D3 단자에 연결되어 있으며, 모터 속도를 제어한다.
   - ✓ Brake A는 D9 단자에 연결되어 있으며, 모터의 동작 여부를 결정한다.
   - ✓ DIR A는 D12 단자에 연결되어 있으며, 모터의 동작 방향을 결정한다.

Motor Example

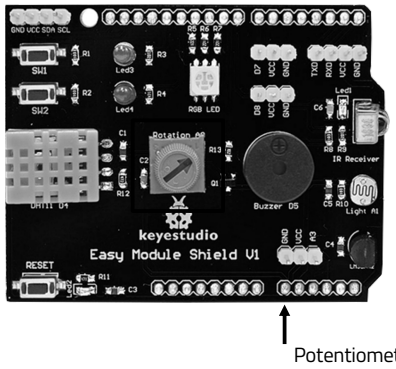
1. Switch 연결 정보 파악
   - ✓ Switch를 사용하기 위해 **Easy Module Shield V1** 확장 보드를 사용한다.



   - ✓ Switch 1은 D2핀에 연결되어 있다.
   - ✓ Switch 의 눌린 상태를 읽어서 모터의 방향을 결정한다.

Motor Example

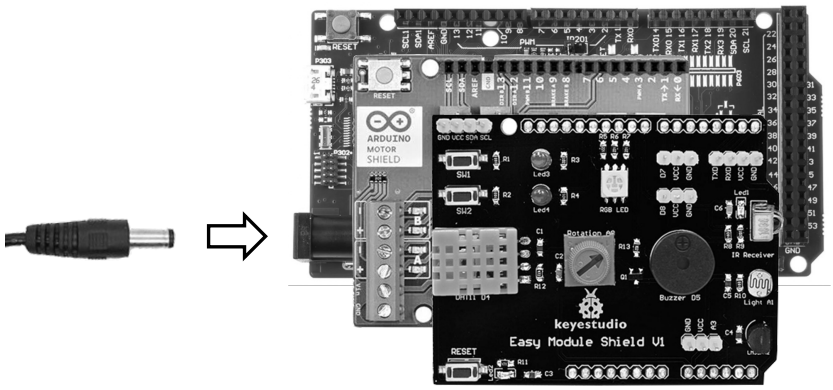
1. Potentiometer 연결 정보 파악
   - ✓ Potentiometer를 사용하기 위해 **Easy Module Shield V1** 확장 보드를 사용한다.



   - ✓ Potentiometer는 A0핀에 연결되어 있다.
   - ✓ Potentiometer의 전압 값을 ADC로 읽어서 Motor 의 속도를 조절한다.

Motor Example

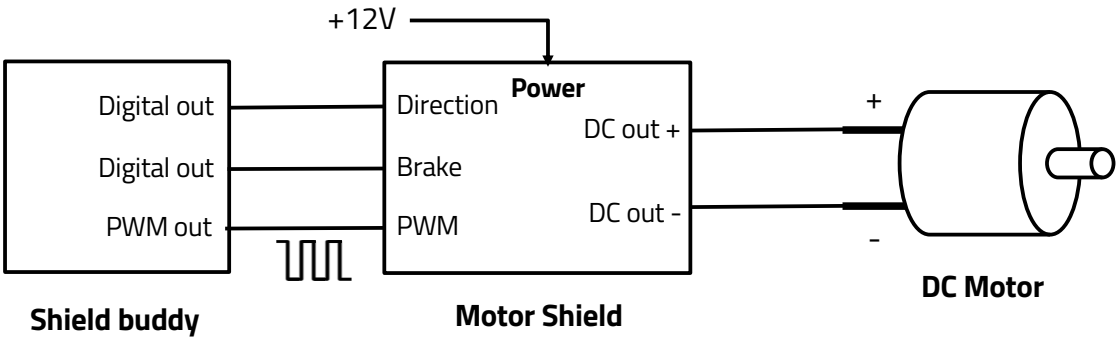
1. 확장 보드 연결 정보 파악
   - ✓ ShieldBuddy 보드 위에 **Arduino Motor Shield** 와 **Easy Module Shield V1** 확장 보드를 연결한 모습은 다음 그림과 같다.



   - ✓ 모터를 구동하기 위한 전원을 공급하기 위해 ShieldBuddy 보드에 12v 어댑터를 연결해야 한다.

Motor Example

1. 확장 보드 연결 정보 파악
   - ✓ **Arduino Motor Shield** 의 동작 구성은 다음과 같다.



   - ✓ Direction 은 모터의 회전 방향을 결정한다. High : 정방향, Low : 역방향
   - ✓ Brake 는 모터의 회전 가능 여부를 결정한다. High : 회전 중지 Low : 회전 가능
   - ✓ PWM 핀은 모터의 속도를 제어한다. 0% ~ 100% Duty 로 속도 조절

Motor Example

1. PWM 신호 출력 정보 파악
   - ✓ Motor PWM 핀이 연결된 PORT02 Pin 1는 GTM 모듈의 TOUT1과 연결되어 있다.
   - ✓ GTM 모듈의 TOUT1이 PWM 신호를 출력하면 PORT02 Pin 1을 통해 Motor Driver에 인가될 수 있다.
   - ✓ PWM 신호를 통해 Motor 속도를 제어하기 위해 해당 Pin을 **GTM 모듈의 TOUT1** 으로 설정해야 한다.

Table 13-14 Port 02 Functions (cont'd)

Port Pin	I/O	Pin Functionality	Associated Reg./ I/O Line	Port I/O Control Select.	
				Reg./Bit Field	Value
P02.1	I	General-purpose input	P02_IN.P1	P02_IOCR0. PC1	$0XXXX_B$
		GTM input	TIN1		
		ASCLIN2 input	ARX2B		
		CAN node 0 input	RXDCAN0A		
		ERAY input	RXDA2		
		CIF input	CIFD1		
		SCU input	REQ14		
	O	General-purpose output	P02_OUT.P1		$1X000_B$
		GTM output	TOUT1		$1X001_B$
		Reserved	–		$1X010_B$
		QSPI3 output	SLSO32		$1X011_B$
		DSADC output	DSCGPWMP		$1X100_B$
		Reserved	–		$1X101_B$
		Reserved	–		$1X110_B$
		CCU60 output	COUT60		$1X111_B$

datasheet: p.1151

Motor Example

2. Data sheet 분석 : IO 설정 (1)
   - ✓ PORT02 Pin 1을 GTM 모듈의 TOUT1으로 설정하기 위해 **P02_IOCR0 Register**의 **PC1 bits**를 **1001b**로 설정한다

P02_IOCR0 Register 주소: F003_A210h (F003A200h + 10h)

P02_IOCR0 Register 구조:

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_IOCR0
Port 02 Input/Output Control Register 0
(10_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		

datasheet: p.1080

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

1) This is the default pull device setting after reset for powertrain applications.

Motor Example

2. Data sheet 분석 : IO 설정 (2)
   - ✓ Motor DIR 핀의 Output을 사용하기 위해 연결된 Pin의 IO 설정이 필요하다.
   - ✓ DIR A 핀이 PORT10 Pin 1에 연결되어 있기 때문에 **P10_IOCR Register**의 **PC1 bits**를 설정한다.

P10_IOCR0 Register 주소: F003_B010h (F003B000h + 10h)

P10_IOCR0 Register 구조:

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

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		

datasheet: p.1080

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

1) This is the default pull device setting after reset for powertrain applications.

Motor Example

2. Data sheet 분석 : IO 설정 (3)

- ✓ Motor BRAKE 핀의 Output을 사용하기 위해 연결된 Pin의 IO 설정이 필요하다.
- ✓ BRAKE 핀이 PORT02 Pin 7에 연결되어 있기 때문에 **P02_IOCR Register**의 **PC7 bits**를 설정한다.

P02_IOCR4 Register 주소: F003_A214h (F003A200h + 14h)

P02_IOCR4 Register 구조:

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

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		

datasheet: p.1082

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

1) This is the default pull device setting after reset for powertrain applications.

Motor Example

2. Data sheet 분석 : IO 설정 (4)

- ✓ Switch 1 의 Input을 사용하기 위해 연결된 Pin의 IO 설정이 필요하다.
- ✓ Switch는 PORT02 Pin 0에 연결되어 있기 때문에 **P02_IOCR Register**의 **PC0 bits**를 설정한다.

P02_IOCR0 Register 주소: F003_A210h (F003A200h + 10h)

P02_IOCR0 Register 구조:

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_IOCR0
Port 02 Input/Output Control Register 0
(10_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		

datasheet: p.1080

Table 13-5 PCx Coding

PCx[4:0]	I/O	Characteristics	Selected Pull-up / Pull-down / Selected Output Function
$0XX00_B$	Input	–	No input pull device connected, tri-state mode
$0XX01_B$			Input pull-down device connected
$0XX10_B$			Input pull-up device connected[1)]
$0XX11_B$			No input pull device connected, tri-state mode

Motor Example

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

GTM_CLC Register 주소: F019_FD00h (F0100000h + 9FD00h)

GTM_CLC Register 구조:

Table 25-63 Registers Address Space

Module	Base Address	End Address	Note
GTM	$F010\ 0000_H$	$F019\ FFFF_H$	

CLC
Clock Control Register **($9FD00_H$)** **Reset Value: $0000\ 0003_H$**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0												EDIS		DIS S	DIS R
r												rw	r	r	rw

Field	Bits	Type	Description
DISR	0	rw	**Module Disable Request Bit** Used for enable/disable control of the GTM module. 0_B No disable requested 1_B Disable requested
DISS	1	r	**Module Disable Status Bit** Bit indicates the current status of the GTM module. 0_B GTM module is enabled 1_B GTM module is disabled
EDIS	3	rw	**Sleep Mode Enable Control** Used for module sleep mode control.
0	2, [31:4]	r	**Reserved** Read as 0; should be written with 0.

datasheet: p.3766

Motor Example

2. Data sheet 분석 : System Critical Register 설정 (1)
   - ✓ 설정해야 하는 GTM_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 7-5 Registers Address Spaces - CCU Registers

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

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

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PW														LCK	END INIT
rwh														rwh	rwh

Motor 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 Passward Access를 통해 Unlock되고, Modify Access를 통해 Lock 된다.
   - ✓ **PW bits**는 SCU_WDTCPU0CON0 Register에 접근하기 위한 Password를 저장하며 해당 값을 읽으면 bits[7:2]가 반전되어 읽힌다.

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.

Motor Example

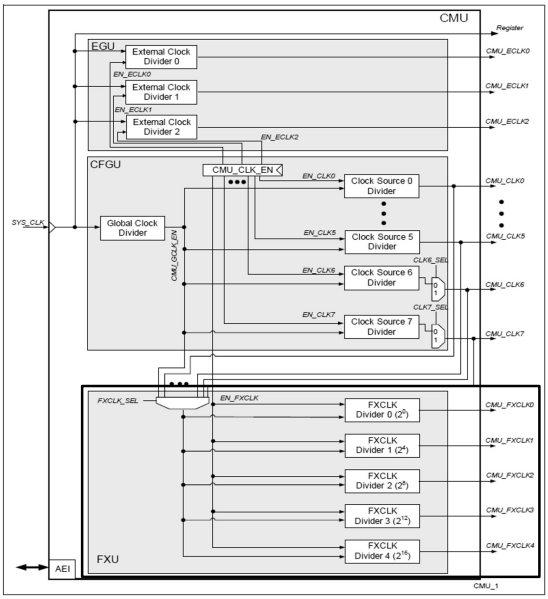
2. Data sheet 분석 : System Critical Register 설정 (3)
   - ✓ SCU_WDTCPU0CON0 Register에 적절한 값을 Write하여 **Password Access**를 수행한다.
   - ✓ **Password Access**는 **SCU_WDTCPU0CON0 Register의 Lock 상태를 해제**하며 과정은 다음과 같다.
     1. SCU_WDTCPU0CON0 Register의 값을 읽어 REL bits, PW bits를 얻는다.
     2. Bits[7:2] (PW bits의 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.

Motor 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 얻는다.
     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 해제되었는지 파악한다. (Modify Access가 정상적으로 수행되면 Lock 상태가 설정되며 LCK bit가 1로 설정된다.)
   - ✓ Modify Access를 통해 System ENDINIT을 해제하면 System Critical Register를 수정할 수 있으며 수정을 완료하면 System ENDINIT을 꼭 다시 설정해야 한다.

Motor Example

2. Data sheet 분석 : GTM 내부 Clock 설정 (1)
   - ✓ GTM 모듈은 내부에 CMU (Clock Management Unit)을 포함하고 있다.
   - ✓ CMU는 GTM 입력 클럭을 분주하여 다양한 내부 클럭을 생성하고, GTM 내부의 하위 모듈에 공급한다.
   - ✓ 본 실습에서 PWM 신호 생성을 위해 사용할 하위 모듈인 **TOM (Timer Output Module)**은 **CMU_FXCLK**에 따라 동작한다.
   - ✓ 따라서, CMU의 **FXU에 대한 설정**을 해야 한다.



Motor Example

2. Data sheet 분석 : GTM 내부 Clock 설정 (2)
   - ✓ GTM_CMU_FXCLK_CTRL Register는 CMU_FXCLK의 소스 클럭을 설정한다.
   - ✓ CMU_FXCLK의 소스 클럭으로 GTM 모듈의 입력 클럭인 CMU_GCLK_EN 또는 GTM 모듈 내부에서 생성된 CMU_CLKx가 사용될 수 있다.
   - ✓ 소스 클럭을 CMU_GCLK_EN으로 설정하기 위해 **FXCLK_SEL bits**를 **0000b**로 설정한다.

GTM_CMU_FXCLK_CTRL Register 주소: F010_0344h (F0100000h + 344h)

GTM_CMU_FXCLK_CTRL Register 구조:

Table 25-63 Registers Address Space

Module	Base Address	End Address	Note
GTM	$F010\ 0000_H$	$F019\ FFFF_H$	

GTM_CMU_FXCLK_CTRL
CMU FXCLK Control Register **(00344_H)** **Reset Value: 00000000_H**

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

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

datasheet: p.2844

Field	Bits	Type	Description
FXCLK_SEL	[3:0]	rw	**Input clock selection for EN_FXCLK line** 0000_BCMU_GCLK_EN selected 0001_BCMU_CLK0 selected 0010_BCMU_CLK1 selected 0011_BCMU_CLK2 selected 0100_BCMU_CLK3 selected 0101_BCMU_CLK4 selected 0110_BCMU_CLK5 selected 0111_BCMU_CLK6 selected 1000_BCMU_CLK7 selected *Note: This value can only be written, when the CMU_FXCLK generation is disabled. See bits 23…22 in register CMU_CLK_EN.* *Note: Other values for FXCLK_SEL are reserved and should not be used, but they behave like FXCLK_SEL = 0.*

Motor Example

2. Data sheet 분석 : GTM 내부 Clock 설정 (3)
   - ✓ GTM_CMU_CLK_EN Register는 CMU 내부의 클럭에 대한 Enable 설정을 한다.
   - ✓ GTM_CMU_CLK_EN Register는 CMU 내부에서 생성된 다양한 클럭에 대한 Enable을 설정할 수 있다.
   - ✓ CMU_FXCLK을 Enable 하기 위해 **EN_FXCLK bits**를 **10b**로 설정한다.

GTM_CMU_CLK_EN Register 주소: F010_0300h (F0100000h + 300h)

GTM_CMU_CLK_EN Register 구조:

Table 25-63 Registers Address Space

Module	Base Address	End Address	Note
GTM	$F010\ 0000_H$	$F019\ FFFF_H$	

GTM_CMU_CLK_EN
CMU Clock Enable Register **(00300_H)** **Reset Value: 00000000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								EN_FXCLK		EN_ECLK2		EN_ECLK1		EN_ECLK0	
r								rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EN_CLK7		EN_CLK6		EN_CLK5		EN_CLK4		EN_CLK3		EN_CLK2		EN_CLK1		EN_CLK0	
rw		rw		rw		rw		rw		rw		rw		rw	

datasheet: p.2835

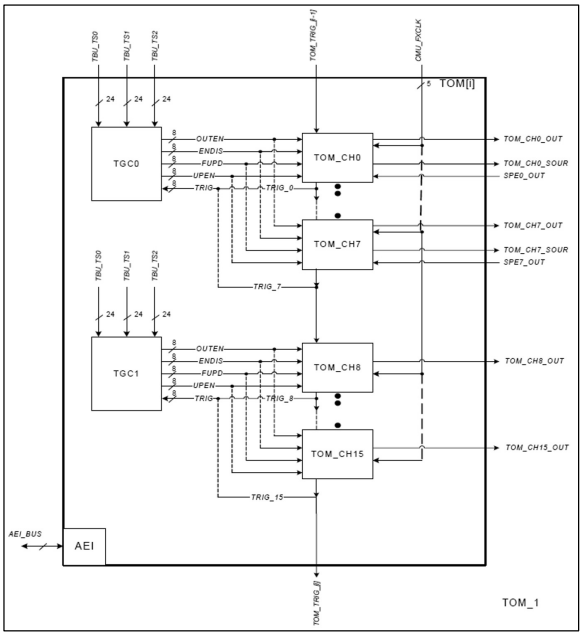
Field	Bits	Type	Description
EN_CLK4	[9:8]	rw	**Enable clock source 4** see bits [1:0]
EN_CLK5	[11:10]	rw	**Enable clock source 5** see bits [1:0]
EN_CLK6	[13:12]	rw	**Enable clock source 6** see bits [1:0]
EN_CLK7	[15:14]	rw	**Enable clock source 7** see bits [1:0]
EN_ECLK0	[17:16]	rw	**Enable ECLK 0 generation subunit** see bits [1:0]
EN_ECLK1	[19:18]	rw	**Enable ECLK 1 generation subunit** see bits [1:0]
EN_ECLK2	[21:20]	rw	**Enable ECLK 2 generation subunit** see bits [1:0]
EN_FXCLK	[23:22]	rw	**Enable all CM** see bits [1:0] *Note: An enable to EN_FXCLK from disable state will be reset internal fixed clock counters.*

00_B clock source is disabled (ignore write access)
01_B disable clock signal and reset internal states
10_B enable clock signal
11_B clock signal enabled (ignore write access)

Motor Example

2. Data sheet 분석 : TOM 구조 분석

- ✓ PWM 신호 생성을 위해 GTM 모듈 내부의 TOM을 사용한다.
- ✓ GTM 모듈은 3개의 TOM을 포함하고 있고, 각 TOM은 2개의 TGC (TOM Global Channel Control)와 16개의 TOM Channel을 가지고 있다.
- ✓ **TGC**는 8개의 TOM Channel과 연결되어 있으며 이를 통해 **TOM Channel을 제어**할 수 있다.
- ✓ **TOM Channel**은 TGC의 제어에 따라 동작을 수행하며 **출력 신호를 생성**한다.
- ✓ 본 실습에서는 **TOM0_CH9**를 사용한다.
- ✓ 따라서, **TOM0_CH9를 사용하기 위한 설정**을 수행한다.



Motor Example

2. Data sheet 분석 : TOM0 – TGC1 설정 (1)

- ✓ GTM_TOM0_TGC1_GLB_CTRL Register는 Channel 8-15를 제어하는 TGC1에 대한 설정을 한다.
- ✓ Channel에 대한 Enable/Disable 설정 및 Output Enable 설정은 트리거 신호에 의해 일괄적으로 반영된다.
- ✓ **HOST_TRIG bit**를 **1**로 설정하여 사용자가 소프트웨어적으로 트리거 신호를 발생시킬 수 있다.

GTM_TOM0_TGC1_GLB_CTRL Register 주소: F010_8230h (F0100000h + 8230h)

GTM_TOM0_TGC1_GLB_CTRL Register 구조:

Table 25-63 Registers Address Space

Module	Base Address	End Address	Note
GTM	$F010\ 0000_H$	$F019\ FFFF_H$	

GTM_TOMi_TGC1_GLB_CTRL (i=0-2)
TOMi TGC1 Global Control Register(08230_H+i*800_H) **Reset Value: 00000000_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UPEN_CTRL7		UPEN_CTRL6		UPEN_CTRL5		UPEN_CTRL4		UPEN_CTRL3		UPEN_CTRL2		UPEN_CTRL1		UPEN_CTRL0	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RST_CH7	RST_CH6	RST_CH5	RST_CH4	RST_CH3	RST_CH2	RST_CH1	RST_CH0	Reserved							HOST_TRIG
w	w	w	w	w	w	w	w	r							w

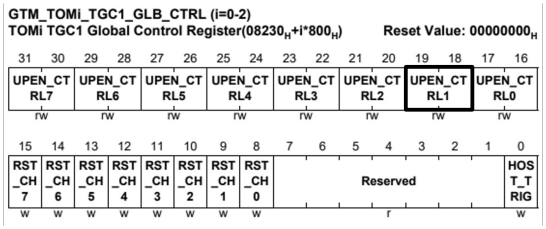
datasheet: p.2937

Field	Bits	Type	Description
HOST_TRIG	0	w	**Trigger request signal (see TGC0, TGC1) to update the register ENDIS_STAT and OUTEN_STAT** 0_B no trigger request 1_B set trigger request Read as 0. *Note:* This flag is cleared automatically after triggering the update
Reserved	[7:1]	r	**Reserved** Read as zero, should be written as zero
RST_CH0	8	w	**Software reset of channel 0** 0_B No action 1_B Reset channel Read as 0. *Note: This bit is cleared automatically after write by CPU. The channel registers are set to their reset values and channel operation is stopped immediately. The S-r FlipFlop SOUR is set to '1'.*

Motor Example

2. Data sheet 분석 : TOM0 – TGC1 설정 (2)
   - ✓ TOM 동작을 위한 CM0 / CM1 / CLK_SRC 값은 먼저 Shadow Register에 저장된다.
   - ✓ 업데이트가 Enable 되어 있으면 업데이트를 할 때 Shadow Register에 저장되어 있는 값이 일괄적으로 반영되어 CM0 / CM1 / CLK_SRC가 설정된다.
   - ✓ TOM Channel 1이 동작하기 위해서는 해당 Channel에 대한 CM0 / CM1 / CLK_SRC 값이 설정되어야 하며 이를 위해 **UPEN_CTRL1 bits**를 **10b**로 설정하여 업데이트를 Enable 한다.

GTM_TOM0_TGC1_GLB_CTRL Register 구조:



datasheet: p.2937

UPEN_CTRL0	[17:16]	rw	**TOM channel 0 enable update of register CM0, CM1 and CLK_SRC from SR0, SR1 and CLK_SRC_SR** Write of following double bit values is possible: 00_B don't care, bits 1:0 will not be changed 01_B update disabled: is read as 00 (see below) 10_B update enabled: is read as 11 (see below) 11_B don't care, bits 1:0 will not be changed Read of following double values means: 00_B channel disabled 11_B channel enabled
UPEN_CTRL1	[19:18]	rw	**TOM channel 1 enable update of register CM0, CM1 and CLK_SRC** See bits 17:16
UPEN_CTRL2	[21:20]	rw	**TOM channel 2 enable update of register CM0, CM1 and CLK_SRC** See bits 17:16

Motor Example

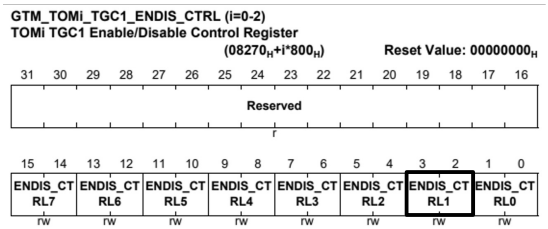
2. Data sheet 분석 : TOM0 – TGC1 설정 (3)
   - ✓ GTM_TOM0_TGC1_ENDIS_CTRL Register는 **트리거 신호에 따른 채널 Enable/Disable**을 설정한다.
   - ✓ 트리거 신호에 따라 각 Channel을 Enable 할지 Disable 할지 설정할 수 있다.
   - ✓ 트리거 신호 발생 시, Channel 9가 Enable 되게 **ENDIS_CTRL1 bits**를 **10b**로 설정한다.

GTM_TOM0_TGC1_ENDIS_CTRL Register 주소: F010_8270h (F0100000h + 8270h)

GTM_TOM0_TGC1_ENDIS_CTRL Register 구조:

Table 25-63 Registers Address Space

Module	Base Address	End Address	Note
GTM	$F010\ 0000_H$	$F019\ FFFF_H$	



datasheet: p.2940

Field	Bits	Type	Description
ENDIS_CTRL0	[1:0]	rw	**(A)TOM channel 0 enable/disable update value** If a TOM channel is disabled, the counter CN0 is stopped and the FlipFlop SOUR is set to the inverse value of control bit SL. On an enable event, the counter CN0 starts counting from its current value. Write of following double bit values is possible: 00_B don't care, bits 1:0 of register ENDIS_STAT will not be changed on an update trigger 01_B disable channel on an update trigger 10_B enable channel on an update trigger 11_B don't change bits 1:0 of this register Note: if the channel is disabled (ENDIS[0]=0) or the output is disabled (OUTEN[0]=0), the TOM channel 0 output TOM_OUT[0] is the inverted value of bit SL.
ENDIS_CTRL1	[3:2]	rw	**(A)TOM channel 1 enable/disable update value** See bits 1:0

Motor Example

2. Data sheet 분석 : TOM0 – TGC1 설정 (4)
   - ✓ GTM_TOM0_TGC1_OUTEN_CTRL Register는 **트리거 신호에 따른 Output Enable**을 설정한다.
   - ✓ 트리거 신호에 따라 각 Channel의 Output을 Enable 할지 Disable 할지 설정할 수 있다.
   - ✓ 트리거 신호 발생 시, Channel 9의 Output이 Enable 되게 **OUTEN_CTRL1 bits**를 **10b**로 설정한다.

GTM_TOM0_TGC1_OUTEN_CTRL Register 주소: F010_8278h (F0100000h + 8278h)

GTM_TOM0_TGC1_OUTEN_CTRL Register 구조:

Table 25-63 Registers Address Space

Module	Base Address	End Address	Note
GTM	$F010\ 0000_H$	$F019\ FFFF_H$	

GTM_TOMi_TGC1_OUTEN_CTRL (i=0-2)
TOMi TGC1 Output Enable Control Register
(08278_H+i*800_H) **Reset Value: 00000000_H**

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

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OUTEN_CTRL7		OUTEN_CTRL6		OUTEN_CTRL5		OUTEN_CTRL4		OUTEN_CTRL3		OUTEN_CTRL2		OUTEN_CTRL1		OUTEN_CTRL0	
rw		rw		rw		rw		rw		rw		rw		rw	

datasheet: p.2945

Field	Bits	Type	Description
OUTEN_CTRL0	[1:0]	rw	**Output (A)TOM_OUT(0) enable/disable update value** Write of following double bit values is possible: 00_B don't care, bits 1:0 of register OUTEN_STAT will not be changed on an update trigger 01_B disable channel output on an update trigger 10_B enable channel output on an update trigger 11_B don't change bits 1:0 of this register Note: if the channel is disabled (ENDIS[0]=0) or the output is disabled (OUTEN[0]=0), the TOM channel 0 output TOM_OUT[0] is the inverted value of bit SL.
OUTEN_CTRL1	[3:2]	rw	**Output (A)TOM_OUT(1)enable/disable update value** See bits 1:0
OUTEN_CTRL2	[5:4]	rw	**Output (A)TOM_OUT(2) enable/disable update value** See bits 1:0
OUTEN_CTRL3	[7:6]	rw	**Output (A)TOM_OUT(3) enable/disable update value** See bits 1:0
OUTEN_CTRL4	[9:8]	rw	**Output (A)TOM_OUT(4) enable/disable update value** See bits 1:0
OUTEN_CTRL5	[11:10]	rw	**Output (A)TOM_OUT(5) enable/disable update value** See bits 1:0

Motor Example

2. Data sheet 분석 : TOM0 – Channel 9 설정 (1)
   - ✓ GTM_TOM0_CHx_CTRL Register는 TOM1의 각 Channel에 대한 동작 설정을 한다.
   - ✓ TOM Channel 9의 동작을 설정하기 위해 **GTM_TOM0_CH9_CTRL Register**를 설정한다.
   - ✓ 출력 신호의 Duty Cycle에 대한 Signal level을 High로 설정하기 위해 **SL bit**를 **1**로 설정한다.

GTM_TOM0_CH9_CTRL Register 주소: F010_8240h (F0100000h + 8240h)

GTM_TOM0_CH9_CTRL Register 구조:

Table 25-63 Registers Address Space

Module	Base Address	End Address	Note
GTM	$F010\ 0000_H$	$F019\ FFFF_H$	

GTM_TOM0_CHx_CTRL (x=0-14)
TOM0 Channel x Control Register'
(08000_H+x*0040_H) **Reset Value: 00000800_H**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		GCM	SPEM	Reserved	OSM	Reserved	TRIGOUT	Reserved			RST_CCU0	Reserved			
r		rw	rw	r	rw	r	rw	r			rw	r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	CLK_SRC_SR			SL	Reserved										
r	rw			rw	r										

datasheet: p.2953

Field	Bits	Type	Description
SL	11	rw	**Signal level for duty cycle** 0_B Low signal level 1_B High signal level If the output is disabled, the output TOM_OUT[x] is set to inverse value of SL.

Motor Example

2. Data sheet 분석 : TOM0 – Channel 9 설정 (2)
   - ✓ TOM Channel 1의 동작 클럭을 CMU_FXCLK1로 설정하기 위해 **CLK_SRC_SR bits**를 **001b**로 설정한다.
   - ✓ CMU_FXCLK1의 주파수는 100MHz / 16 = 6,250kHz 이다.
   - ✓ CLK_SRC_SR bits가 업데이트를 할 때 반영되기 때문에 TOM Channel 1의 동작 클럭 또한 업데이트를 할 때 반영된다.

GTM_TOM0_CH9_CTRL Register 주소: F010_8240h (F0100000h + 8240h)

GTM_TOM0_CH9_CTRL Register 구조:

Table 25-63 Registers Address Space

Module	Base Address	End Address	Note
GTM	F010 0000$_H$	F019 FFFF$_H$	

GTM_TOM0_CHx_CTRL (x=0-14)
TOM0 Channel x Control Register'

(08000$_H$+x*0040$_H$) **Reset Value: 00000800$_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		GCM	SPEM	Reserved	OSM	Reserved	TRIGOUT	Reserved			RST_CCU0	Reserved			
r		rw	rw	r	rw	r	rw	r			rw	r			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	CLK_SRC_SR			SL	Reserved										
r	rw			rw	r										

datasheet: p.2953

CLK_SRC_SR	[14:12]	rw	**Clock source select for channel** The register CLK_SRC is updated with the value of CLK_SRC_SR together with the update of register CM0 and CM1. The input of the FX clock divider depends on the value of FXCLK_SEL (see CMU). 000_B CMU_FXCLK(0) selected: clock selected by FXCLKSEL 001_B CMU_FXCLK(1) selected: clock selected by FXCLKSEL/ 2^4 010_B CMU_FXCLK(2) selected: clock selected by FXCLKSEL / 2^8 011_B CMU_FXCLK(3) selected: clock selected by FXCLKSEL / 2^12 100_B CMU_FXCLK(4) selected: clock selected by FXCLKSEL / 2^16 101_B no CMU_FXCLK selected, clock of channel stopped 110_B no CMU_FXCLK selected, clock of channel stopped 111_B no CMU_FXCLK selected, clock of channel stopped Note: if clock of channel is stopped (i.e. CLK_SRC = 101/110/111), the channel can only be restarted by resetting CLK_SRC_SR to a value of 000 to 100 and forcing an update via the force update mechanism.

Motor Example

2. Data sheet 분석 : TOM1 – Channel 9 설정 (3)
   - ✓ GTM_TOM1_CHx_SR0 Register는 CM0에 대한 Shadow Register이다.
   - ✓ TOM Channel 9의 CM0를 설정하기 위해 **GTM_TOM1_CH9_SR0 Register**를 설정한다.
   - ✓ GTM_TOM1_CH9_SR0 Register에 설정할 CM0 값을 저장하면 업데이트를 할 때 CM0에 반영된다.
   - ✓ 본 실습에서는 PWM 신호의 주기를 2ms로 설정하기 위해 해당 Register의 값을 **(12500 – 1)**로 설정한다.

GTM_TOM0_CH1_SR0 Register 주소: F010_8244h (F0100000h + 8244h)

GTM_TOM0_CH1_SR0 Register 구조:

Table 25-63 Registers Address Space

Module	Base Address	End Address	Note
GTM	F010 0000$_H$	F019 FFFF$_H$	

GTM_TOM0_CHx_SR0 (x=0-15)
TOM0 Channel x CCU0 Compare Shadow Register

(08004$_H$+x*0040$_H$) **Reset Value: 00000000$_H$**

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
Reserved																SR0															
r																rw															

datasheet: p.2962

$$(Period\ of\ PWM) = \frac{(Value\ of\ CM0) + 1}{(Freq.of\ CMU\_FXCLK1)}$$

$$= \frac{12500}{6250kHz} = 0.002s$$

Motor Example

2. Data sheet 분석 : TOUT 설정 (1)
   - ✓ GTM 모듈 내 하위 모듈에서 생성한 출력 신호를 외부에 전달하기 위해서는 GTM 모듈의 출력 포트 (TOUT)와 연결 설정을 해야 한다.
   - ✓ 하나의 출력 포트에는 하위 모듈에서 생성된 출력 신호 4개가 MUX를 통해 연결되어 있으며 MUX 제어를 통해 하나의 신호가 출력 포트와 연결된다.
   - ✓ **GTM_TOUTSEL Register**는 MUX에 제어 신호를 입력하며 하나의 Register가 16개의 MUX를 제어한다.
   - ✓ 따라서, LED가 연결된 TOUT1 (PORT02 Pin 1)은 **GTM_TOUTSEL0 Register**의 **SEL1 bits**를 통해 설정할 수 있다.

Motor Example

2. Data sheet 분석 : TOUT 설정 (2)
   - ✓ GTM_TOUTSEL Register는 TOUT을 통해 출력될 신호를 설정한다.
   - ✓ TOUT1에 대해 설정하기 위해 **GTM_TOUTSEL0 Register**의 **SEL1 bits**를 설정한다.
   - ✓ TOM0 Channel 9를 통해 생성한 PWM 신호를 TOUT1로 출력하기 위해 **SEL1 bits**를 **00b**로 설정한다.

GTM_TOUTSEL0 Register 주소: F019_FD30h (F0100000h + 9FD30h)

GTM_TOUTSEL0 Register 구조:

Table 25-63 Registers Address Space

Module	Base Address	End Address	Note
GTM	F010 0000$_H$	F019 FFFF$_H$	

TOUTSELn (n = 0-14)
Timer Output Select Register (9FD30$_H$+n*4$_H$) Reset Value: 0000 0000$_H$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SEL15		SEL14		SEL13		SEL12		SEL11		SEL10		SEL9		SEL8	
rw		rw		rw		rw		rw		rw		rw		rw	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEL7		SEL6		SEL5		SEL4		SEL3		SEL2		SEL1		SEL0	
rw		rw		rw		rw		rw		rw		rw		rw	

Field	Bits	Type	Description
SELx (x = 0-15)	[x*2+1: x*2]	rw	**TOUT(n*16+x) Output Selection** This bit defines which timer out is connected as TOUT(n*16+x). The mapping for each pin is defined by **Table 25-67Table 25-68**. 00_B Timer A form **Table 25-67Table 25-68** is connected as TOUT(n*16+x) to the ports 01_B Timer B form **Table 25-67Table 25-68** is connected as TOUT(n*16+x) to the ports 10_B Timer C form **Table 25-67Table 25-68** is connected as TOUT(n*16+x) to the ports 11_B Timer D form **Table 25-67Table 25-68** is connected as TOUT(n*16+x) to the ports *Note: If TOUT(n*16+x) is not defined in **Table 25-67Table 25-68** this bit field has to be treated as reserved.*

datasheet: p.3541

Motor 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.
   - ✓ 해당 Register를 수정하기 위해서는 System ENDINIT을 해제해야 한다.

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

datasheet: p.3889

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**

Motor 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 AN00이 Group 4에 입력되기 때문에 **VADC_G4ARBPR Register**를 설정한다.
   - ✓ Pin AN00에 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	

datasheet: p.3945

Motor 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	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	

datasheet: p.3945

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**

ACE Lab.

Motor 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	RPTDIS
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	FLUSH	TREV	CLRV	0	0	0	0	0	ENTR	ENGT	
r	r	r	r	w	w	w	w	r	r	r	r	r	rw	rw	

datasheet: p.3914

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.

ACE Lab.

Motor 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	

datasheet: p.3914

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.

Motor Example

2. Data sheet 분석 : Group 설정 (5)
   - ✓ VADC_GxARBCFG Register는 Group의 AD Converter에 대한 설정을 한다.
   - ✓ Group 4를 사용하므로 **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
SAM PLE	BU SY	CAL S	CAL	0	0	SYN RUN	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	ARB M	0	ARBRND		0	0	ANONC	
r	r	r	r	r	r	r	r	rw	r	rw		r	r	rw	

datasheet: p.3942

- ANONS = 11_B: **Normal Operation**
  The converter is active, conversions are started immediately. Requires no wakeup time.

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**

Motor Example

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				

datasheet: p.3952

Motor Example

2. Data sheet 분석 : Group 설정 (7)
   - ✓ Conversion Mode를 12-bit Conversion으로 설정하기 위해 **CMS bits**를 **000b**로 설정한다.

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				

datasheet: p.3952

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**

Motor 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 입력되므로 **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	

datasheet: p.3949

Motor 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 4의 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	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	

datasheet: p.3949

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_BStore result in group result register GxRES0 ... 1111_BStore 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

Motor Example

2. Data sheet 분석 : Conversion Request 설정
   - ✓ VADC_GxQINR Register는 Request Source의 Conversion Request에 대한 설정을 한다.
   - ✓ Group 4의 Request Source 0을 사용하므로 **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. (해당 bit를 1b로 설정하면 Conversion이 끝난 후 다시 Conversion Request가 발생하여 Continuous Mode로 동작한다.)

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				

datasheet: p.3918

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)

Motor 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: $0000\ 0000_H$**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VF	FCR	CRS		EMUX			CHNR					DRC			
rh	rh	rh		rh			rh					rh			

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

datasheet: p.3972

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

Motor Example

3. 프로그래밍

1) 모터와 Potentiometer를 사용하기 위해 필요한 레지스터 주소와 비트 필드를 정의한다. (1)

```
#include "Ifx_Types.h"
#include "IfxCpu.h"
#include "IfxScuWdt.h"

/* Port Registers */
#define PORT02_BASE         (0xF003A200)
#define PORT02_OMR          (*(volatile unsigned int*) (PORT02_BASE + 0x04))
#define PORT02_IOCR0        (*(volatile unsigned int*) (PORT02_BASE + 0x10))
#define PORT02_IOCR4        (*(volatile unsigned int*) (PORT02_BASE + 0x14))
#define PORT02_IN           (*(volatile unsigned int*) (PORT02_BASE + 0x24))

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

#define P0                  0
#define PC0                 3
#define PC1                 11
#define PC7                 27
#define PS0                 0
#define PS1                 1
#define PS7                 7
#define PCL0                16
#define PCL1                17
#define PCL7                23
```

PORT 관련 레지스터 주소 및 비트 필드 정의

Motor Example

3. 프로그래밍

1) 모터와 Potentiometer를 사용하기 위해 필요한 레지스터 주소와 비트 필드를 정의한다. (2)

```
/* SCU Registers */
#define SCU_BASE            (0xF0036000)
#define SCU_WDTSCON0        (*(volatile unsigned int*)(SCU_BASE + 0x0F0))
#define SCU_WDT_CPU0CON0    (*(volatile unsigned int*)(SCU_BASE + 0x100))
#define SCU_EICR2           (*(volatile unsigned int*)(SCU_BASE + 0x218))
#define SCU_IGCR0           (*(volatile unsigned int*)(SCU_BASE + 0x22C))

#define LCK                 1
#define ENDINIT             0
#define INP0                12
#define EIEN0               11
#define REN0                9
#define FEN0                8
#define EXIS0               4
#define IGP0                14


/* SRC Registers */
#define SRC_BASE            (0xF0038000)
#define SRC_CCU60_SR0       (*(volatile unsigned int*)(SRC_BASE + 0x420))
#define SRC_SCUERU0         (*(volatile unsigned int*)(SRC_BASE + 0xCD4))

#define TOS                 11
#define SRE                 10
#define SRPN                0
```

SCU, SRC 관련 레지스터 주소 및 비트 필드 정의

```
/* CCU60 Registers */
#define CCU60_BASE          (0xF0002A00)
#define CCU60_CLC           (*(volatile unsigned int*)(CCU60_BASE + 0x00))
#define CCU60_T12           (*(volatile unsigned int*)(CCU60_BASE + 0x20))
#define CCU60_T12PR         (*(volatile unsigned int*)(CCU60_BASE + 0x24))
#define CCU60_TCTR0         (*(volatile unsigned int*)(CCU60_BASE + 0x70))
#define CCU60_TCTR4         (*(volatile unsigned int*)(CCU60_BASE + 0x78))
#define CCU60_INP           (*(volatile unsigned int*)(CCU60_BASE + 0xAC))
#define CCU60_IEN           (*(volatile unsigned int*)(CCU60_BASE + 0xB0))

#define DISS                1
#define DISR                0
#define CTM                 7
#define T12PRE              3
#define T12CLK              0
#define T12STR              6
#define T12RS               1
#define INPT12              10
#define ENT12PM             7
```

CCU60 타이머 관련 레지스터 주소 및 비트 필드 정의

Motor Example

3. 프로그래밍

1) 모터와 Potentiometer를 사용하기 위해 필요한 레지스터 주소와 비트 필드를 정의한다. (3)

```
/* 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 VADC_G4CHCTR7   (*(volatile unsigned int*)(VADC_BASE + 0x161C))
#define VADC_G4CHASS    (*(volatile unsigned int*)(VADC_BASE + 0x1488))

#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
#define ASSCH7          7
```

VADC 관련 레지스터 주소 및 비트 필드 정의

```
/* GTM Registers */
// GTM - CMU
#define GTM_BASE                    (0xF0100000)
#define GTM_CLC                     (*(volatile unsigned int*)(GTM_BASE + 0x9FD00))
#define GTM_TOUTSEL0                (*(volatile unsigned int*)(GTM_BASE + 0x9FD30))
#define GTM_CMU_CLK_EN              (*(volatile unsigned int*)(GTM_BASE + 0x00300))
#define GTM_CMU_FXCLK_CTRL          (*(volatile unsigned int*)(GTM_BASE + 0x00344))

#define EN_FXCLK                    22
#define FXCLK_SEL                   0
#define DISS                        1
#define DISR                        0
#define SEL1                        2

// GTM - TOM0
#define GTM_TOM0_TGC1_GLB_CTRL      (*(volatile unsigned int*)(GTM_BASE + 0x08230))
#define GTM_TOM0_TGC1_ENDIS_CTRL    (*(volatile unsigned int*)(GTM_BASE + 0x08270))
#define GTM_TOM0_TGC1_OUTEN_CTRL    (*(volatile unsigned int*)(GTM_BASE + 0x08278))
#define GTM_TOM0_TGC1_FUPD_CTRL     (*(volatile unsigned int*)(GTM_BASE + 0x08238))
#define GTM_TOM0_CH9_CTRL           (*(volatile unsigned int*)(GTM_BASE + 0x08240))
#define GTM_TOM0_CH9_SR0            (*(volatile unsigned int*)(GTM_BASE + 0x08244))
#define GTM_TOM0_CH9_SR1            (*(volatile unsigned int*)(GTM_BASE + 0x08248))

#define UPEN_CTRL1                  18
#define HOST_TRIG                   0
#define ENDIS_CTRL1                 2
#define OUTEN_CTRL1                 2
#define RSTCN0_CH1                  18
#define FUPD_CTRL1                  2
#define CLK_SRC_SR                  12
#define SL                          11
```

GTM 관련 레지스터 주소 및 비트 필드 정의

Motor Example

3. 프로그래밍

2) Switch와 Motor가 연결된 PORT의 IO 설정을 한다.

Switch 1	-	PORT02 Pin 0
Motor DIR_A	-	PORT10 Pin 1
Motor PWM_A	-	PORT02 Pin 1
Motor BRAKE_A	-	PORT02 Pin 7

Switch 와 Motor의 연결 PORT 정보

```
void initButton(void)
{
    PORT02_IOCR0 &= ~(0x1F << PC0);

    PORT02_IOCR0 |= (0x02 << PC0);
}
```

Switch 관련 IO 설정 코드

```
void initMotor(void)
{
    PORT10_IOCR0 &= ~(0x1F << PC1);
    PORT02_IOCR0 &= ~(0x1F << PC1);
    PORT02_IOCR4 &= ~(0x1F << PC7);

    PORT10_IOCR0 |= (0x10 << PC1);  // Set D12 to OUTPUT (DIRA)
    PORT02_IOCR0 |= (0x11 << PC1);  // Set D3 to PWM OUTPUT (PWMA)
    PORT02_IOCR4 |= (0x10 << PC7);  // Set D9 to OUTPUT (BRAKE_A)
}
```

Motor 관련 IO 설정 코드

Motor Example

3. 프로그래밍

2) VADC를 설정하기 위한 함수를 구현한다.

```
228⊖void initVADC(void)
229 {
230     /* VADC Enable */
231     /* Password Access to unlock WDTCPU0CON0 */
232     SCU_WDT_CPU0CON0 = ((SCU_WDT_CPU0CON0 ^ 0xFC) & ~(1 << LCK)) | (1 << ENDINIT);
233     while((SCU_WDT_CPU0CON0 & (1 << LCK)) != 0);
234
235     /* Modify Access to clear ENDINIT bit */
236     SCU_WDT_CPU0CON0 = ((SCU_WDT_CPU0CON0 ^ 0xFC) | (1 << LCK)) & ~ (1 << ENDINIT);
237     while((SCU_WDT_CPU0CON0 & (1 << LCK)) == 0);
238
239     VADC_CLC &= ~(1 << DISR);               // Enable VADC Module
240
241     /* Password Access to unlock WDTSCPU0CON0 */
242     SCU_WDT_CPU0CON0 = ((SCU_WDT_CPU0CON0 ^ 0xFC) & ~(1 << LCK)) | (1 << ENDINIT);
243     while((SCU_WDT_CPU0CON0 & (1 << LCK)) != 0);
244
245     /* Modify Access to clear ENDINIT bit */
246     SCU_WDT_CPU0CON0 = ((SCU_WDT_CPU0CON0 ^ 0xFC) | (1 << LCK)) | (1 << ENDINIT);
247     while((SCU_WDT_CPU0CON0 & (1 << LCK)) == 0);
248
249     while((VADC_CLC & (1 << DISS)) != 0);     // Wait until module is enabled
```

VADC System Critical Lock 해제 코드

```
252     VADC_G4ARBPR |=  ((0x3) << PRIO0);        // Highest Priority for Request Source 0
253     VADC_G4ARBPR &= ~((0x01) << CSM0);        // Conversion Start Mode : Wait-for-start mode
254     VADC_G4ARBPR |=  ((0x01) << (ASEN0));     // Arbitration Source Input 0 Enable
255
256     VADC_G4QMR0  &= ~((0x03) << ENGT);        // Enable Conversion Requests
257     VADC_G4QMR0  |=  ((0x01) << ENGT);
258
259     VADC_G4QMR0  |=  ((0x01) << FLUSH);       // Clear all Queue Entries
260
261     VADC_G4ARBCFG |=  ((0x3) << ANONC);       // Analog Converter : Normal Operation
262
263     VADC_G4ICLASS0 &= ~((0x7) << CMS);        // Group-specific Class 0
264
265     /* VADC Group 4 Channel 7 Setting */
266     VADC_G4CHCTR7 |=  ((0x01) << RESPOS);     // Read Results Right-aligned
267     VADC_G4CHCTR7 &= ~((0xF) << RESREG);      // Store Result in Group Result Register G4RES1
268     VADC_G4CHCTR7 &= ~((0x3) << ICLSEL);      // Use Group-specific Class 0
269
270     VADC_G4CHCTR7 |=  ((0x01) << RESREG);
271     VADC_G4CHASS |= 0x1 << ASSCH7;
272 }
```

VADC 설정 코드

Motor Example

3. 프로그래밍

2) VADC를 사용하기 위한 함수를 구현한다.

```
274⊖void VADC_startConversion(void)
275 {
276     /* No fill and Start Queue */
277     VADC_G4QINR0 &= ~(0x1F);                  // Request Channel Number : 6
278     VADC_G4QINR0 |=  (0x07);
279
280     VADC_G4QINR0 &= ~((0x01) << RF);          // No fill : it is converted once
281
282     VADC_G4QMR0 |=   ((0x01) << TREV);        // Generate a Trigger Event
283 }
```

ADC 변환을 시작하는 코드

```
285⊖unsigned int VADC_readResult(void)
286 {
287     unsigned int result;
288
289     while( (VADC_G4RES1 & (0x1 << VF)) == 0);
290     result = VADC_G4RES1 & ((0xFFFF) << RESULT);
291
292     return result;
293 }
```

ADC 변환이 완료되었을 때 값을 반환하는 코드

Motor Example

3. 프로그래밍

2) PWM을 설정하기 위한 GTM 설정 함수를 구현한다.

```
295⊖void initGTM(void)
296 {
297     /* Password Access to unlock WDTSCON0 */
298     SCU_WDT_CPU0CON0 = ((SCU_WDT_CPU0CON0 ^ 0xFC) & ~(1 << LCK)) | (1 << ENDINIT);
299     while((SCU_WDT_CPU0CON0 & (1 << LCK)) != 0);
300
301
302     /* Modify Access to clear ENDINIT bit */
303     SCU_WDT_CPU0CON0 = ((SCU_WDT_CPU0CON0 ^ 0xFC) | (1 << LCK)) & ~ (1 << ENDINIT);
304     while((SCU_WDT_CPU0CON0 & (1 << LCK)) == 0);
305
306     GTM_CLC &= ~(1 << DISR); // enable VADC
307
308     /* Password Access to unlock WDTSCON0 */
309     SCU_WDT_CPU0CON0 = ((SCU_WDT_CPU0CON0 ^ 0xFC) & ~(1 << LCK)) | (1 << ENDINIT);
310     while((SCU_WDT_CPU0CON0 & (1 << LCK)) != 0);
311
312     /* Modify Access to clear ENDINIT bit */
313     SCU_WDT_CPU0CON0 = ((SCU_WDT_CPU0CON0 ^ 0xFC) | (1 << LCK)) | (1 << ENDINIT);
314     while((SCU_WDT_CPU0CON0 & (1 << LCK)) == 0);
315
316     while((GTM_CLC & (1 << DISS)) != 0);       // Wait until module is enabled
```

GTM(PWM) System Critical Lock 해제 코드

```
319     GTM_CMU_FXCLK_CTRL &= ~((0x0F) << FXCLK_SEL);
320     GTM_CMU_CLK_EN |= ((0x02) << EN_FXCLK);
321
322     GTM_TOM0_TGC1_GLB_CTRL |= ((0x02) << UPEN_CTRL1);
323     GTM_TOM0_TGC1_ENDIS_CTRL |= ((0x02) << ENDIS_CTRL1);
324     GTM_TOM0_TGC1_OUTEN_CTRL |= ((0x02) << OUTEN_CTRL1);
325
326     GTM_TOM0_CH9_CTRL |= ((0x01) << SL);
327     GTM_TOM0_CH9_CTRL |= ((0x01) << CLK_SRC_SR);
328
329     GTM_TOM0_CH9_SR0 = 12500 - 1;
330
331     GTM_TOUTSEL0 &= ~((0x03) << SEL1);
332 }
```

GTM(PWM) 설정 코드

Motor Example

3. 프로그래밍

3) 동작에 따라 'main' 함수를 구현한다.

```
177     while(1)
178     {
179         VADC_startConversion();
180         unsigned int adcResult = VADC_readResult();
181
182         duty = 12500 * adcResult / 4096;
183
184         if( (PORT02_IN & (0x1 << P0)) == 0) {
185             // Run MOTOR Foward
186             // to run motor Foward :
187             // PWMA (D3) = HIGH
188             // DIRA (D12) = HIGH
189             // BRAKE_A (D9) = LOW
190
191             PORT10_OMR |= ((0x01) << PS1);      // Set DIRA to HIGH
192             PORT02_OMR |= ((0x01) << PCL7);     // Set BRAKE_A to LOW
193
194         } else {
195             // Run MOTOR Backward
196             // to run motor Foward :
197             // PWMA (D3) = HIGH
198             // DIRA (D12) = LOW
199             // BRAKE_A (D9) = LOW
200
201             PORT10_OMR |= ((0x01) << PCL1);     // Set DIRA to LOW
202             PORT02_OMR |= ((0x01) << PCL7);     // Set BRAKE_A to LOW
203         }
204
205         GTM_TOM0_CH9_SR1 = duty;    // Set PWMA
```

ADC 값을 읽어 모터의 속도가 변화하고, Switch 1 이 눌린 상태에 따라 모터가 정회전하거나 역회전 한다.

Switch 1이 눌린 상태라면 모터를 정방향으로 구동

Switch 1이 떨어진 상태라면 모터를 역방향으로 구동

PWM duty를 ADC 결과값으로 설정하여 모터의 속도를 변경한다.

Switch Example

4. 동작 확인
   - ✓ Build 및 Debug 후 ('Resume' 버튼 클릭), Switch와 Potentiometer의 위치에 따라 모터의 동작이 변하는 것을 확인한다.

Q & A

Thank you for your attention

Architecture and
Compiler
for Embedded Systems Lab.

School of Electronics Engineering, KNU
ACE Lab (jehongjeon27@gmail.com)