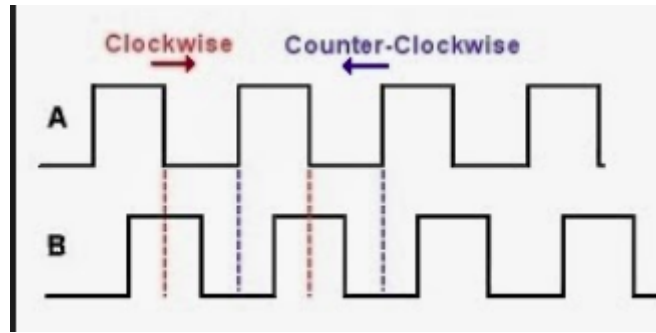


## <eQEP>

### - 엔코더 신호



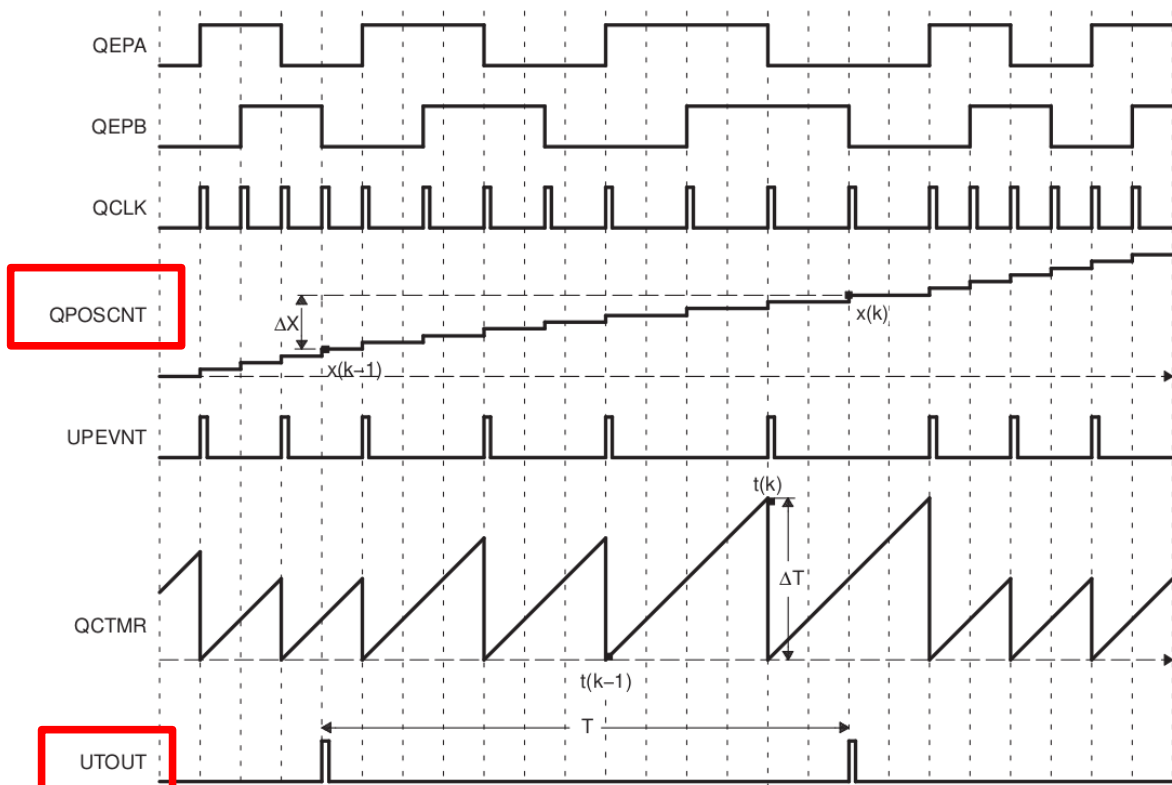
A와 B의 파형을 이용해서 회전 방향, 속도를 알 수 있음

- 방향 : A가 B보다 앞선다면 CW, B가 A보다 앞선다면 CCW 이런 식 임
- 속도: 파형의 길이로 알 수 있음.  
길이가 짧으면 속도가 빠름, 길이가 길면 속도가 느림.

### \* MCU에서 각속도를 알아내는 방법 \*

- 각속도 = 각도 / 시간
- 위의 식을 참고하여서 각속도를 계산함.

Figure 34-17. eQEP Edge Capture Unit - Timing Details



Velocity Calculation Equations:

$$v(k) = \frac{x(k) - x(k-1)}{T} = \frac{\Delta X}{T}$$

$v(k)$ : Velocity at time instant  $k$

$x(k)$ : Position at time instant  $k$

$x(k-1)$ : Position at time instant  $k-1$

$T$ : Fixed unit time or inverse of velocity calculation rate

$\Delta X$ : Incremental position movement in unit time

1) 각도 → QPOSLAT \* 1펄스 당 각

- 1펄스 (QEPA 나 QEPB에서 들어오는 펄스) 당 각도는 엔코더의 분해능에 의해 계산함

→ 우리가 사용하는 엔코더의 분해능은 500 ( 1바퀴 당 , 360도 당 펄스가 500개 나온다는 이야기)

따라서, 1펄스에 움직이는 각도는 360/500



E40H10-500-3-T-24



Autonics

분해능

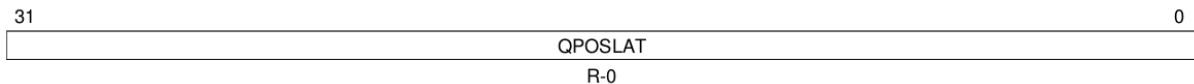
- QPOSLAT 값은 unit time out event 발생 시 QPOSCNT를 캡처함.

www.ti.com

eQEP Registers

### 34.3.7 eQEP Position Counter Latch Register (QPOSLAT)

Figure 34-27. eQEP Position Counter Latch Register (QPOSLAT) [offset = 18h]



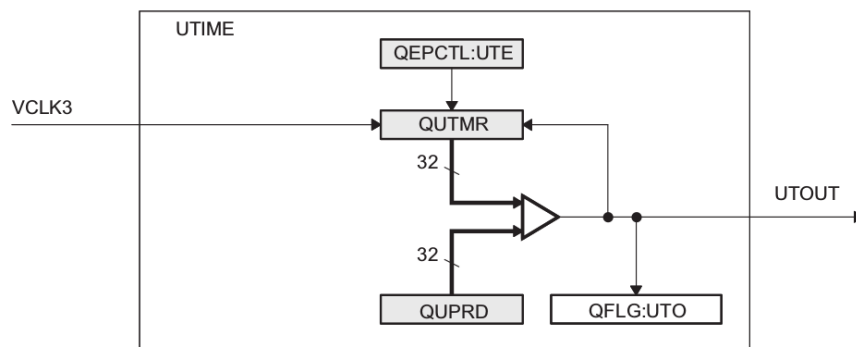
LEGEND: R = Read only; -n = value after reset

Table 34-10. eQEP Position Counter Latch Register (QPOSLAT) Field Descriptions

Bits	Name	Description
31-0	QPOSLAT	The position-counter value is latched into this register on unit time out event.

2) 시간 T → QUPRD에서 설정한 값

Figure 34-19. eQEP Unit Time Base



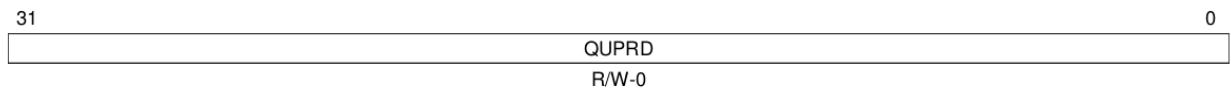
- 위 34-19 그림에서 VCLK3을 기준으로 QUTMR이 QUPRD 값과 일치하면 QFLG:UTO 비트가 set 되어 UTOUT(Unit Time Out interrupt)가 발생함

→ QUPRD를 일정한 시간으로 맞추면, 그 시간마다 인터럽트가 발생함

=> 즉, QUPRD에서 정한 시간 만큼,  $(360 / \text{엔코더의 분해능}) * \text{QPOSLAT}$  만큼의 각도로 이동했다

### 34.3.9 eQEP Unit Period Register (QUPRD)

Figure 34-29. eQEP Unit Period Register (QUPRD) [offset = 20h]



LEGEND: R/W = Read/Write; -n = value after reset

Table 34-12. eQEP Unit Period Register (QUPRD) Field Descriptions

Bits	Name	Description
31-0	QUPRD	This register contains the period count for unit timer to generate periodic unit time events to latch the eQEP position information at periodic interval and optionally to generate interrupt.

## 1. HalCoGen 설정

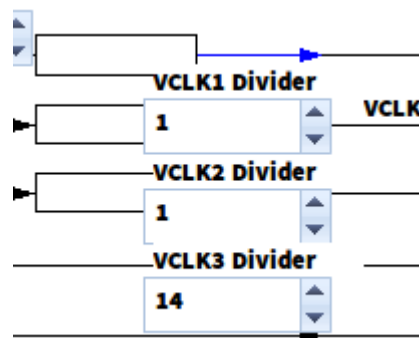
### 1) Driver Enable

- ☒ **Enable SCI drivers**
  - ☒ **Enable SCI3 driver \*\***
  - ☐ **Enable SCI4 driver \*\***
- ☐ **Enable LIN drivers**
  - ☐ **Enable LIN1 driver \*\*** / ☒ **Enable SCI1 driver \*\***
  - ☐ **Enable LIN2 driver \*\*** / ☐ **Enable SCI2 driver \*\***
- ☒ **Enable EQEP driver**
  - ☐ **Enable EQEP1 driver \*\***
  - ☒ **Enable EQEP2 driver \*\***
- ☒ **Enable ETPWM driver**

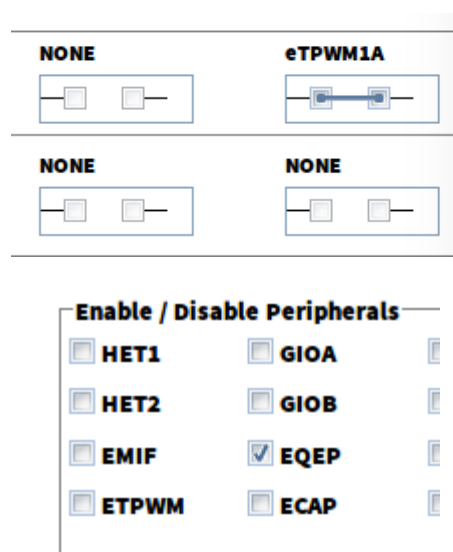
- SCI1 → UART

- SCI3 → BLUETOOTH

### 2) GCM 설정



### 3) PINMUX 설정



- EQEP2 사용

4) Special Pin Muxing → Enable TBCLK sync 활성화

☐ Use HET1\_LOOP\_SYNC for time-base sync

☒ Enable TBCLK sync\*\*

nTZ1 ASYNC ▼

nTZ2 ASYNC ▼

nTZ3 ASYNC ▼

5) ETPWM 설정

General ETPWM1 ETPWM2 ETPWM3 ETPWM4

Enable ETPWM modules

☒ Enable ETPWM1

☐ Enable ETPWM2

☐ Enable ETPWM3

**Clock configuration**

TB Clock (MHz): 110.000

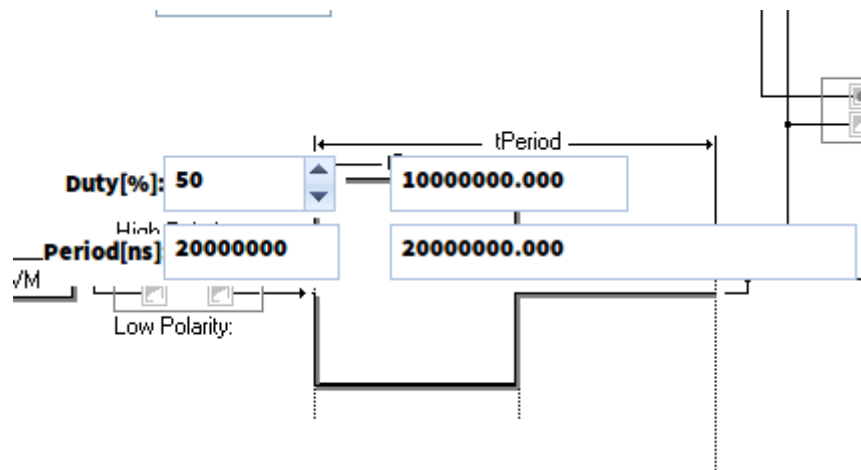
VCLK3 (MHz): 10.000

ActualTB Clock (MHz): 1.000

HSPCLKDIV: 10

CLKDIV: 0

Clock Prescale



## 6) EQEP 설정

**General Configuration**

Position Counter Mode: **QUADRATURE\_COUNT**

- QUADRATURE\_COUNT : QEPA 한 펄스에 4개의 QCLK 발생한다는 의미

**Interrupt Configuration**

<input type="checkbox"/> Position counter error Interrupt	<input type="checkbox"/> Position-compare ready Interrupt
<input type="checkbox"/> Quadrature phase error Interrupt	<input type="checkbox"/> Position-compare match Interrupt
<input type="checkbox"/> Quadrature direction change Interrupt	<input type="checkbox"/> Strobe event latch Interrupt
<input type="checkbox"/> Watchdog time out Interrupt	<input type="checkbox"/> Index event latch Interrupt
<input type="checkbox"/> Position counter underflow Interrupt	<input checked="" type="checkbox"/> Unit time out interrupt
<input type="checkbox"/> Position counter overflow Interrupt	

- Unit time out interrupt 사용

**Position Counter Configuration**

Counter Init Index Event: <b>RISING_EDGE</b>	<b>Max Position Count: 0xFFFFFFFF</b>
Counter Init Strobe Event: <b>DIRECTON_DEPENDENT</b>	<input type="checkbox"/> Init Counter on Index Event
<b>Position Counter Reset On: UNITTIME_EVENT</b>	<input type="checkbox"/> Init Counter on Strobe Event
Counter Latch Index Event: <b>RISING_EDGE</b>	<input type="checkbox"/> Enable SW Initialization
Counter Latch Strobe: <b>RISING_EDGE</b>	Init Position Count <b>0x00000000</b>

- unit timeout event 발생 시 capture 될 수 있도록 설정

**Capture Configuration**

<b>Capture Timer Prescaler:</b>	PS_8	<input type="checkbox"/> Init Counter on Strobe Event
<b>Unit Pos Event Prescaler:</b>	PS_1	<b>Unit Init Period:</b> 0x00000000
<b>Cap Timer Pos Mode:</b>	ON_UNIT_TIMEOUT_EVENT	

## 2. CCS 코드

```
50 #include "HL_sys_common.h"
51
52 /* USER CODE BEGIN (1) */
53 #include "HL_etpwm.h"
54 #include "HL_eqep.h"
55 #include "HL_sci.h"
56 #include <stdio.h>
57 #include <string.h>
58 /* USER CODE END */
59
60 #define UnitPeriod 1000000
61
62 char buf[128] = {0};
63 uint8 receive_data[5]={0};
64 unsigned int buf_len;
65 float value = 0.0;
66 float Kp = 0.0;
67 int flag = 0;
68
69 uint32 set_p = 0;
70
71 void wait(int);
72 void sci_display(sciBASE_t * sci, uint8 * text, uint32 len);
73 void catch_command(sciBASE_t * sci);
74 /* USER CODE END */
```

- SCI, etPWM, eQEP init

```
85 int main(void)
86 {
87     /* USER CODE BEGIN (3) */
88
89     uint32 deltapos = 0U;
90     uint32 deltaT = 0U;
91     float velocity = 0U;
92
93     sciInit();
94     wait(1000);
95
96     //sciREG1 : UART
97     sprintf(buf, "SCI Init Success!!!\n\r\0");
98     buf_len = strlen(buf);
99     sci_display(sciREG1, (uint8 *)buf, buf_len);
100    wait(100);
101
102    etpwmInit();
103    wait(1000);
104
105    sprintf(buf, "PWM Init Success!!!\n\r\0");
106    buf_len = strlen(buf);
107    sci_display(sciREG1, (uint8 *)buf, buf_len);
108    wait(100);
109
110    QEPInit();
111    wait(1000);
112
113    //Unit Period Setting (QUPRD 설정)
114    eqepSetUnitPeriod(eqepREG2, UnitPeriod);
115    //Counter 활성화
116    eqepEnableCounter(eqepREG2);
117    //Unit Timer 활성화
118    eqepEnableUnitTimer(eqepREG2);
119    //capture 활성화
120    eqepEnableCapture(eqepREG2);
121
122    sprintf(buf, "QEP Init Success!!!\n\r\0");
123    buf_len = strlen(buf);
124    sci_display(sciREG1, (uint8 *)buf, buf_len);
125    wait(100);
```



```

144     for(;;)
145     {
146         //위에서 QFLG:UTO비트(12번)가 set 되었는지 확인
147         if((eqepREG2->QFLG & 0x800U) == 0x800U)
148         {
149             // QEPSTS의 0x20은 direction을 나타냄
150             //현재의 방향을 shift해서 flag에 저장, 1이면 정방향, 0이면 역방향
151             flag = (eqepREG2->QEPSTS & 0x20) >> 5;
152             //0010 0000
153
154             sprintf(buf, "direction : %d\n\r\0", flag);
155             buf_len = strlen(buf);
156             sci_display(sciREG1, (uint8 *)buf, buf_len);
157             wait(100);
158
159             //deltaT : QUPRD값을 저장하는 변수
160             deltaT = 0;
161
162             //flag가 1이면 정방
163             if(flag == 1)
164             {
165                 //deltapos에 QPOSLAT 값을 읽음
166                 deltapos = eqepReadPosnLatch(eqepREG2);
167
168                 sprintf(buf, "delta : %d\n\r\0", deltapos);
169                 buf_len = strlen(buf);
170                 sci_display(sciREG1, (uint8 *)buf, buf_len);
171                 wait(100);
172             }
173             else
174             { //0이면 역방향
175                 deltapos = eqepReadPosnLatch(eqepREG2);
176                 deltapos = ~deltapos + 1;
177
178                 sprintf(buf, "delta : %d\n\r\0", deltapos);
179                 buf_len = strlen(buf);
180                 sci_display(sciREG1, (uint8 *)buf, buf_len);
181                 wait(100);
182             }
183
184             deltaT = eqepREG2->QUPRD;
185
186             sprintf(buf, "QUPRD : %d\n\r\0", deltaT);
187             buf_len = strlen(buf);
188             sci_display(sciREG1, (uint8 *)buf, buf_len);
189             wait(100);
190
191

```

```

193 //각속도 계산 = 각도 / 시간
194 // 시간 = deltaT(QUPRD) * (VCLK3 주기)
195 // VCLK3 = 10MHz이므로 주기는 1/10000000
196 // 각도 = (360/500) * deltapos * 0.25
197 // 0.25를 곱하는 이유는 우리가 사용하는 QCLK를 QUADRATURE_COUNT로 설정했기 때문
198 t_time = (float)deltaT * 0.0000001;
199 t_angle = (360.0 / 500.0) * (float)deltapos * 0.25;
200 velocity = t_angle / t_time;
201
202 sprintf(buf, "velocity : %f\n\r\0", velocity);
203 buf_len = strlen(buf);
204 sci_display(sciREG1, (uint8 *)buf, buf_len);
205 wait(100);
206
207 //위의 velocity는 1초당 이동 각도
208 //rpm은 1분당 rotation이므로 60 / 360 을 곱해줌
209 sprintf(buf, "rpm : %f\n\r\0", velocity / 6.0);
210 buf_len = strlen(buf);
211 sci_display(sciREG1, (uint8 *)buf, buf_len);
212 wait(100);
213
214 //interrupt flag를 clear함
215 eqepClearInterruptFlag(eqepREG2, QEINT_Uto);
216 }
217 }
218
219 /* USER CODE END */
220
221 return 0;
222 }
223
224 /* USER CODE BEGIN (4) */
225 void sci_display(sciBASE_t * sci, uint8 * text, uint32 len)
226 {
227     while(len--)
228     {
229         while((sci->FLR & 0x4) == 4) //SCI receive in idle state
230             ; //0 : ready to receive
231         sciSendByte(sci, *text++); //1 : not receive any data.
232     }
233 }
234
235 void wait(int delay)
236 {
237     int i;
238     for(i=0; i<delay; i++)
239         ;
240 }

```