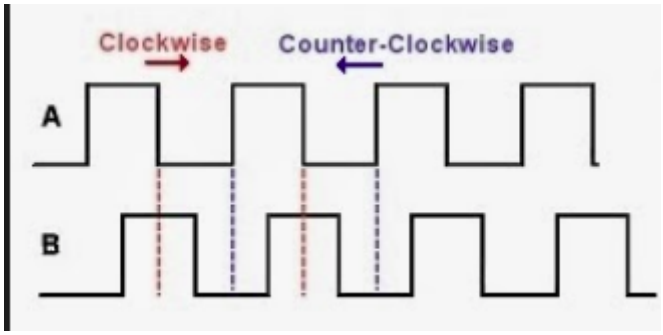
<eQEP>

- 엔코더 신호



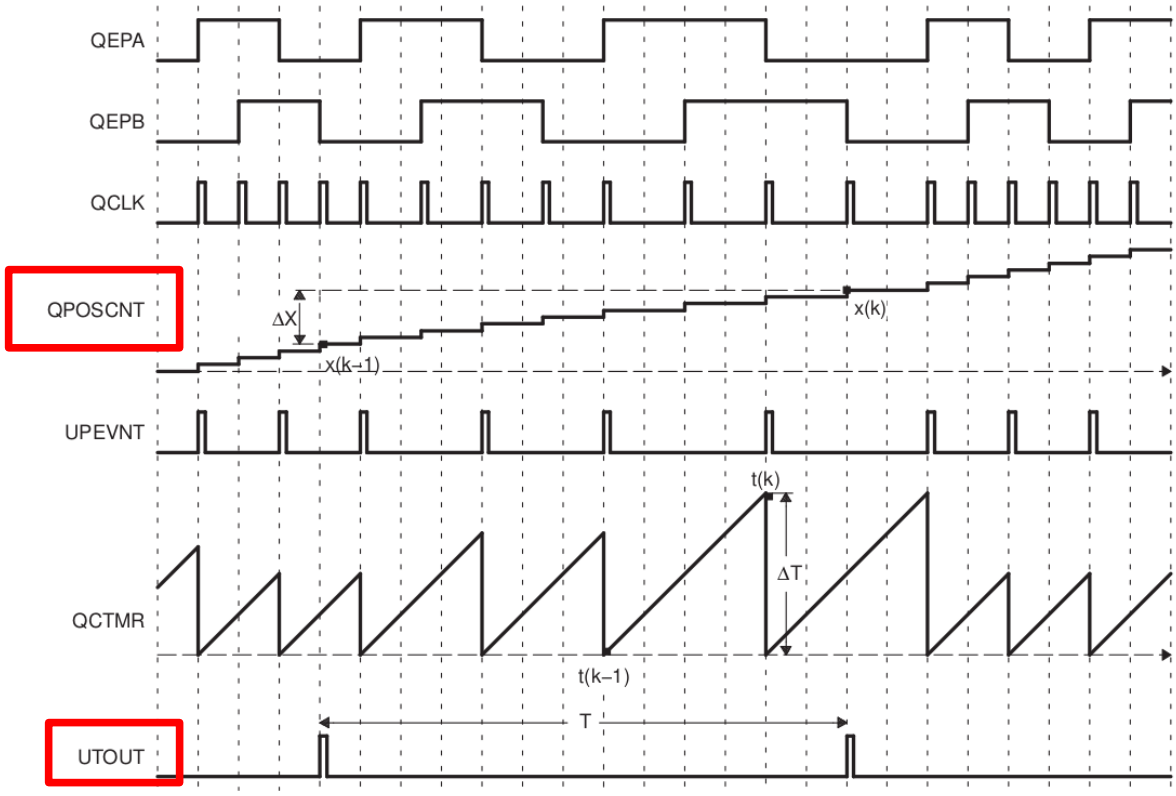
A와 B의 파형을 이용해서 회전 방향, 속도를 알 수 있음
- 방향 : A가 B보다 앞선다면 CW, B가 A보다 앞선다면 CCW 이런 식 임
- 속도: 파형의 길이로 알 수 있음.
길이가 짧으면 속도가 빠름, 길이가 길면 속도가 느림.

* MCU에서 각속도를 알아내는 방법 *

- 각속도 = 각도 / 시간

- 위의 식을 참고하여 서 각속도를 계산함.

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



Velocity Calculation Equations:

$$v(k) = \frac{x(k) - x(k-1)}{T} = \frac{\Delta X}{T} \quad (71)$$

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
ΔX: 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

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

34.3.7 eQEP Position Counter Latch Register (QPOSLAT)

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

31	0
QPOSLAT	
R-0	

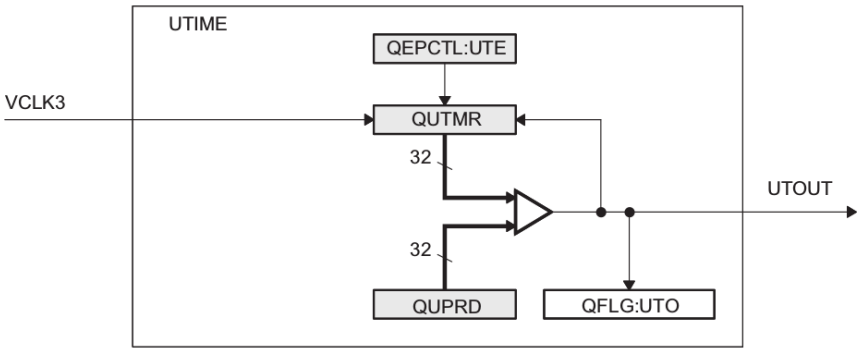
LEGEND: R = Read only; -*n* = value after reset

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 / 엔코더의 분해능) * QPOSLAT 만큼의 각도로 이동했다

34.3.9 eQEP Unit Period Register (QUPRD)

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

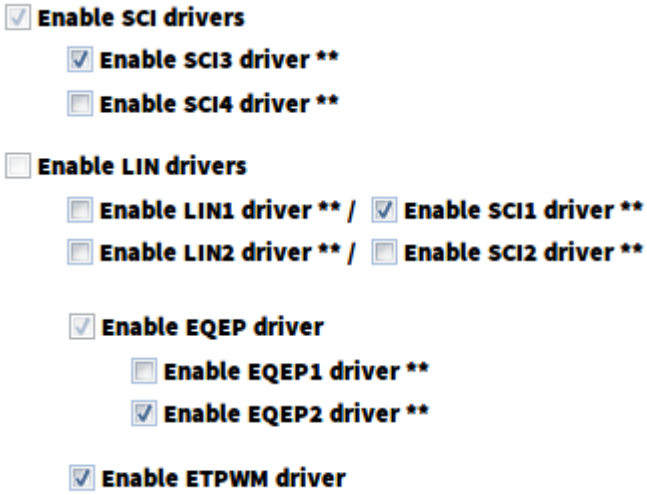
31	0
QUPRD	
R/W-0	

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

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

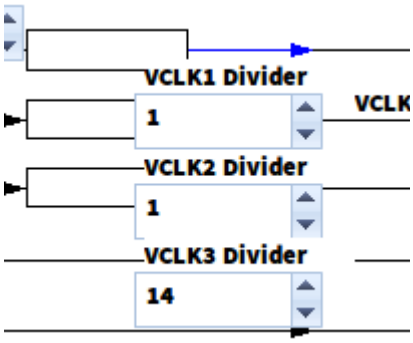
Bits	Name	Description
31-0	QUPRD	This register contains the period count for unit timer to generate periodic unit time events to latch the eQEP position information at periodic interval and optionally to generate interrupt.

1. HalCoGen 설정
1) Driver Enable

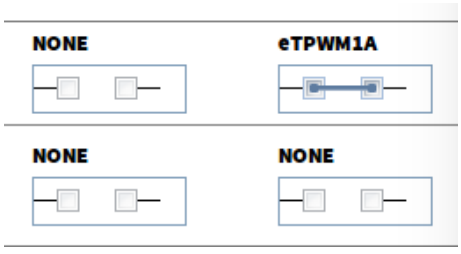


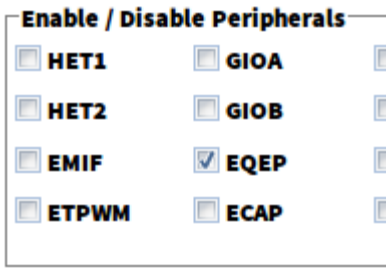
- SCI1 → UART
- SCI3 → BLUETOOTH

2) GCM 설정



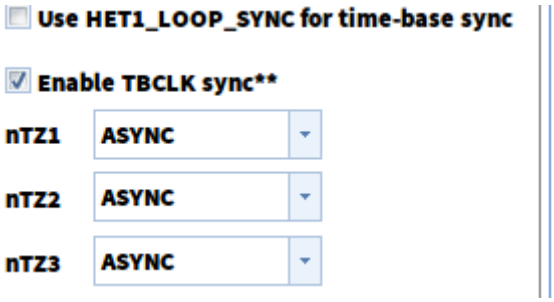
3) PINMUX 설정



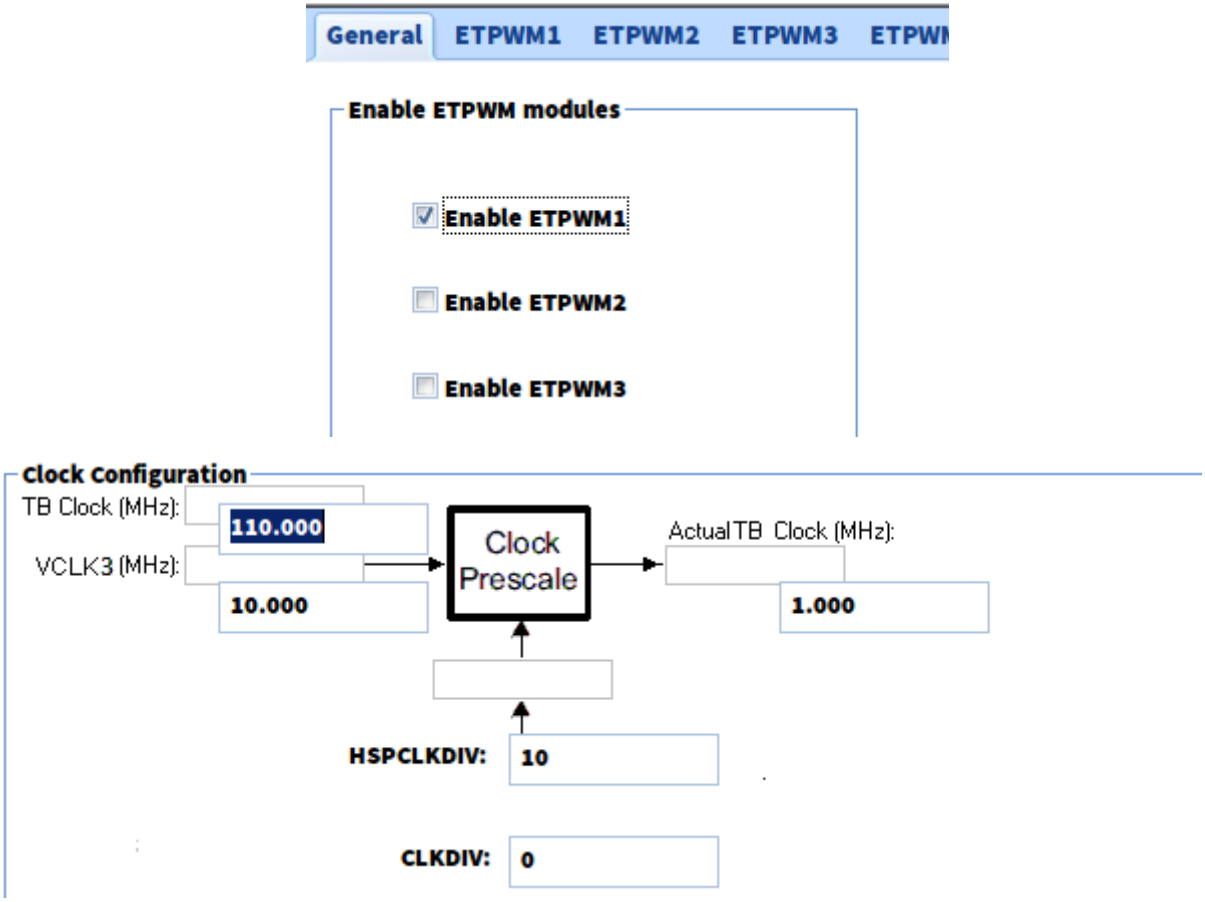


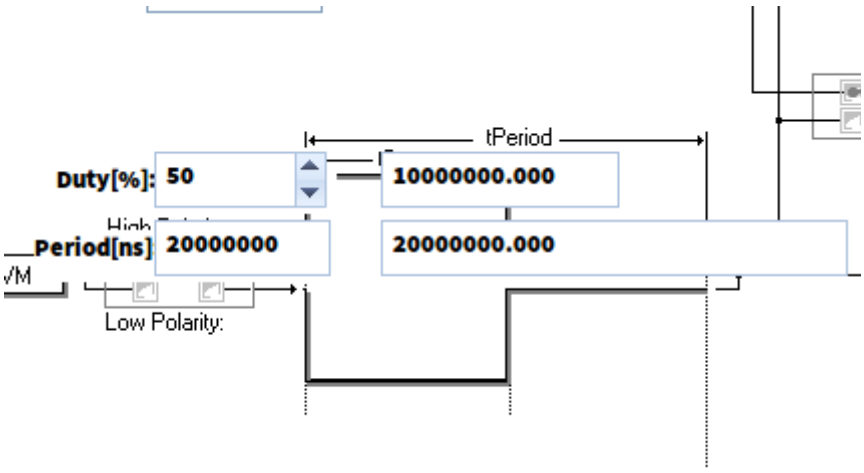
- EQEP2 사용

4) Special Pin Muxing → Enable TBCLK sync 활성화

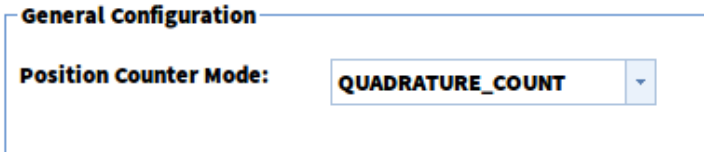


5) ETPWM 설정



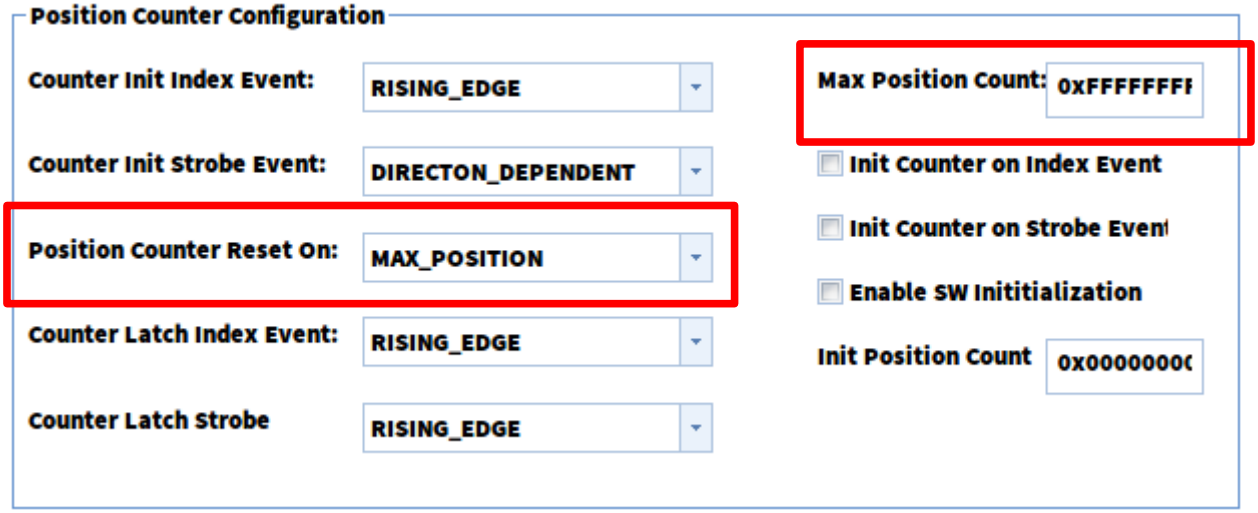


6) EQEP 설정

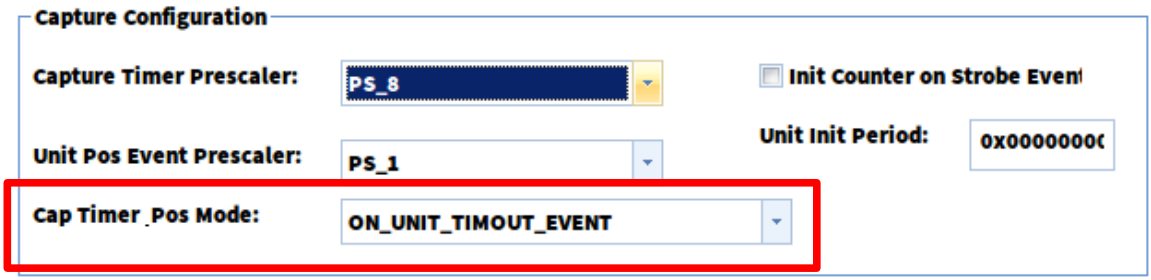


- QUADRATURE_COUNT : QEPA 한 펄스에 4개의 QCLK 발생한다는 의미

- Unit time out interrupt 사용



- unit timeout event 발생 시 capture 될 수 있도록 설정



2. CCS 코드

```
#include "HL_sys_common.h"

/* USER CODE BEGIN (1) */
#include "HL_etpwm.h"
#include "HL_eqep.h"
#include "HL_sci.h"
#include <stdio.h>
#include <string.h>
/* USER CODE END */

#define UnitPeriod 1000000

char buf[128] = {0};
uint8 receive_data[5]={0};
unsigned int buf_len;
float value = 0.0;
float Kp = 0.0;
int flag = 0;

uint32 set_p = 0;

void wait(int);
void sci_display(sciBASE_t * sci, uint8 * text, uint32 len);
void catch_command(sciBASE_t * sci);
/* 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);

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 활성
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           }
174            else
175           {//0이면 역방향
176                deltapos = eqepReadPosnLatch(eqepREG2);
177                deltapos = ~deltapos + 1;
178
179                sprintf(buf, "delta : %d\n\r\0", deltapos);
180                buf_len = strlen(buf);
181                sci_display(sciREG1, (uint8 *)buf, buf_len);
182                wait(100);
183
184           }
185
186            deltaT = eqepREG2->QUPRD;
187
188            sprintf(buf, "QUPRD : %d\n\r\0", deltaT);
189            buf_len = strlen(buf);
190            sci_display(sciREG1, (uint8 *)buf, buf_len);
191            wait(100);
```

```
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             //rpm은 1분당 rotation이므로 6으로 나누어줌
208             sprintf(buf, "rpm : %f\n\n\r\0", velocity / 6.0);
209             buf_len = strlen(buf);
210             sci_display(sciREG1, (uint8 *)buf, buf_len);
211             wait(100);
212
213             //interrupt flag를 clear함
214             eqepClearInterruptFlag(eqepREG2, QEINT_Uto);
215         }
```

```
230     }
231
232 /* USER CODE END */
233
234     return 0;
235 }
236
237
238 /* USER CODE BEGIN (4) */
239 void sci_display(sciBASE_t * sci, uint8 * text, uint32 len)
240 {
241     while(len--)
242     {
243         while((sci->FLR & 0x4) == 4)   //SCI receive in idle state
244             ;                           //0 : ready to receive
245         sciSendByte(sci, *text++);     //1 : not receive any data.
246     }
247 }
248
249 void wait(int delay)
250 {
251     int i;
252     for(i=0; i<delay; i++)
253         ;
254 }
```