# Электромобильдерге арналған микроконтроллер жүйелерін бағдарламалау

Дәуренов ӘД

# Дәріс №6. Таймерлер, үзілістер және нақты уақыт оқиғалары

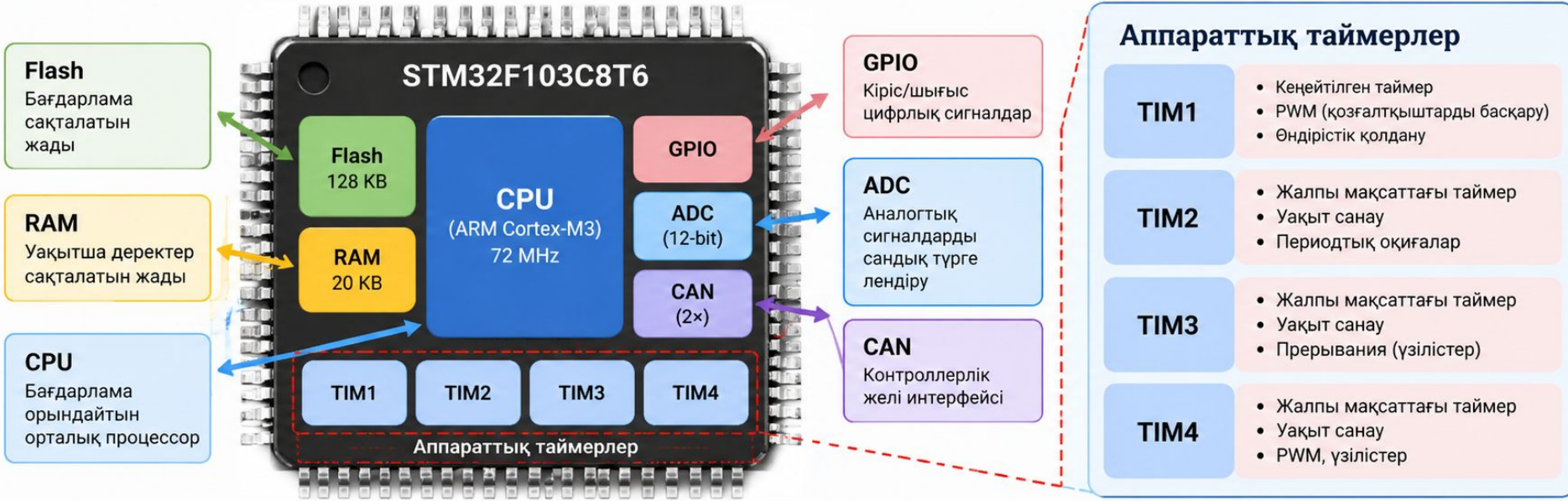
Нақты уақыт жүйесі (Real-Time System) — белгілі бір әрекетті тек дұрыс орындап қана қоймай, оны белгіленген уақыт шегінде орындауы қажет жүйе.

Мысалы, электромобильдің басқару контроллері белгілі бір уақыт аралығында датчиктердің мәндерін оқып, оларды өңдеп, қажетті басқару шешімін қабылдауы тиіс. Нақты орындалу жиілігі жүйенің архитектурасына, аппараттық құралдарына және қауіпсіздік талаптарына байланысты анықталады.

| Электромобиль жүйесі | Уақытпен байланысты тапсырма |
|---|---|
| VCU | Педаль мен тежегіш сигналдарын периодтық тексеру |
| BMS | Аккумулятор параметрлерін белгіленген аралықпен бақылау |
| Inverter Controller | Қозғалтқышты басқару циклдерін дәл уақытпен орындау |
| Thermal Controller | Температураны бақылау және салқындатуды басқару |
| CAN Communication | Хабарламаларды белгіленген мерзімде өңдеу |

# Аппараттық таймер

**Аппараттық таймер** — микроконтроллердің ішінде орналасқан, уақытты санауға және белгілі бір уақыт мезетінде оқиғалар қалыптастыруға арналған арнайы электрондық модуль.




**Таймерлер** — STM32 микроконтроллерінің ішіндегі аппараттық блоктар, сыртқы жеке құрылғылар емес. Таймер процессордың әр командасын күтпей, өзінің аппараттық санауышы арқылы жұмыс істей алады.

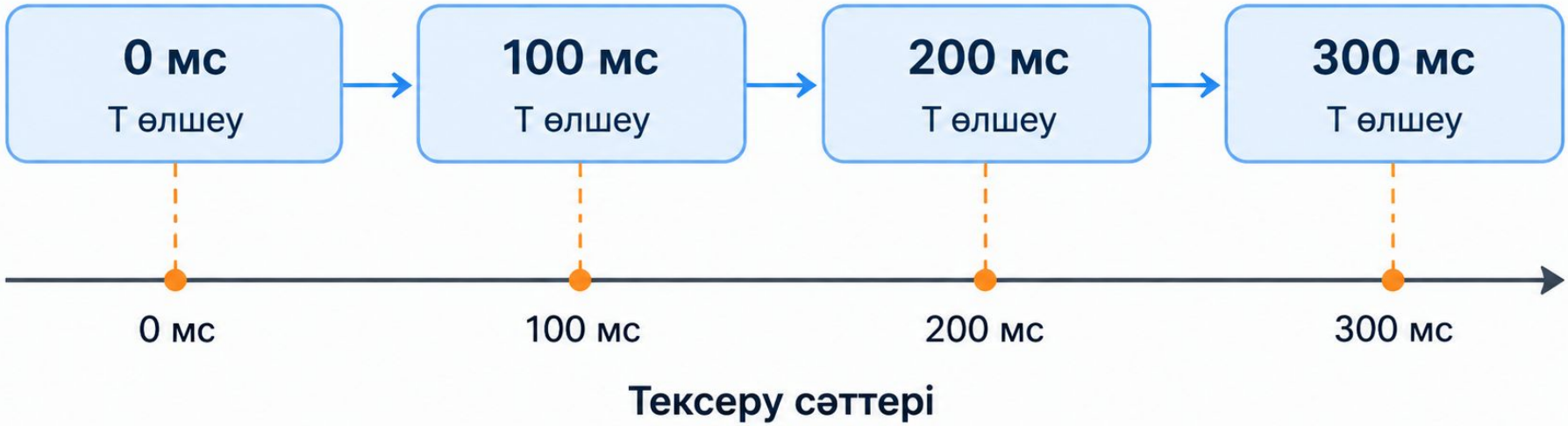
Қарапайым тілмен айтқанда, таймер — микроконтроллердің ішіндегі электрондық сағат немесе санауыш.
Мысалы, электромобиль аккумуляторының температурасын әрбір 100 миллисекунд сайын тексеру қажет делік.

Мұны екі жолмен орындауға болады:

- Процессор уақытты бағдарламалық түрде бақылап отырады.
- Аппараттық таймер уақытты өздігінен санап, қажетті сәтте процессорға хабар береді.

Екінші әдіс тиімді, өйткені процессор таймердің әрбір импульсін жеке санап отырмайды.

## Мысал: температураны әрбір 100 мс сайын тексеру




Таймердің өзі температураны өлшемейді.
Ол тек температураны тексеру уақыты келгенін білдіретін оқиға (Update Event) қалыптастырады.

# 2. STM32 микроконтроллеріндегі таймерлер

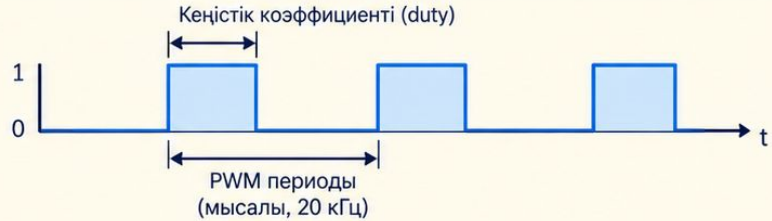
STM32F103C8T6 микроконтроллерінде бірнеше аппараттық таймер бар. Олар микроконтроллердің ішінде орналасқан және сыртқы құрылғылар емес.

| Таймер | Негізгі қызметі | Ерекшеліктері / қолдану мысалы |
|---|---|---|
| **TIM1** | Кеңейтілген таймер, күрделі PWM басқаруы | • Жоғары деңгейлі PWM (complementary output, dead-time)<br>• Моторларды басқару (BLDC, үш фазалы), қуат электроникасы |
| **TIM2** | Жалпы мақсаттағы таймер | • Таймер, периодтық үзіліс, уақыт өлшеу<br>• Оқу жобасында: аккумулятор температурасын әрбір 100 мс сайын тексеру |
| **TIM3** | Жалпы мақсаттағы таймер | • Жалпы мақсаттар: PWM, Input Capture, Output Compare<br>• Оқу жобасында: желдеткішті PWM арқылы басқару |
| **TIM4** | Жалпы мақсаттағы таймер | • Жалпы мақсаттар: уақыт өлшеу, PWM, оқиғаларды санау<br>• Әртүрлі перифериялармен жұмыс (мысалы, сенсорлар) |

**Мысал 1. TIM2 – аккумулятор температурасын әрбір 100 мс сайын тексеру**




**Мысал 2. TIM3 – желдеткішті PWM арқылы басқару**



# Регистр (Register)

Регистр (Register) — микроконтроллердің ішінде орналасқан, белгілі бір санды немесе басқару параметрлерін сақтайтын арнайы жады ұяшығы.

Мысалы, электрондық сағатты елестетейік. Онда уақытты орнатуға, оятқышты қосуға және ағымдағы уақытты көруге болады.

Аппараттық таймерде де осындай параметрлер бар. Олар регистрлер арқылы басқарылады.

## Регистрдің жалпы құрылымы




## Мысал: TIM2_CNT — санаушы регистрі

TIM2_CNT регистрі таймердің ағымдағы санын сақтайды.
Таймер тактілік сигнал келген сайын бұл мән 1-ге артады.




Бұл мәнді программа оқуға да, қажет болған жағдайда жазуға да болады (мысалы, таймерді нөлден бастау үшін 0 жазу).

## Аппараттық таймердің негізгі регистрлері

| Регистр | Қызметі | Түсіндірме / қолдану мысалы |
|---|---|---|
| **TIMx_PSC** | Санау жиілігін төмендетеді | • Таймердің кіріс тактілік жиілігін бөлетін prescaler.<br>• $f_{counter} = f_{timer} / (PSC + 1)$ |
| **TIMx_CNT** | Ағымдағы санау мәнін сақтайды | • Таймердің санаушы (counter) регистрі.<br>• Программа арқылы мәнін оқу/жазу мүмкін. |
| **TIMx_ARR** | Санау кезеңінің шегін анықтайды | • Таймердің автоқайта жүктеу (auto-reload) регистрі.<br>• CNT осы мәнге жеткенде update event пайда болады. |
| **TIMx_CCR** | Салыстыру мәнін сақтайды | • Output Compare режимінде қолданылады.<br>• CNT == CCR болғанда салыстыру оқиғасы жасалады. |
| **TIMx_DIER** | Үзілістер мен DMA сұраныстарына рұқсат береді | • Қай оқиғалардан үзіліс (interrupt) немесе DMA сұранысы шығатынын анықтайды. |
| **TIMx_SR** | Таймер оқиғаларының күй жалаушыларын сақтайды | • Оқиғалардың болғанын көрсететін жалаулар (UIF, CCxIF, TIF, ...).<br>• Программа бұл жалауларды оқып, қажет болса тазартады. |

Мұндағы x — таймердің нөмірі (мысалы, TIM2, TIM3). Сондықтан **TIM2_CNT** — TIM2 таймерінің санаушы регистрі.
Бұл регистрлер микроконтроллердің ішінде орналасқан және тікелей жады мекенжайлары арқылы қол жеткізіледі (STM32-де олар HAL немесе LL кітапханалары арқылы ыңғайлы түрде басқарылады).

# PSC — бөлгіш (Prescaler)

Микроконтроллер өте жоғары жиілікте жұмыс істейді. Мысалы, таймердің кіріс тактілік жиілігі 72 МГц деп алайық.

Бұл бір секундта 72 миллион тактілік импульс келетінін білдіреді.
Бірақ бізге мұндай жоғары жиілікте санау қажет емес. Сондықтан PSC регистрі жиілікті төмендетеді.

# PSC (Prescaler) қалай жиілікті төмендетеді?

PSC регистрі таймерге келетін тактілік импульстерді (жиілікті) белгілі бір санға бөледі.
Нәтижесінде таймердің санаушысы (CNT) баяулайды.

## 1. Таймердің бастапқы жиілігі

$f_{timer}$ = 72 МГц



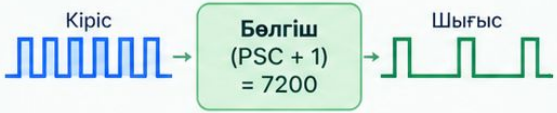
72 000 000 импульс / секунд

- Микроконтроллер таймерге секундына 72 миллион тактілік импульс береді (72 МГц).
- Бұл импульстер өте жылдам.

## 2. Prescaler (PSC)

**PSC = 7199**

Жиілікті 7200 есе бөледі




- PSC регистрінің мәні 7199.
- Нақты бөлу коэффициенті: PSC + 1 = 7200 (себебі есептеу нөлден басталады).
- Яғни, әрбір 7200 импульстің тек біреуі таймердің санаушысына беріледі.

## 3. Таймердің санау жиілігі (CNT)

$f_{CNT}$ = 10 кГц



10 000 импульс / секунд

- Бөлуден кейін таймердің санаушысы секундына 10 000 рет артады.
- Бастапқы 72 МГц жиілік 7200 есе азаяды: 72 000 000 ÷ 7200 = 10 000.

## PSC мәні қалай есептелді?

1. Бастапқы жиілік:
   $f_{timer}$ = 72 000 000 Гц
2. Қажетті санау жиілігі:
   $f_{CNT}$ = 10 000 Гц
3. Бөлу коэффициенті:
   $\frac{72\,000\,000}{10\,000}$ = 7200
4. PSC мәні (нолден басталады):
   PSC = 7200 – 1 = 7199

## Формула

$$f_{CNT} = \frac{f_{timer}}{PSC + 1}$$

- $f_{timer}$ — таймердің кіріс жиілігі (Гц)
- PSC — prescaler регистрінің мәні
- $f_{CNT}$ — санаушының жиілігі (Гц)

## Неліктен PSC + 1?

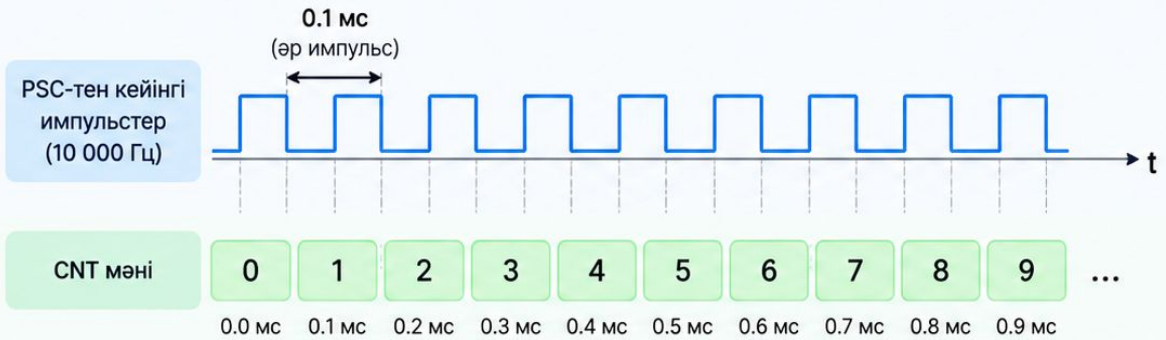
PSC регистрі нөлден бастап есептеледі,
яғни нақты бөлгіш коэффициенті PSC + 1 болады.

| PSC мәні | Нақты бөлу коэффициенті |
|---|---|
| 0 | 1 (бөлмейді) |
| 1 | 2 |
| 9 | 10 |
| 7199 | 7200 |

# 5. CNT — санаушы (Counter)

**CNT (Counter)** — таймердің ағымдағы санау мәнін көрсететін **регистр**. PSC бөлгішінен кейін келетін импульстерді санайды.
Біздің мысалда PSC = 7199 болғандықтан, таймерге секундына 10 000 импульс келеді. CNT осы импульстерді 0-ден бастап 1-ге, 2-ге, 3-ке ... бір-бірден арттырады.

## CNT қалай өседі? (мысал: 10 000 Гц)




Әрбір 0,1 мс сайын (10 000 Гц = 0,1 мс) CNT мәні 1-ге артады.
Кестеде санаудың алғашқы 10 мәні көрсетілен.

## Негізгі идея

- PSC жиілікті төмендетеді (бөледі).
- PSC-тен кейінгі әрбір импульс келгенде CNT мәні 1-ге артады.
- CNT — таймердің ағымдағы санау мәні.
- Ол әр импульс сайын біртіндеп өседі: 0, 1, 2, 3, ...

## Біздің мысалда

- Таймердің бастапқы жиілігі: 72 МГц
- PSC = 7199 → 72 000 000 ÷ 7200 = 10 000 Гц
- Демек, санаушыға секундына 10 000 импульс келеді.
- Бір импульстің уақыты: 1 / 10 000 = 0,1 мс
- Сондықтан CNT мәні әр 0,1 мс сайын 1-ге артады.

## CNT мәні нақты уақытта




## Есте сақтау

- CNT уақытты тікелей өлшемейді, ол тек импульстердің санын көрсетеді.
- CNT қай мәнге дейін санайтынын ARR регистрі анықтайды.

# 6. ARR — санау шегі (Auto-Reload Register)

**ARR (Auto-Reload Register)** — таймердің санаушысы (CNT) қай мәнге дейін санайтынын анықтайды.
CNT осы мәнге жеткенде, келесі тактта 0-ге қайтып, **Update Event** (жаңарту оқиғасы) қалыптасады.

## 1. Санаушының жиілігі

Алдыңғы мысалда PSC = 7199 болғандықтан, санаушының жиілігі:

$$f_{CNT} = 10\,000\ \text{Гц}$$

Яғни CNT секундына 10 000 рет артады (бір импульс = 0,1 мс).

## 2. Қажетті уақыт

Әр 100 мс сайын оқиға қалыптастыру керек.

$$T = 100\ \text{мс} = 0{,}1\ \text{с}$$

## 3. Қажетті санау қадамы

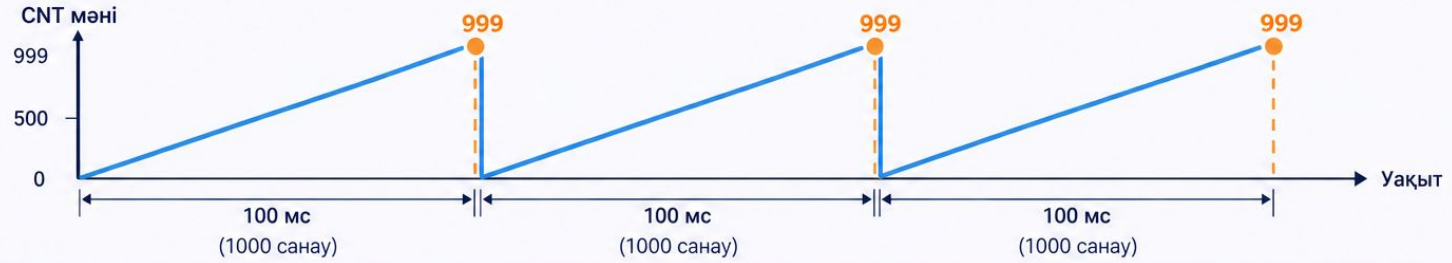
Осы уақыт аралығында қанша импульс санау керектігін табамыз:

$$N = f_{CNT} \times T = 10\,000 \times 0{,}1 = 1000$$

Санау 0-ден басталатындықтан:

$$ARR = N - 1 = 1000 - 1 = 999$$

## CNT және ARR арасындағы байланыс (ARR = 999)




## Жұмыс істеу реті

1. CNT 0-ден бастап әр импульсте 1-ге артады.
2. CNT 999 мәніне жетеді.
3. Келесі тактта CNT автоматты түрде 0-ге қайтады.
4. Осы сәтте **Update Event** (жаңарту оқиғасы) қалыптасады.
5. Бұл процесс әр 100 мс сайын қайталанады.

## Нәтиже

Оқиға қалыпасу периоды:
**100 мс (10 Гц)**

**Параметрлер:**
- PSC = 7199 ($f_{CNT}$ = 10 000 Гц)
- ARR = 999 (1000 санау)

**Нәтиже:**
Әр 100 мс сайын Update Event.

## Есте сақтау

- **ARR** — уақыты емес, санау шегі.
- Нақты уақыт таймердің санау жиілігіне ($f_{CNT}$) байланысты.
- Санау 0-ден басталатындықтан, **ARR = N − 1.**
- CNT = ARR болғанда, келесі тактта 0-ге қайтадан өтеді және **Update Event** пайда болады.

# 7. Таймердің периодын есептеу

Таймердің қандай уақыт аралығында (периодта) оқиға қалыптастыратынын PSC және ARR регистрлері анықтайды.
Период (T) — санаушы 0-ден ARR мәніне дейін санап, келесі тактта 0-ге қайта оралатын уақыт.

## 1. Таймердің кіріс жиілігі

$$f_{\text{timer}} = 72\,000\,000 \text{ Гц}$$
(72 МГц)



Таймерге секундына 72 миллион тактілік импульс келеді.

## 2. PSC — жиілікті төмендетеді

PSC = 7199 → 7200 есе бөледі.

$$f_{CNT} = \frac{f_{\text{timer}}}{PSC + 1}$$

$f_{CNT}$ = 72 000 000 ÷ (7199 + 1)
= 72 000 000 ÷ 7200 = 10 000 Гц



Енді санаушының жиілігі 10 000 Гц
(секундына 10 000 рет).

## 3. ARR — санау қадамын анықтайды

ARR = 999 → 1000 санау қадамы.

CNT 0-ден 999-ға дейін санайды,
келесі тактта 0-ге қайтады.




## 4. Периодты есептеу формуласы

$$T = \frac{(PSC + 1)(ARR + 1)}{f_{\text{timer}}}$$

Мұндағы:

- $T$ — таймер кезеңі, секунд.
- $PSC$ — бөлгіш мән.
- $ARR$ — санау шегі.
- $f_{\text{timer}}$ — таймердің кіріс жиілігі.

## 5. Мысал: 100 мс период

PSC = 7199 — 7200 есе бөледі
ARR = 999 — 1000 санау
$f_{\text{timer}}$ = 72 000 000 Гц — таймердің кіріс жиілігі

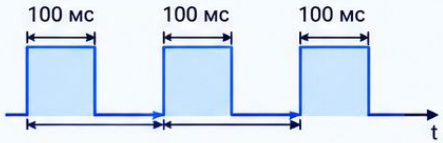
$$T = \frac{(7199 + 1)(999 + 1)}{72\,000\,000}$$

$$T = \frac{7200 \times 1000}{72\,000\,000}$$

$$T = 0.1 \text{ с} = 100 \text{ мс}$$

## Нәтиже және уақыт диаграммасы

Таймер әр 100 мс сайын Update Event
(жаңарту оқиғасы) қалыптастырады.




Яғни, **таймер секундына 10 рет** жаңарту оқиғасын қалыптастырады (1 / 0.1 = 10 Гц).

# 10. Таймердің толық жұмыс істеу принципі

Таймер тактілік импульстерді қабылдайды, жиілікті PSC арқылы төмендетеді, CNT санауышысын арттырады, ARR мәніне жеткенде Update Event (жаңарту оқиғасы) пайда болады. Бұл оқиға бойынша үзіліс шақырылуы немесе PWM сигналы қалыптастырылуы мүмкін.

**1 Timer Clock**

72 МГц



72 000 000 импульс / сек

**2 PSC = 7199**

Жиілікті 7200 есе төмендетеді

$f_{CNT}$ = 10 000 Гц

10 000 импульс / сек (бір импульс = 0,1 мс)

**3 CNT (санауышы)**

Әр импульсте 1-ге артады

0 1 2 3 ⋯

**4 ARR = 999**

CNT осы мәнге жеткенде...

CNT = 999

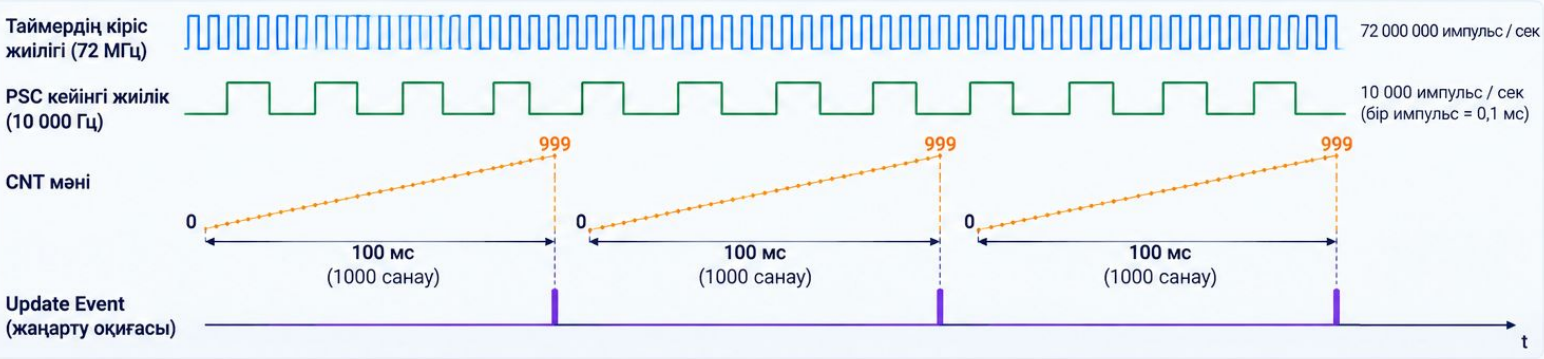
**5 Update Event**

Жаңарту оқиғасы қалыптасады

**6 Interrupt / PWM**

Update Event арқылы үзіліс шақырылады немесе PWM сигналы қалыптастырылады






## Периодты есептеу формуласы

$$T = \frac{(PSC + 1)(ARR + 1)}{f_{timer}}$$

T — таймер периоды, секунд

PSC — бөлгіш мәні

ARR — санау шегі

$f_{timer}$ — таймердің кіріс жиілігі (Гц)

## Біздің мысал

$f_{timer}$ = 72 000 000 Гц

PSC = 7199 (÷ 7200)

ARR = 999 (1000 санау)

$$T = \frac{(7199 + 1)(999 + 1)}{72\,000\,000} = 0{,}1 \text{ с} = 100 \text{ мс}$$

## Нәтиже

- Таймер әр 100 мс сайын Update Event қалыптастырады.
- Бұл оқиға бойынша CPU-ға үзіліс жібереді (TIMx_IRQHandler).
- Немесе PWM сигналы CNT және CCR мәндерін салыстыру арқылы қалыптастырылады.

| Термин | Қарапайым түсіндірме |
| --- | --- |
| Timer | Микроконтроллер ішіндегі электрондық санауыш |
| Register | Параметр немесе мән сақталатын арнайы ұяшық |
| Timer Clock | Таймердің тактілік импульстері |
| PSC | Санау жиілігін төмендетеді |
| CNT | Импульстерді санайды |
| ARR | Санау кезеңінің шегін белгілейді |
| CCR | Салыстыру мәнін орнатады, PWM үшін қолданылады |
| Update Event | Таймердің жаңарту оқиғасы |
| Interrupt | Процессорға оқиғаны өңдеу қажеттігін хабарлайды |
| PWM | Импульс енін өзгерту арқылы басқару |

Апппараттық таймер — STM32 микроконтроллерінің ішіндегі дербес жұмыс істейтін санауыш. Бағдарламашы PSC және ARR регистрлерін баптайды. Таймер импульстерді санап, белгілі бір уақыт сайын оқиға қалыптастырады. Үзіліс арқылы процессор сол оқиғаға жауап бере алады, ал CCR регистрінің көмегімен PWM сигналын қалыптастыруға болады.

Ең алдымен PSC, CNT және ARR регистрлерінің айырмашылығын және олардың көмегімен 100 мс уақыт аралығын есептеуді түсінгені дұрыс. Бұл үзілістер мен PWM тақырыптарын меңгерудің негізі болады.

# Interrupt (Үзіліс) дегеніміз не?

Interrupt (Үзіліс) — микроконтроллерге маңызды оқиға болғанын хабарлайтын механизм.
Қарапайым тілмен айтқанда, үзіліс — процессорға «Маңызды оқиға болды, соған назар аудар!» деген сигнал.

**1. Күнделікті өмірден мысал**
Сіз компьютерде жұмыс істеп отырсыз. Бір кезде телефоныңыздың оятқышы шырылдайды. Сіз:

1. Компьютердегі жұмысыңызды уақытша тоқтатасыз.
2. Телефондағы оятқышты өшіресіз.
3. Қайтадан компьютердегі жұмысыңызды жалғастырасыз.

Микроконтроллерде де осыған ұқсас процесс жүреді.

# Таймер мен Interrupt арасындағы айырмашылық

Бұл екеуі бір нәрсе емес.

- Timer — уақытты санайды.
- Interrupt — процессорға белгілі бір оқиға болғанын хабарлайды.

Мысалы, TIM2 таймерін 100 мс сайын оқиға қалыптастыратындай етіп баптадық. Таймер жұмыс істеп тұрады:

0 мс → 100 мс → 200 мс → 300 мс → ...

Әрбір 100 мс сайын, егер үзіліске рұқсат берілсе, процессорға үзіліс сұранысы жіберіледі.

# Үзіліс болғанда CPU не істейді?

Процессор үзілісті қабылдаған кезде арнайы функцияны орындайды.
Бұл функция **Interrupt Service Routine (ISR)** — үзілісті өңдеу функциясы деп аталады.

Мысалы:

```
void timerInterrupt() {
    checkTemperature = true;
}
```

Бұл — үзіліс өңдегішінің қарапайым шартты мысалы.
Таймер үзілісі келген кезде checkTemperature айнымалысы true болады. Содан кейін негізгі бағдарлама температураны тексере алады.

# Неліктен үзіліс қажет?

Үзіліс болмаса, бағдарлама оқиғаның болған-болмағанын үнемі тексеріп отыруы мүмкін. Бұл әдіс Polling деп аталады.
Ал үзіліс қолданылғанда, процессор басқа тапсырмаларды орындап отыра алады. Оқиға болған кезде ғана тиісті өңдегіш іске қосылады.

Есте сақта:

- Timer — уақытты санайды.
- Interrupt — оқиға туралы хабарлайды.
- ISR — сол оқиғаға жауап ретінде орындалатын функция.
- CPU — ISR орындалғаннан кейін негізгі бағдарламаға қайтады.

Бір сөйлеммен: Interrupt — микроконтроллердің негізгі бағдарламасын уақытша тоқтатып, маңызды оқиғаны өңдеуге, содан кейін бұрынғы жұмысына оралуға мүмкіндік беретін механизм.

# Polling және Interrupt айырмашылығы

Polling — процессор құрылғының күйін қайта-қайта тексереді.

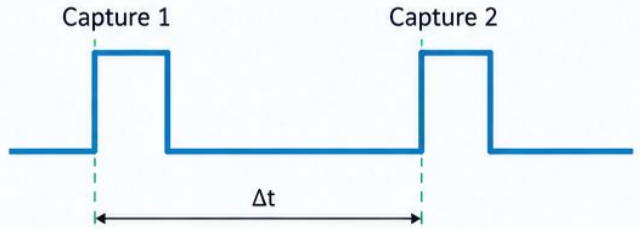
Interrupt — оқиға болғанда аппараттық модуль процессорға үзіліс сұранысын береді.

**Periodic Interrupt**/Периодтық үзіліс — таймердің белгілі бір тұрақты уақыт аралығында үзіліс қалыптастыруы.

# 10. Input Capture — кіріс оқиғасының уақытын тіркеу

**Input Capture** — таймердің кіріс сигналының белгілі бір өзгерісі (көтерілу немесе түсу фронты) кезінде санаушының ағымдағы мәнін аппараттық түрде тіркеу механизмі.

### Импульстер арасындағы уақыт




### Есептеу мысалы

Егер санаушының жиілігі 1 МГц болса, бір санақ 1 мкс-қа тең. Бірінші тіркелген мән 1000, екіншісі 6000 болсын:

$$\Delta t = (6000 - 1000) \times 1\ \text{мкс} = 5\ \text{мс}$$

Сәйкес жиілік:

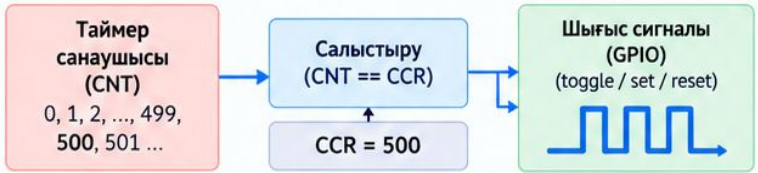
$$f = \frac{1}{0.005} = 200\ \text{Гц}$$

### Қолдану мысалдары

- Импульстік жылдамдық датчиктері (RPM)
- Ультрадыбыстық қашықтық өлшеу
- Сыртқы сигналдардың уақытын талдау
- Период пен жиілікті өлшеу

# 11. Output Compare — шығыс оқиғасын салыстыру арқылы қалыптастыру

**Output Compare** — таймер санаушысының мәні алдын ала берілген салыстыру мәніне (CCR) жеткенде аппараттық оқиға қалыптастыру механизмі. Бұл кезде шығыс сигналының күйін өзгертуге болады (GPIO toggle, PWM, импульс генерациясы және т.б.).




### Мысал

CNT = **500** болғанда
CCR1 = 500 мәнімен сәйкестік пайда болады.

Осы оқиға арқылы таймер кесінді оқиғасын қалыптастырады немесе шығыс сигналының күйін өзгертеді.

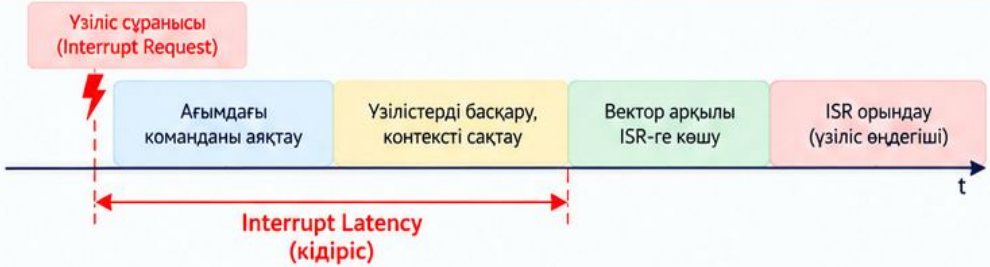
### Нені басқаруға болады?

- Шығыс сигналдарының уақытын тіркеу
- Шығыс оқиғасын жоспарлау
- Дәл уақыттық импульстар (PWM)
- Бір реттік немесе периодтық оқиғалар

| Режим / қолдану | Сыртқы сигнал | Кіріс оқиғасы | Нәтиже |
|---|---|---|---|
| Input Capture | Уақытын тіркейді | Кіріс оқиғасын өлшейді | Уақыт, период, жиілік өлшеу |
| Output Compare | Шығыс сигналын қалыптастырады | Салыстыру мәніне жеткенде | Дәл уақыттық сигналдар (PWM, импульс) |

# 12. Interrupt Latency — үзілістің кідірісі

**Interrupt Latency** — үзіліс сұранысы пайда болғаннан бастап процессордың оны өңдеуге кіруіне дейінгі уақыт.




**Оған мына факторлар әсер етеді:**

- Ағымдағы орындалатын команда;
- Үзілістердің басымдықтары (priority);
- Уақытша бұғатталған үзілістер (disable/enable);
- Басқа ISR өңдегіштері;
- Процессор архитектурасы (Cortex-M3/M4);
- Жадтан оқу және стекке сақтау уақыты.

Нақты уақыт жүйесінде тек тапсырманы орындау уақыты ғана емес, оны **қажетті мерзімде орындау (latency + execution time)** маңызды.