Programowanie mikrokontrolerów 2.0

Liczniki

Marcin Engel Marcin Peczarski

Instytut Informatyki Uniwersytetu Warszawskiego

2 listopada 2021

Liczniki

- Układy sprzętowe wyposażone w wewnętrzny rejestr modyfikowany o 1 przy odpowiedniej zmianie stanu wejścia taktującego
- Na wejście taktujące licznika można podać (podzielony przez preskaler):
  - sygnał zegarowy
  - sygnały zewnętrzne z pewnych wyprowadzeń mikrokontrolera
  - sygnały wewnętrzne (np. zdarzenia generowane przez inne liczniki)
- Licznik może służyć do:
  - cyklicznego zgłaszania przerwania
  - generowania różnego rodzaju przebiegów na pewnych wyprowadzeniach mikrokontrolera
  - zliczania zdarzeń zewnętrznych
  - odmierzania czasu między pewnymi zdarzeniami zewnętrznymi
  - ...

Liczniki w STM32F411

- STM32F411 jest wyposażony w następujące liczniki:
  - rozbudowany 16-bitowy licznik `TIM1` podłączony do szyny `APB2`
  - liczniki podłączone do szyny `APB1`:
    - 16-bitowe `TIM3`, `TIM4`
    - 32-bitowe `TIM2`, `TIM5`
  - 16-bitowe liczniki podłączone do szyny `APB2`: `TIM9`, `TIM10`, `TIM11`
- Poszczególne liczniki oferują różną funkcjonalność w zakresie:
  - trybu zliczania (w górę, w dół, w obie strony)
  - liczby kanałów wejściowych i wyjściowych
  - rodzajów generowanych zdarzeń i przerwań
  - dodatkowych możliwości (liczniki powtórzeń, wyjścia komplementarne z opóźnieniami, synchronizacja z innymi układami)

Rejestry liczników

- `TIMx->CNT` – counter
  - 16-bitowy lub 32-bitowy rejestr, zawiera aktualną wartość licznika
  - może być odczytywany i zapisywany nawet podczas pracy licznika
- `TIMx->PSC` – prescaler
  - 16-bitowy rejestr umożliwiający podzielenie częstotliwości taktowania licznika przez wartość od 1 do 65536
  - buforowany
- `TIMx->ARR` – auto-reload register
  - 16-bitowy lub 32-bitowy rejestr określający wartość, do której zlicza licznik
  - może być buforowany
- `TIMx->CR1`, `TIMx->CR2` – control register
- `TIMx->DIER` – DMA and interrupt enable register
- `TIMx->SR` – status register
- ...

Sygnał zegarowy

- Licznik może być taktowany
  - sygnałem z zewnętrznego wyprowadzenia – o tym opowiadać nie będziemy
  - sygnałami generowanymi przez inne liczniki – o tym opowiemy nieco później
  - zegarem wewnętrznym
- W STM32F411 po uruchomieniu mikrokontrolera licznik `TIMx` jest taktowany zegarem o częstotliwości

$$f_{\mathrm{CK\_TIM}x} = 16 \text{ MHz}$$

- Jak zmienić tę częstotliwość – opowiemy później

Preskaler

- Umożliwia zmniejszenie częstotliwości zmian wartości licznika
- Jest wyposażony we własny licznik wewnętrzny
- Konfiguruje się go za pomocą rejestru `TIMx->PSC`
- Umożliwia zmniejszenie częstotliwości zmian wartości licznika

$$f_{\mathrm{CK\_CNT}x} = \frac{f_{\mathrm{CK\_TIM}x}}{\texttt{TIM}x\texttt{->PSC} + 1}$$

- Zmiany wartości rejestru `TIMx->PSC` są propagowane do preskalera w ściśle określonych momentach

Tryby pracy liczników

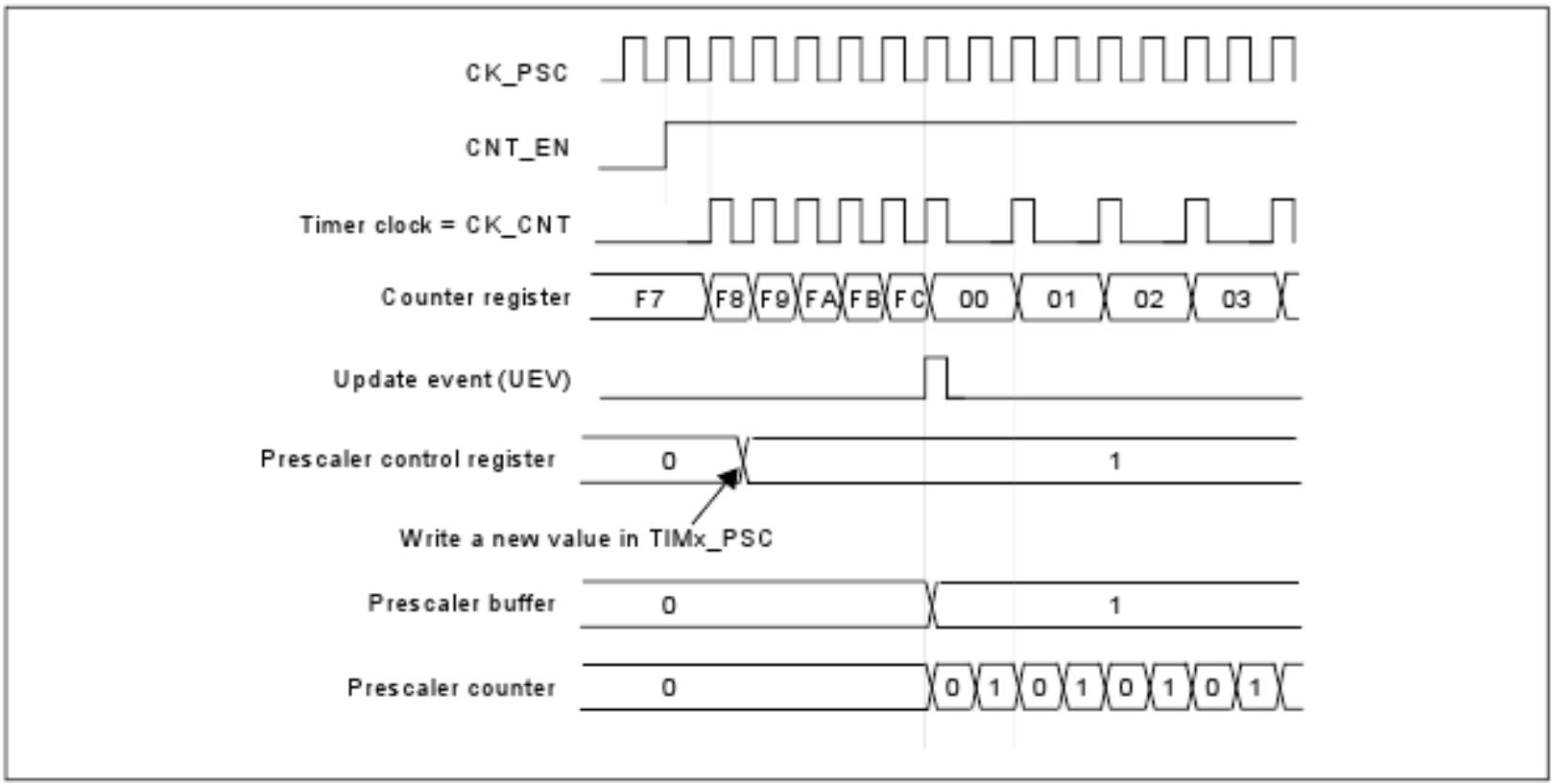
- Zliczanie w górę
  - licznik zlicza od 0 do `TIMx->ARR`
  - zgłasza zdarzenie nadmiaru i powtarza cykl
- Zliczanie w dół
  - licznik zlicza od `TIMx->ARR` do 0
  - zgłasza zdarzenie niedomiaru i powtarza cykl
- Zliczanie w górę i w dół
  - licznik zlicza od 0 do `TIMx->ARR` $- 1$
  - zgłasza zdarzenie nadmiaru
  - następnie zlicza od `TIMx->ARR` do 1
  - zgłasza zdarzenie niedomiaru i powtarza cykl

Zdarzenia uaktualnienia

- Zdarzenie uaktualnienia to zdarzenie nadmiaru lub niedomiaru
- Powoduje zapisanie wartości rejestrów buforowanych (`TIMx->PSC`, ew. `TIMx->ARR`) do rzeczywistych rejestrów
- Jego wystąpienie powoduje ustawienie flagi `TIM_SR_UIF`
- Może też wyzwolić przerwanie

Trochę rysunków – aktualizacja rejestru `TIMx->PSC`

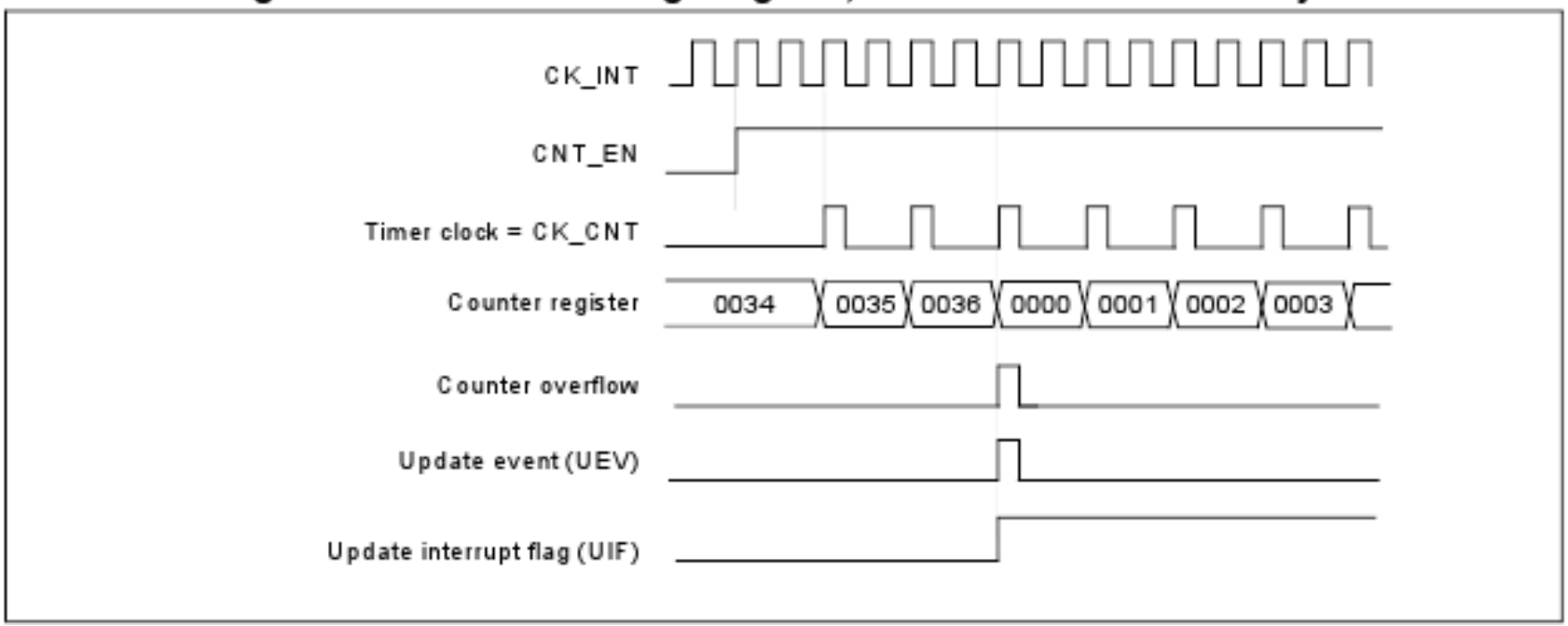
Figure 88. Counter timing diagram with prescaler division change from 1 to 2



Źródło: RM0383 – Reference manual, STM32F411xC/E advanced ARM-based 32-bit MCUs

Trochę rysunków – zliczanie w górę

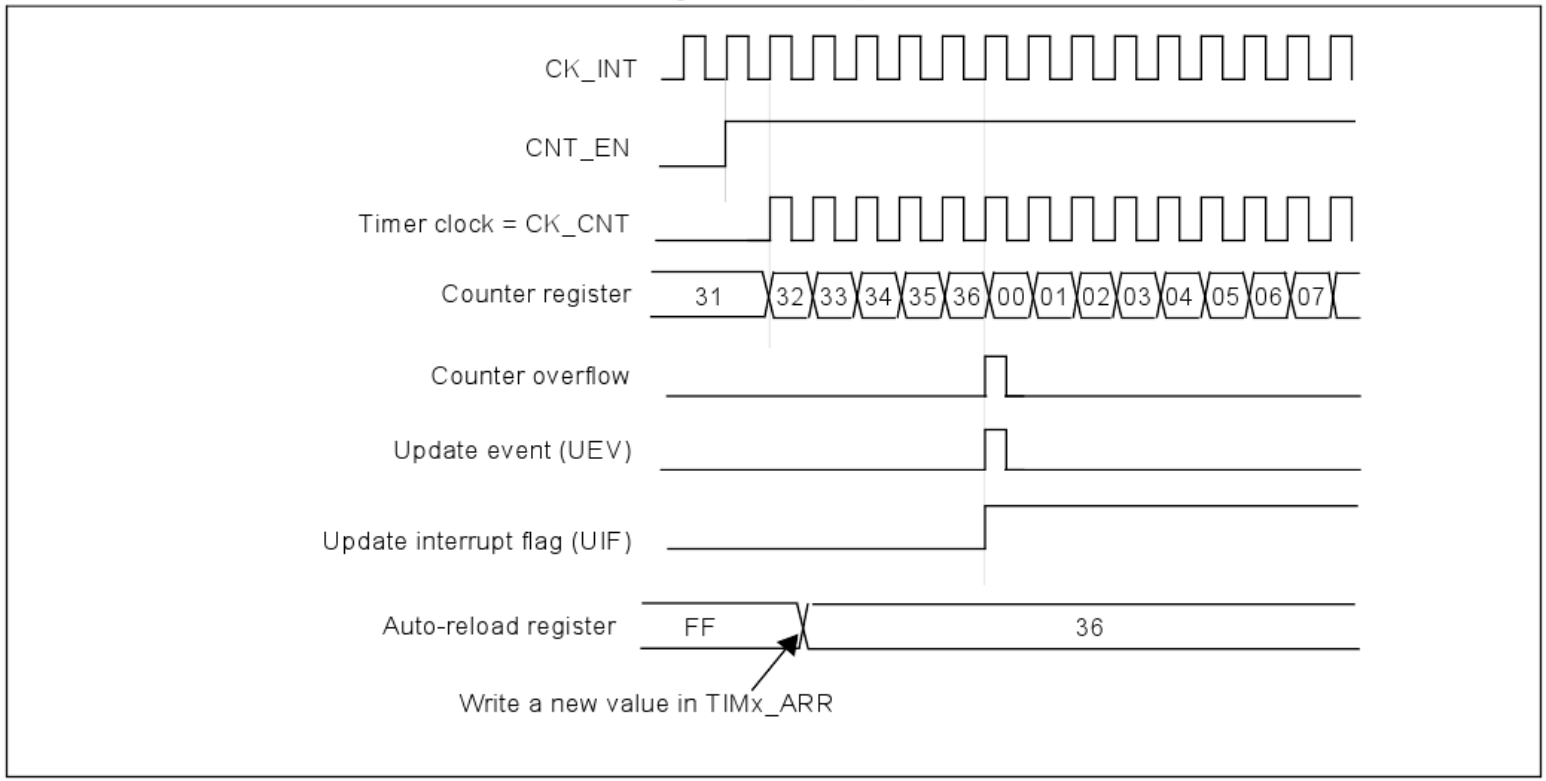
Figure 91. Counter timing diagram, internal clock divided by 2



Źródło: RM0383 – Reference manual, STM32F411xC/E advanced ARM-based 32-bit MCUs

Trochę rysunków – zmiana wartości `TIMx->ARR`

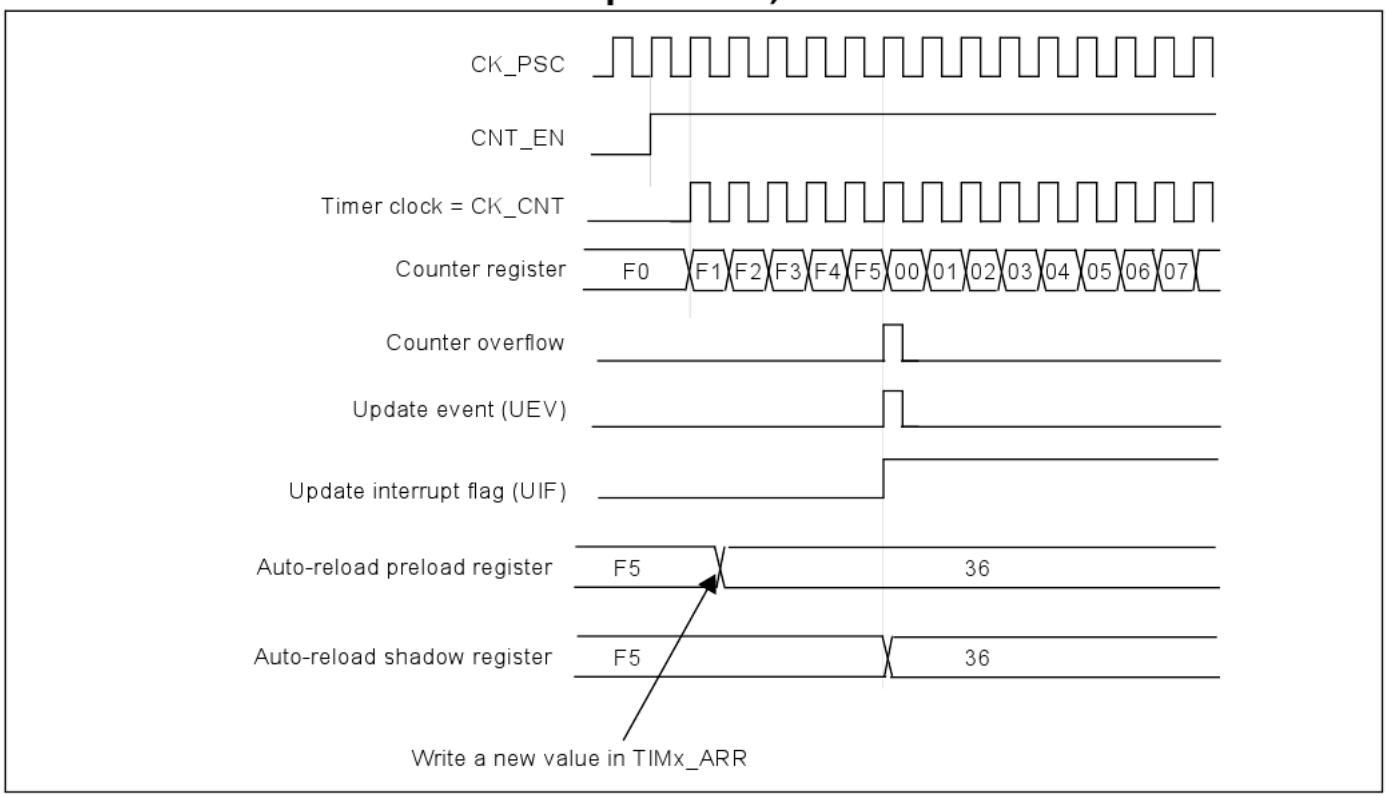
Figure 94. Counter timing diagram, Update event when ARPE=0 (TIMx_ARR not preloaded)



Źródło: RM0383 – Reference manual, STM32F411xC/E advanced ARM-based 32-bit MCUs

Trochę rysunków – zmiana buforowanej wartości TIMx->ARR

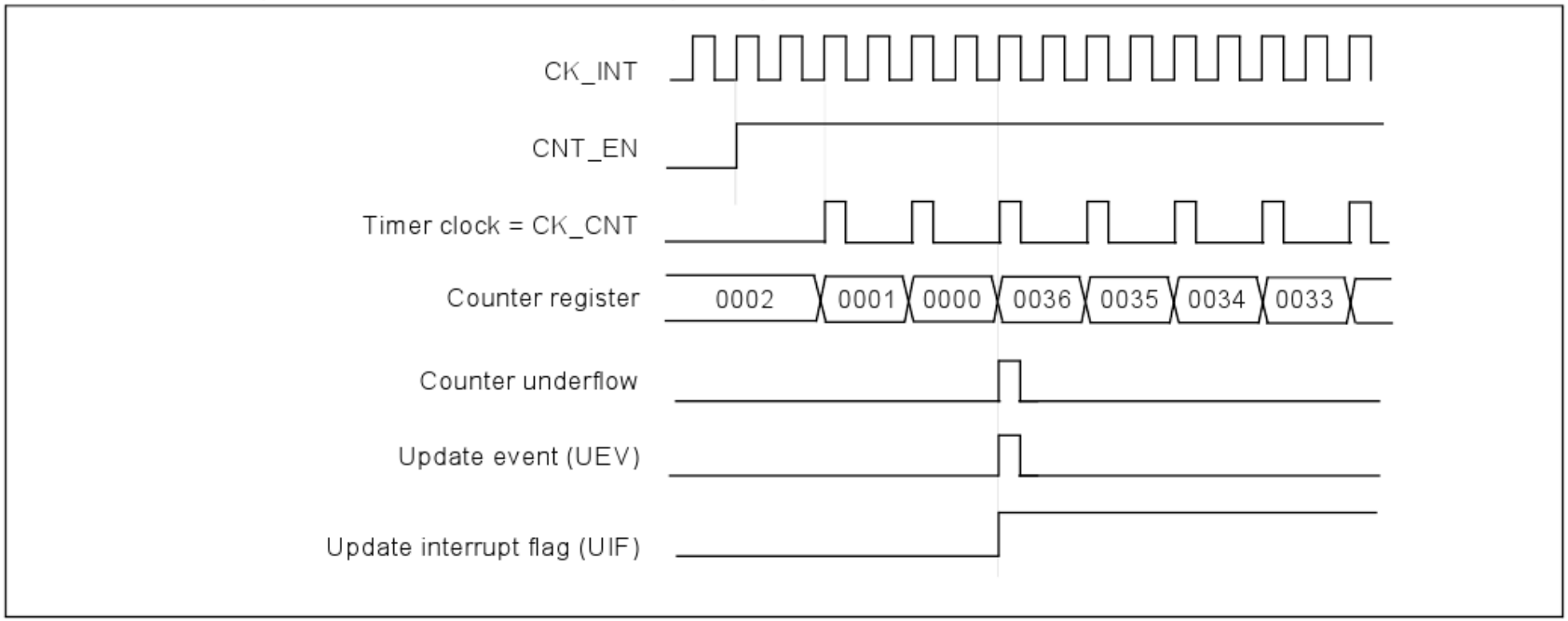
Figure 95. Counter timing diagram, Update event when ARPE=1 (TIMx_ARR preloaded)



Źródło: RM0383 – Reference manual, STM32F411xC/E advanced ARM-based 32-bit MCUs

Trochę rysunków – zliczanie w dół

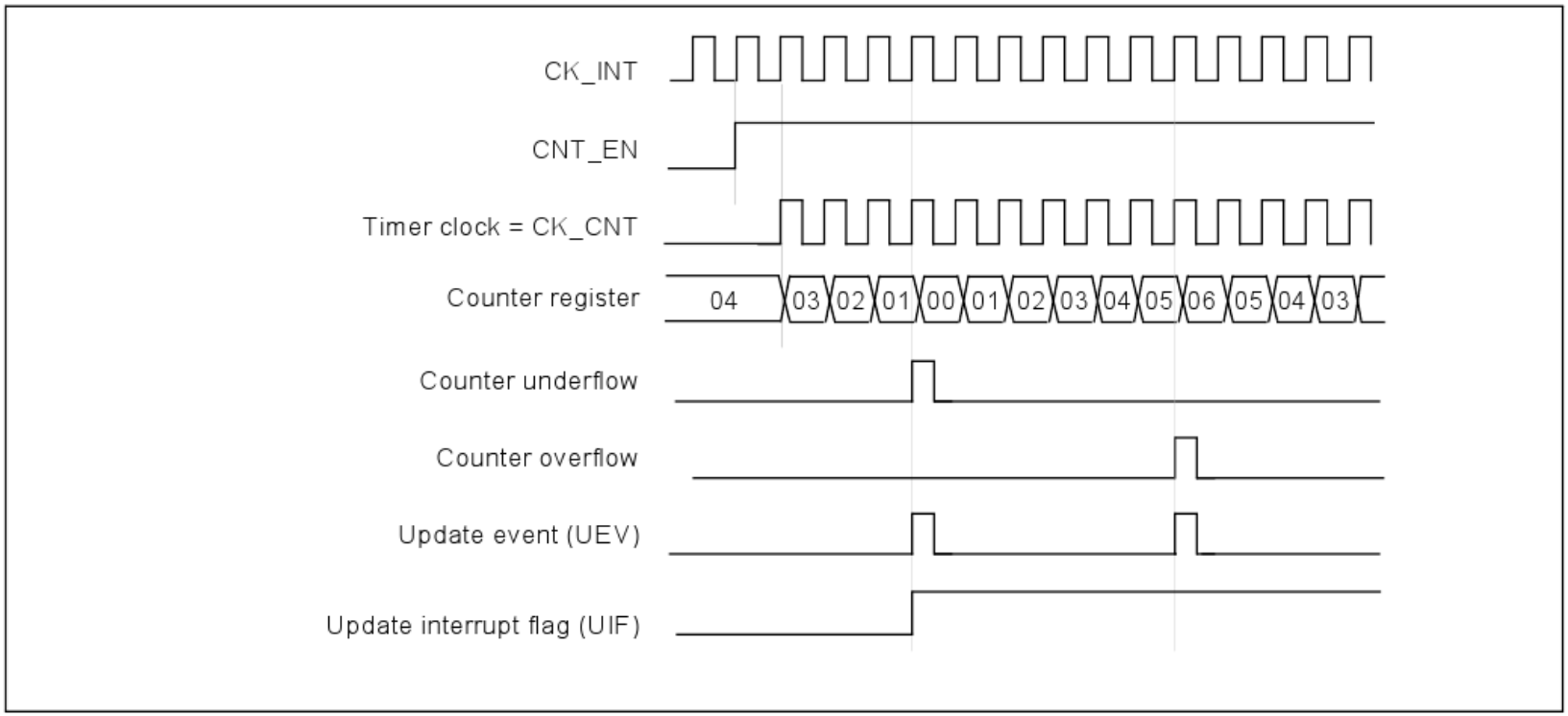
Figure 97. Counter timing diagram, internal clock divided by 2



Źródło: RM0383 – Reference manual, STM32F411xC/E advanced ARM-based 32-bit MCUs

Trochę rysunków – zliczanie w górę i w dół

Figure 101. Counter timing diagram, internal clock divided by 1, TIMx_ARR=0x6



Źródło: RM0383 – Reference manual, STM32F411xC/E advanced ARM-based 32-bit MCUs

Inicjowanie licznika

- Skonfigurowanie trybu zliczania
  - w górę

    ```
    TIM3->CR1 = 0;
    ```
  - w dół

    ```
    TIM3->CR1 = TIM_CR1_DIR;
    ```
  - w obie strony (bit `TIM_CR1_DIR` jest wtedy tylko do odczytu i jest ustawiany sprzętowo)

    ```
    TIM3->CR1 = TIM_CR1_CMS_0 | TIM_CR1_CMS_1;
    ```
- Skonfigurowanie preskalera i zakresu zliczania

  ```
  TIM3->PSC = ...;
  TIM3->ARR = ...;
  ```
- Wartość preskalera jest buforowana, a wartość maksymalna licznika może być buforowana
- Aby już pierwszy cykl zliczania odbywał się z nowymi wartościami, trzeba wymusić zdarzenie uaktualnienia

Wymuszanie zdarzenia uaktualnienia

- Ustawiamy

  `TIM3->EGR = TIM_EGR_UG;`
- Ustawienie bitu `TIM_EGR_UG` powoduje:
  - zainicjowanie rejestru `TIMx->PSC` i ew. `TIMx->ARR`
  - zainicjowanie rejestru `TIMx->CNT` na
    - `TIMx->ARR`, jeśli jest ustawiony bit `TIM_CR1_DIR`
    - 0 w przeciwnym razie
  - wyzerowanie wewnętrznego rejestru preskalera
  - wygenerowanie zdarzenia uaktualnienia, o ile nie jest ustawiony bit `TIM_CR1_UDIS`
  - ustawienie znacznika `TIM_SR_UIF` (i ewentualne wyzwolenie przerwania), o ile nie jest ustawiony bit `TIM_CR1_URS`
- Bit `TIM_EGR_UG` jest automatycznie zerowany sprzętowo

Inicjowanie licznika w skrócie

- Włączenie taktowania

```
RCC->APB1ENR |= RCC_APB1ENR_TIM3EN;
```

- Skonfigurowanie

```
TIM3->CR1 = TIM_CR1_URS | ...;
TIM3->PSC = ...;
TIM3->ARR = ...;
TIM3->EGR = TIM_EGR_UG;
```

- Opcjonalne nadanie wartości początkowej

```
TIM3->CNT = ...;
```

- Ewentualne włączenie przerwań – o tym za chwilę
- Uruchomienie

```
TIM3->CR1 |= TIM_CR1_CEN;
```

Buforowanie

- Bywa wygodne, bo umożliwia uzyskanie regularnych przebiegów bez konieczności ręcznej synchronizacji zmian rejestrów z aktualną wartością licznika
- Możemy w dowolnej chwili zmienić zakres zliczania

```
TIM3->PSC = ...;
TIM3->ARR = ...;
```

- Czy aby na pewno dobrze?
- Trzeba zablokować możliwość wystąpienie zdarzenia uaktualnienia pomiędzy przypisaniami
- Jak? Wyłączyć przerwania?
- To nie zadziała

Blokowanie zdarzeń uaktualnienia

- Ustawienie znacznika `TIM_CR1_UDIS` powoduje, że:
  - zdarzenia uaktualnienia nie są generowane
  - znacznik `TIM_SR_UIF` nie jest ustawiany
  - nie dochodzi do przerwań z powodu tego zdarzenia
- Przykładowa sekwencja

```
TIM3->CR1 |= TIM_CR1_UDIS;
TIM3->PSC = ...;
TIM3->ARR = ...;
TIM3->CR1 &= ~TIM_CR1_UDIS;
```

Rejestry sprzętowe liczników, cd.

- `TIMx->CCR1`, `TIMx->CCR2`, ...
  - przechowują wartości, z którymi może być porównywana wartość licznika
  - lub wartości przechwycone przez licznik
  - mogą być buforowane
  - ich liczba zależy od liczby kanałów konkretnego licznika (do czterech)
- `TIMx->CCMR1`, `TIMx->CCMR2`
  - konfigurują kanały wejściowe i wyjściowe
  - ich liczba zależy od konkretnego licznika (po dwa kanały w rejestrze)
- `TIMx->CCER`
  - decyduje, czy kanał wyjściowy licznika steruje zewnętrznym wyprowadzeniem

Zdarzenie zgodności

- Gdy wartość licznika jest równa wartości rejestru TIM*x*->CCR*y*, jest ustawiany znacznik TIM_SR_CC*y*IF i może być wyzwolone przerwanie
- W trybie zliczania w górę i w dół można określić, czy zdarzenie zgodności jest generowane podczas zliczania tylko w górę, tylko w dół, czy w obie strony
- Gdy licznik zlicza w górę lub w górę i w dół, a TIM*x*->CCR*y* jest większe niż TIM*x*->ARR, to znacznik jest ustawiany przy nadmiarze
- Gdy licznik zlicza w dół, a TIM*x*->CCR*y* jest większe niż TIM*x*->ARR, to znacznik jest ustawiany przy niedomiarze
- Znacznik TIM_SR_CC*y*IF zeruje się przez ustawienie go na 0

Przerwania

- Dla liczników `TIM2` do `TIM5` NVIC dostarcza przerwania o numerach `TIMx_IRQn`, gdzie $x = 2, 3, 4, 5$
- Licznik `TIM1` ma kilka przerwań, niektóre współdzieli z przerwaniami liczników `TIM9` do `TIM11`: `TIM1_BRK_TIM9_IRQHandler`, `TIM1_UP_TIM10_IRQHandler`, `TIM1_TRG_COM_TIM11_IRQHandler`, `TIM1_CC_IRQHandler`
- Przerwanie może nastąpić na skutek m.in. zdarzenia zgodności lub uaktualnienia

Obsługa przerwań

- Trzeba pamiętać o włączeniu przerwań
  - na poziomie licznika, przykładowo

    ```
    TIM3->SR = ~(TIM_SR_UIF | TIM_SR_CC1IF);
    TIM3->DIER = TIM_DIER_UIE | TIM_DIER_CC1IE;
    ```

  - na poziomie NVIC, przykładowo

    ```
    NVIC_EnableIRQ(TIM3_IRQn);
    ```

- Bity rejestru `TIMx->SR`
  - mogą być odczytywane
  - mogą być zerowane przez wpisanie 0
  - nie mogą być programowo ustawione na 1 — zapis 1 nie zmienia ich wartości
  - oznaczone jako zarezerwowanie nie powinny być modyfikowane

Szablon funkcji obsługi przerwań

```
void TIM3_IRQHandler(void) {
  uint32_t it_status = TIM3->SR & TIM3->DIER;
  if (it_status & TIM_SR_UIF) {
    TIM3->SR = ~TIM_SR_UIF;
    ...
  }
  if (it_status & TIM_SR_CC1IF) {
    TIM3->SR = ~TIM_SR_CC1IF;
    ...
  }
}
```

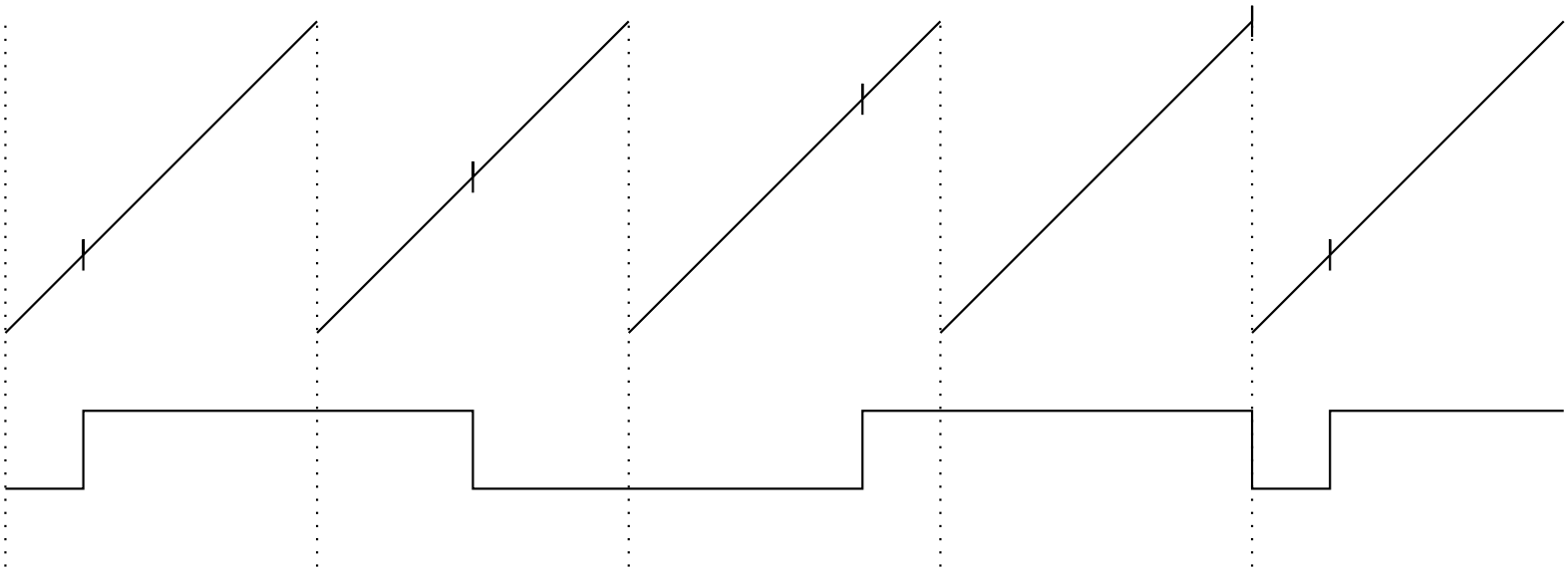
Generowanie przebiegów na wyjściu licznika

- Każdy kanał jest wyposażony w linię wyjściową `OCyREF`
- Stan linii wyjściowej zależy od wartości rejestru `TIMx->CCRy` i konfiguracji w rejestrach `TIMx->CCMR1` i `TIMx->CCMR2`
- Kanały 1 i 2 są konfigurowane w rejestrze `TIMx->CCMR1`, a kanały 3 i 4 – w rejestrze `TIMx->CCMR2`

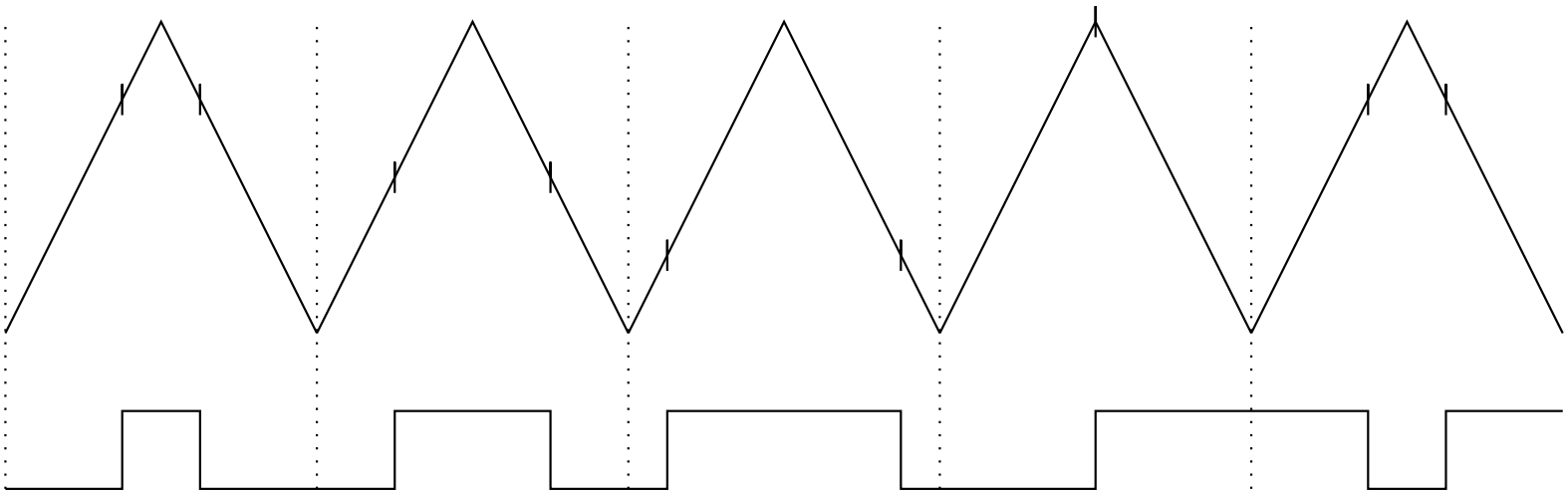
`OCyM`	stan linii `OCyREF`
000	stan nie zmienia się
001	aktywny po zdarzeniu zgodności
010	nieaktywny po zdarzeniu zgodności
011	zmiana stanu przy zdarzeniu zgodności
100	nieaktywny natychmiast
101	aktywny natychmiast
110	PWM 1
111	PWM 2

Przykładowe przebiegi

zmiana stanu przy zgodności, tryb zliczenia w górę



zmiana stanu przy zgodności, tryb zliczania w górę i w dół

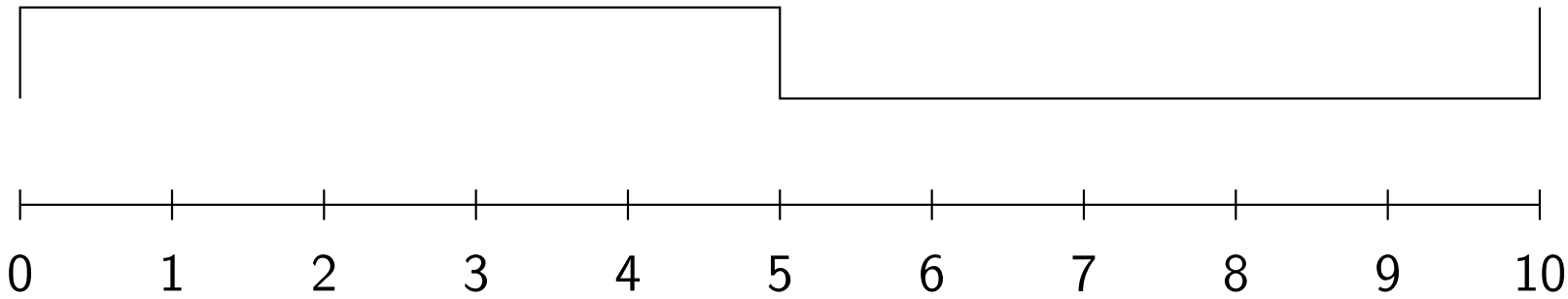


Pulse Width Modulation

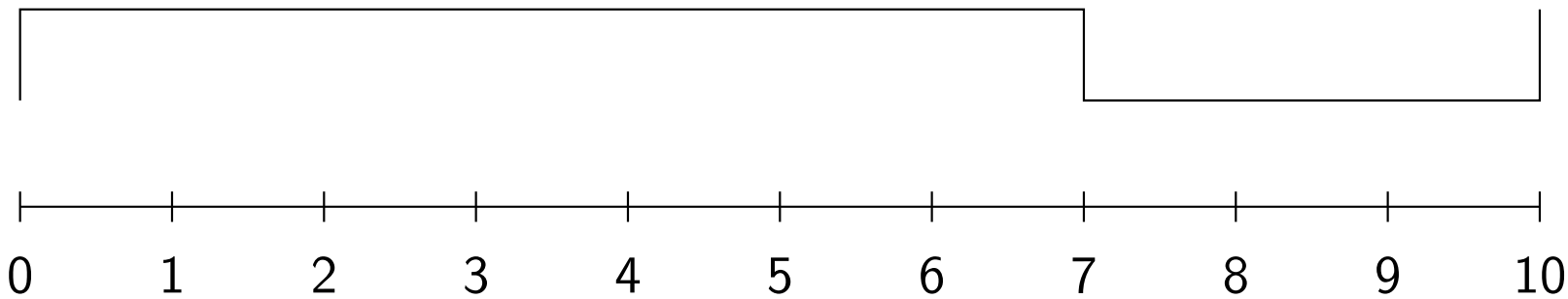
- Pulse Width Modulation
- Modulacja czasu trwania impulsów
- Ważna technika sterowania mocą odbiorników
- Zastosowania:
  - regulacja prędkości obrotowej silnika
  - regulacja jasności źródła światła
  - ustawianie kąta wychylenia serwomechanizmu
  - 1-bitowy przetwornik cyfrowo-analogowy
  - wzmacniacz akustyczny klasy D
  - ...

Czas trwania impulsu

- Sygnał o współczynniku wypełnienia 50%
- Czas trwania fazy wysokiej = czas trwania fazy niskiej sygnału



- Można też inaczej, np. współczynnik wypełnienia 70% oznacza, że poziom sygnału przez 70% czasu jest wysoki



Sterowanie jasnością diody

- Podajemy na diodę impulsy o częstotliwości od kilkudziesięciu do kilkuset Hz
- Jasność diody zależy (nieliniowo) od współczynnika wypełnienia
- Można próbować uzyskać liniowość jasności, zmieniając nieliniowo współczynnik wypełnienia

Przykłady innych zastosowań

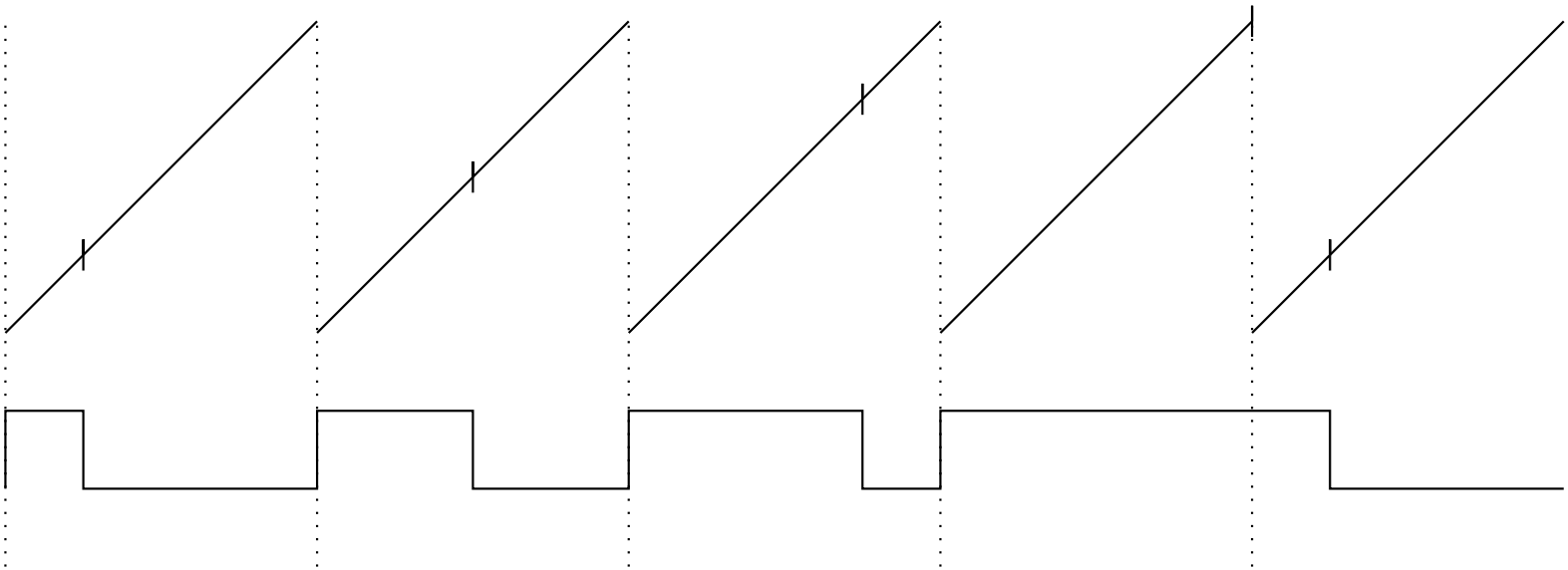
- Sterowanie kątem wychylenia serwa modelarskiego
  - kąty między -90 stopni a 90 stopni
  - kąt 0 stopni przy długości impulsu 1,5 ms
  - dopuszczalny zakres długości impulsu od 0,3 ms do 2,7 ms
  - częstotliwość powtarzania impulsów 50 Hz
- Sterowanie prędkością obrotową silnika
  - konieczny układ pośredniczący
  - gwarancja właściwego zabezpieczenia
  - możliwość regulacji prędkości obrotowej i kierunku obrotów
- Konwersja cyfrowo-analogowa
  - sygnał PWM podany na filtr dolnoprzepustowy
  - napięcie wyjściowe proporcjonalne do współczynnika wypełnienia
  - filtr dolnoprzepustowy jest niepotrzebny, jeśli membrana głośnika ma odpowiednio dużą bezwładność
  - wychylenie membrany jest proporcjonalne do współczynnika wypełnienia

Tryby PWM

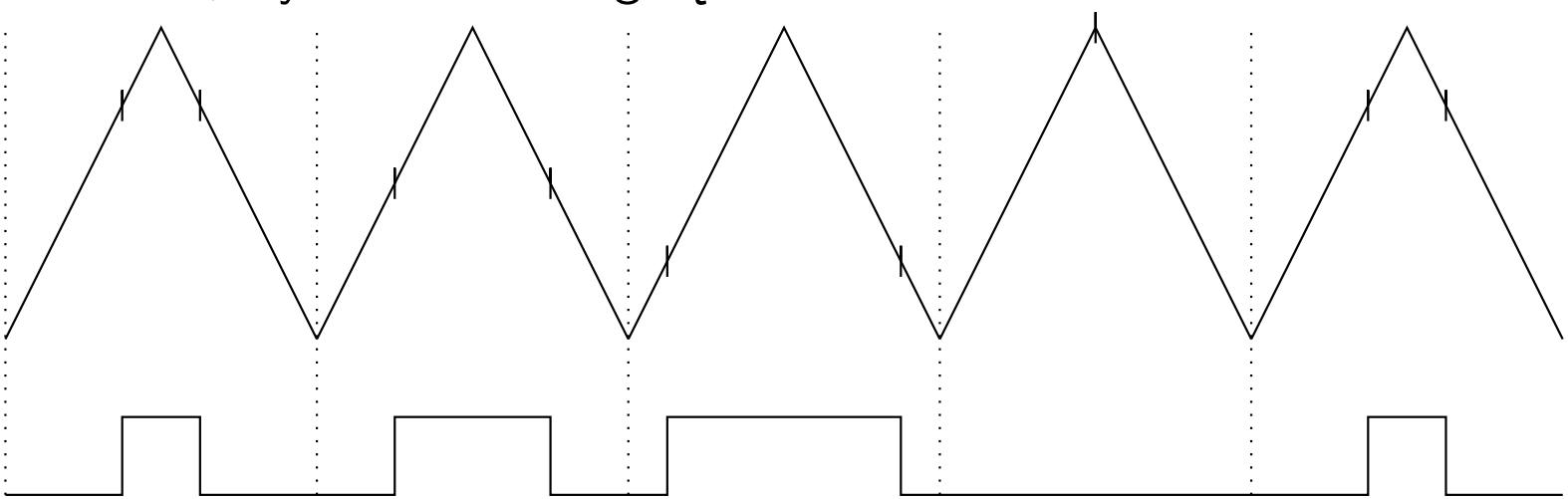
- Rejestr `TIMx->CCRy` musi być buforowany – włączony bit `TIM_CCMR1_OCyPE` (lub `TIM_CCMR2_OCyPE`)
- Rejestr `TIMx->CNT` jest porównywany z rejestrem `TIMx->CCRy`
- PWM 1
  - w trybie zliczenia w górę, linia jest w stanie aktywnym wtw., gdy `TIMx->CNT` $<$ `TIMx->CCRy`
  - w trybie zliczenia w dół, linia jest w stanie aktywnym wtw., gdy `TIMx->CNT` $>$ `TIMx->CCRy`
- PWM 2
  - w trybie zliczenia w górę, linia jest w stanie nieaktywnym wtw., gdy `TIMx->CNT` $<$ `TIMx->CCRy`
  - w trybie zliczenia w dół, linia jest w stanie nieaktywnym wtw., gdy `TIMx->CNT` $>$ `TIMx->CCRy`

Przykładowe przebiegi

PWM 1, tryb zliczenia w górę



PWM 2, tryb zliczenia w górę i w dół



Parametry generowanych przebiegów PWM

- W trybie zliczania w górę lub w dół:
  - częstotliwość

$$\frac{f_{\mathrm{CK\_CNT}x}}{\mathtt{TIM}x\mathtt{->ARR} + 1}$$

  - współczynnik wypełnienia

$$\frac{\mathtt{TIM}x\mathtt{->CCR}y}{\mathtt{TIM}x\mathtt{->ARR} + 1}$$

- W trybie zliczania w górę i w dół:
  - częstotliwość

$$\frac{f_{\mathrm{CK\_CNT}x}}{2 \cdot \mathtt{TIM}x\mathtt{->ARR}}$$

  - współczynnik wypełnienia

$$\frac{\mathtt{TIM}x\mathtt{->CCR}y}{\mathtt{TIM}x\mathtt{->ARR}}$$

Podłączanie wyjścia licznika do wyprowadzenia

- Linia wyjściowa `OCyREF` licznika `TIMx` może zostać podłączona do wyprowadzenia `TIMx_CHy` mikrokontrolera
- Na przykład
  - `TIM3_CH1` to wyprowadzenie `PA6` skonfigurowane jako funkcja alternatywna (dioda czerwona w ekspanderze)
  - `TIM3_CH2` to wyprowadzenie `PA7` skonfigurowane jako funkcja alternatywna (dioda zielona w ekspanderze)
  - `TIM3_CH3` to wyprowadzenie `PB0` skonfigurowane jako funkcja alternatywna numer (dioda niebieska w ekspanderze)
- Żeby podłączyć linię wyjściową do odpowiedniego wyprowadzenia, należy:
  - skonfigurować wyprowadzenie
  - włączyć bit `TIM_CCER_CCyE`
  - ustawić bit `TIM_CCER_CCyP`, jeśli stan aktywny linii wyjściowej ma odpowiadać stanowi niskiemu wyprowadzenia

Przykład

- Włączamy taktowanie

```
RCC->AHB1ENR |= RCC_AHB1ENR_GPIOAEN;
RCC->APB1ENR |= RCC_APB1ENR_TIM3EN;
```

- Konfigurujemy wyprowadzenia

```
GPIOafConfigure(GPIOA, 6, GPIO_OType_PP,
                GPIO_Low_Speed,
                GPIO_PuPd_NOPULL, GPIO_AF_TIM3);
GPIOafConfigure(GPIOA, 7, GPIO_OType_PP,
                GPIO_Low_Speed,
                GPIO_PuPd_NOPULL, GPIO_AF_TIM3);
```

- Ustawiamy parametry licznika

```
TIM3->PSC = 63999;
TIM3->ARR = 749;
TIM3->EGR = TIM_EGR_UG;
TIM3->CCR1 = 249;
TIM3->CCR2 = 499;
```

Przykład, cd.

- Licznik będzie zliczał w górę; konfigurujemy linię wyjściową `OC1REF` w trybie PWM 1, a linię `OC2REF` – w trybie PWM 2; rejestry `TIM3->CCR1` i `TIM3->CCR2` są buforowane

```
TIM3->CCMR1 =
  TIM_CCMR1_OC1M_2 | TIM_CCMR1_OC1M_1 |
  TIM_CCMR1_OC1PE  |
  TIM_CCMR1_OC2M_2 | TIM_CCMR1_OC2M_1 |
  TIM_CCMR1_OC2M_0 | TIM_CCMR1_OC2PE;
```

- Podłączamy linie wyjściowe do wyprowadzeń, stan aktywny to stan niski, bo diodę włączamy stanem niskim

```
TIM3->CCER = TIM_CCER_CC1E | TIM_CCER_CC1P |
             TIM_CCER_CC2E | TIM_CCER_CC2P;
```

- Włączamy licznik w trybie zliczania w górę z buforowaniem rejestru `TIM3->ARR`

```
TIM3->CR1 = TIM_CR1_ARPE | TIM_CR1_CEN;
```

- Ćwiczenie: korzystając z przerwań zgodności, włączaj niebieską diodę w przerwie między czerwoną a zieloną