# Programowanie mikrokontrolerów 2.0

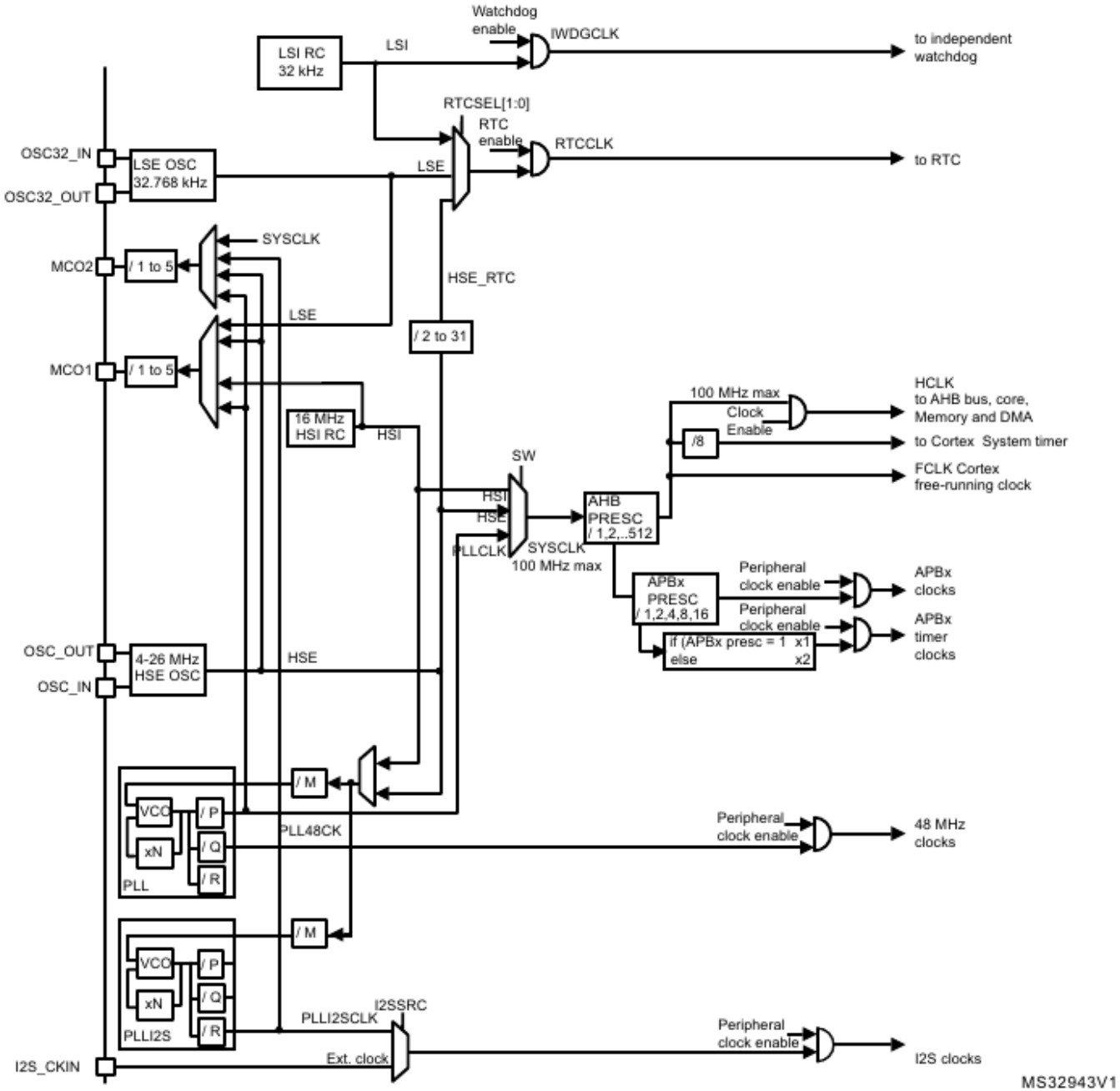
## Taktowanie

Marcin Engel  Marcin Peczarski

Instytut Informatyki Uniwersytetu Warszawskiego

16 listopada 2021

# Drzewo taktowania w STM32F411




Źródło: RM0383 – Reference manual, STM32F411xC/E advanced ARM-based 32-bit MCUs, Figure 12. Clock tree

# Drzewo taktowania w STM32F411

- Cztery oscylatory: LSI, LSE, HSI, HSE
- Dwie pętle fazowe: PLL, PLLI2S
- Kilkanaście wewnętrznych sygnałów zegarowych
- Dwa wyjścia sygnałów zegarowych: MCO1, MCO2
- Jedno wejście sygnału zegarowego: I2S_CKIN
- Kilka układów wybierających jeden z kilku sygnałów zegarowych
- Mnóstwo dzielników częstotliwości (ang. *prescaler*)
- Niezależne włączanie sygnału zegarowego dla poszczególnych peryferii
- Uwaga: jeśli nie zaznaczono tego jawnie, to podane na kolejnych slajdach wartości liczbowe dotyczą tego modelu mikrokontrolera

# LSI (ang. *Low Speed Internal*)

- Wewnętrzny generator RC małej częstotliwości

$$f_{\mathrm{LSI}} = 32 \pm 15 \text{ kHz}$$

- Taktowanie strażnika (ang. *watchdog*)
- Taktowanie zegara czasu rzeczywistego RTC (ang. *Real Time Clock*) – niezalecane

# LSE (ang. *Low Speed External*)

- Generator kwarcowy małej częstotliwości

$$f_{\mathrm{LSE}} = 32768 \text{ Hz}$$

- Wymaga podłączenia „zegarkowego” rezonatora kwarcowego
- Dokładność wynika z parametrów zastosowanego rezonatora
- Taktowanie zegara czasu rzeczywistego RTC
- Zamiast rezonatora kwarcowego można podłączyć zewnętrzne źródło sygnału zegarowego

# HSI (ang. *High Speed Internal*)

- Wewnętrzny generator RC dużej częstotliwości

$$f_{\mathrm{HSI}} = 16 \text{ MHz}$$

- Dokładność zależna od zakresu temperatur pracy: od $\pm 1\%$ do $\pm 8\%$
- Możliwość programowej kalibracji
- Taktowanie większości układów mikrokontrolera, z wyjątkiem strażnika i zegara czasu rzeczywistego

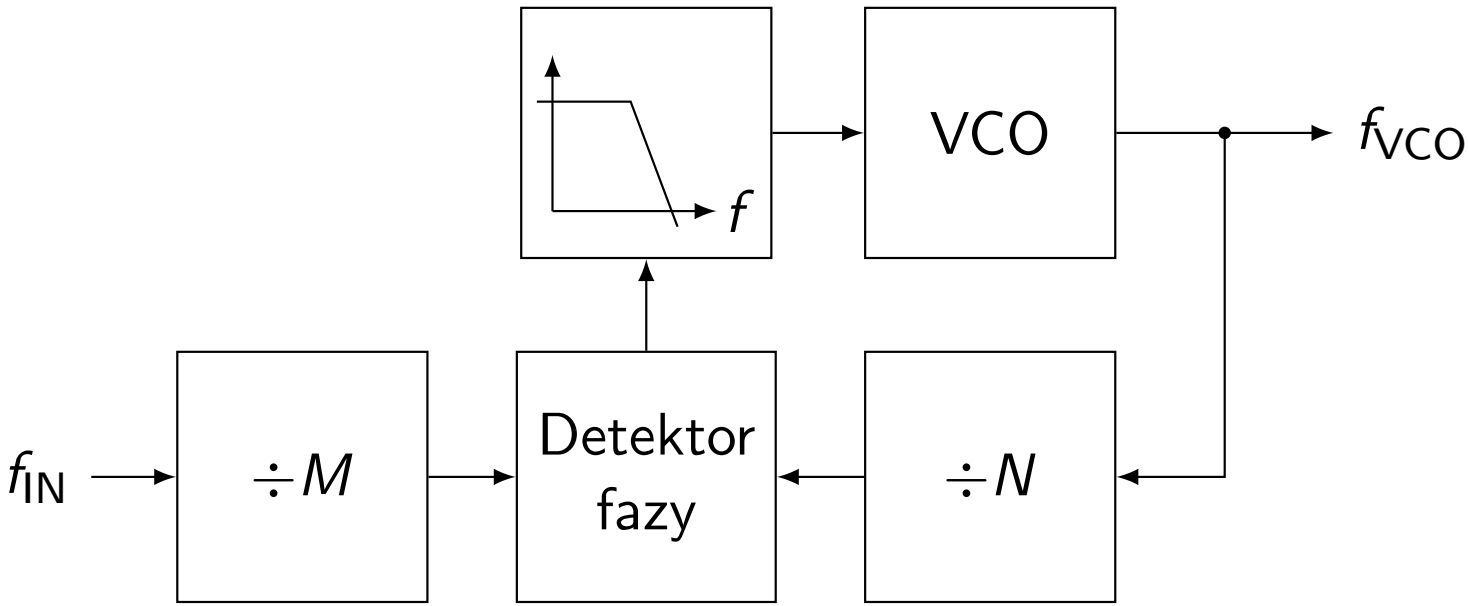
# HSE (ang. *High Speed External*)

- Generator kwarcowy dużej częstotliwości

$$4\ \text{MHz} \leqslant f_{\text{HSE}} \leqslant 26\ \text{MHz}$$

- Dokładność wynika z parametrów zastosowanego rezonatora
- Zamiast rezonatora kwarcowego można podłączyć zewnętrzne źródło sygnału zegarowego
- Taktowanie większości układów mikrokontrolera, z wyjątkiem strażnika

# Pętla fazowa (ang. *Phase Locked Loop*)




$$f_{VCO} = \frac{N}{M} \cdot f_{IN}$$

# PLL w STM32F411

- Oscylator pętli fazowej wytwarza sygnał zegarowy o częstotliwości

$$f_{\mathrm{VCO}} = \frac{N}{M} \cdot f_{\mathrm{PLL\_IN}}$$

- gdzie

$$f_{\mathrm{PLL\_IN}} = f_{\mathrm{HSI}} \quad \text{lub} \quad f_{\mathrm{PLL\_IN}} = f_{\mathrm{HSE}}$$

- Sygnał tego oscylatora służy do wytworzenia sygnału wyjściowego

$$f_{\mathrm{PLL\_OUT}} = f_{\mathrm{VCO}}/P$$

- oraz sygnału zegarowego 48 MHz dla interfejsu USB

$$f_{\mathrm{PLL48CK}} = f_{\mathrm{VCO}}/Q$$

# PLL w STM32F411, cd.

- Ograniczenia częstotliwości

$$
\begin{aligned}
1\ \text{MHz} \leqslant\ & f_{\text{PLL\_IN}}/M & \leqslant 2\ \text{MHz} \\
100\ \text{MHz} \leqslant\ & f_{\text{VCO}} & \leqslant 432\ \text{MHz} \\
24\ \text{MHz} \leqslant\ & f_{\text{PLL\_OUT}} & \leqslant 100\ \text{MHz}
\end{aligned}
$$

- Ograniczenia wartości współczynników

$$
\begin{aligned}
M &= 2, 3, 4, \ldots, 63 \\
N &= 50, 51, 52, \ldots, 432 \\
P &= 2, 4, 6, 8 \\
Q &= 2, 3, 4, \ldots, 15
\end{aligned}
$$

# Pytania kontrolne

- Czy można uzyskać jednocześnie $f_{\mathrm{PLL48CK}} = 48$ MHz i $f_{\mathrm{PLL\_OUT}} = 100$ MHz?
- Dlaczego możemy chcieć taktować mikrokontroler z częstotliwością mniejszą niż maksymalna dopuszczalna?

# PLLI2S w STM32F411

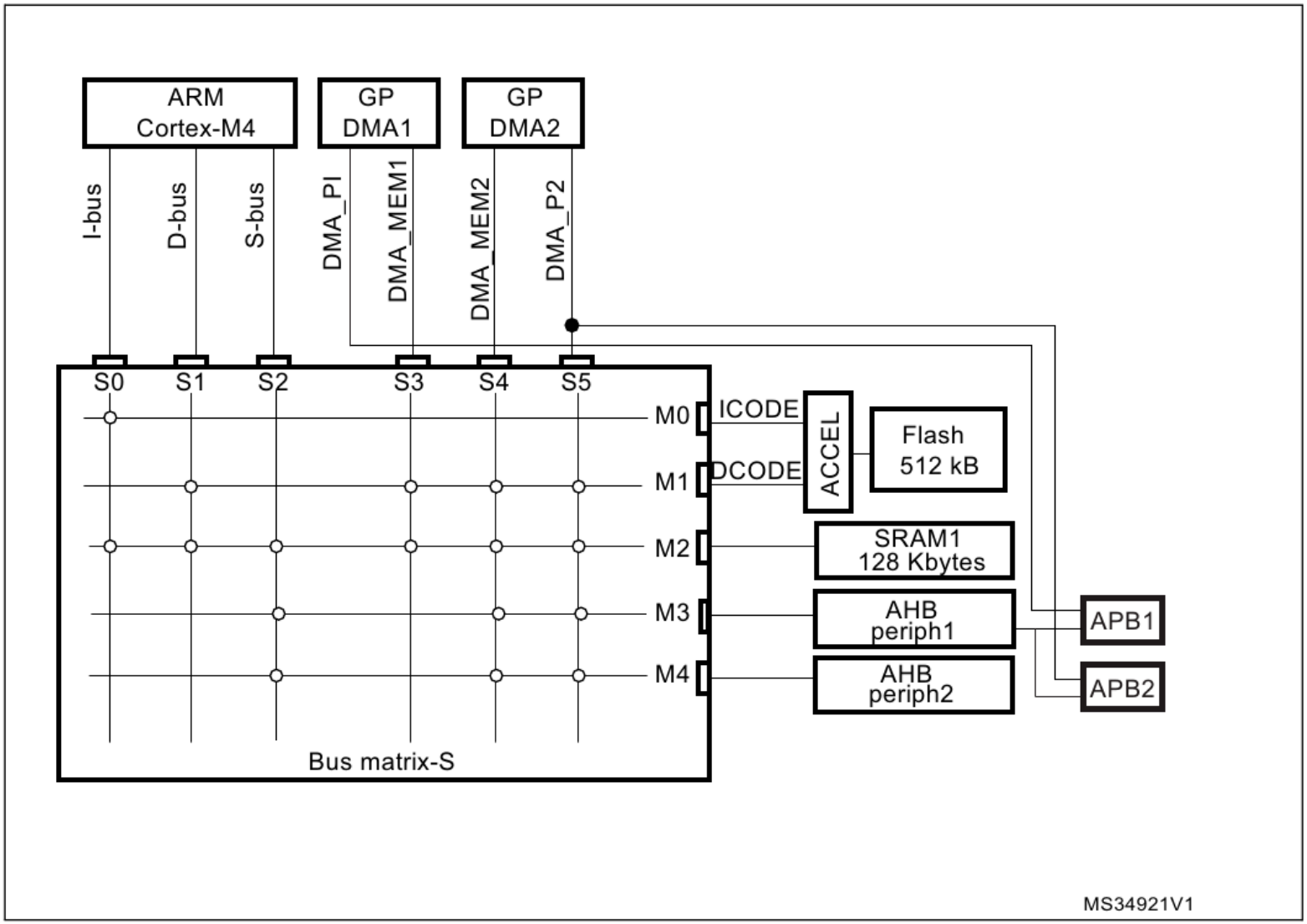
- Pętla fazowa wytwarzająca sygnał zegarowy do taktowania interfejsu I2S

$$f_{\text{PLLI2SCLK}} = \frac{N}{M \cdot R} \cdot f_{\text{PLL\_IN}}$$

- To samo źródło sygnału wejściowego $f_{\text{PLL\_IN}}$
- Niezależne współczynniki
- Podobne ograniczenia jak dla głównej PLL

# Szyny w STM32F411

**Figure 4. Multi-AHB matrix**




Źródło: STM32F411xC/E Data sheet

# Dystrybucja sygnałów zegarowych w STM32F411

- Główny sygnał zegarowy

$$f_{\mathrm{SYSCLK}} \quad = \quad f_{\mathrm{HSI}} \quad \text{lub} \quad f_{\mathrm{HSE}} \quad \text{lub} \quad f_{\mathrm{PLL\_OUT}}$$

- Służy do wytworzenia sygnałów taktujących rdzeń, pamięci, DMA oraz szyn AHB1 (GPIO, rejestry konfiguracyjne DMA) i AHB2 (USB)

$$f_{\mathrm{HCLK}} = f_{\mathrm{SYSCLK}} / HPRE$$

- gdzie

$$HPRE = 1, 2, 4, 8, 16, 64, 128, 256, 512$$

- Ograniczenie

$$f_{\mathrm{HCLK}} \leqslant 100 \text{ MHz}$$

- Do szyny AHB1 podłączone są szyny APB1 i APB2, do których podłączone są pozostałe peryferie

# Dystrybucja sygnałów zegarowych w STM32F411

- Szyna APB1, do której podłączone są „wolne” peryferie: `TIM2`, `TIM3`, `TIM4`, `TIM5`, `WWDG`, `SPI2`, `SPI3`, `USART2`, `I2C1`, `I2C2`, `I2C3`, ...

$$f_{\mathrm{PCLK1}} = f_{\mathrm{HCLK}}/\mathit{PPRE1}, \quad \text{gdzie} \quad \mathit{PPRE1} = 1, 2, 4, 8, 16$$

- Ograniczenie

$$f_{\mathrm{PCLK1}} \leqslant 50\ \mathrm{MHz}$$

- Szyna APB2, do której podłączone są „szybkie” peryferie: `TIM1`, `TIM9`, `TIM10`, `TIM11`, `USART1`, `USART6`, `ADC1`, `SDIO`, `SPI1`, `SPI4`, `SPI5`, `SYSCFG`, ...

$$f_{\mathrm{PCLK2}} = f_{\mathrm{HCLK}}/\mathit{PPRE2}, \quad \text{gdzie} \quad \mathit{PPRE2} = 1, 2, 4, 8, 16$$

- Ograniczenie

$$f_{\mathrm{PCLK2}} \leqslant 100\ \mathrm{MHz}$$

# Dystrybucja sygnałów zegarowych w STM32F411

- Liczniki (`TIMx`) podłączone do szyny APB$y$ taktowane są zegarem o częstotliwości

$$f_{\mathrm{PCLK}y}, \quad \text{gdy} \quad PPREy = 1$$

$$2f_{\mathrm{PCLK}y}, \quad \text{gdy} \quad PPREy > 1$$

gdzie $x = 1, 2, 3, \ldots$, $y = 1, 2$

- Uwaga: w większości modeli STM dystrybucja sygnałów zegarowych jest podobna do wyżej opisanej, ale występują drobne acz znaczące różnice – trzeba czytać dokumentację

# Maksymalne częstotliwości taktowania w wybranych modelach STM32

| [MHz] | L0xx L1xx | F0xx | F10x F3xx[1] | F2xx | L4xx | L4Rx L4Sx |
|---|---|---|---|---|---|---|
| $f_{PCLK1}$ | 32 | 48 | 36 | 30 | 40 | 120 |
| $f_{PCLK2}$ | 32 | 48 | 72 | 60 | 80 | 120 |
| $f_{HCLK}$ | 32 | 48 | 72 | 120 | 80 | 120 |

| [MHz] | F401 | F410 F411 | F405/415 F407/417 | F427/437 F429/439 F446 F469/479 | F7xx |
|---|---|---|---|---|---|
| $f_{PCLK1}$ | 42 | 50 | 42 | 45 | 54 |
| $f_{PCLK2}$ | 84 | 100 | 84 | 90 | 108 |
| $f_{HCLK}$ | 84 | 100 | 168 | 180 | 216 |

[1] Niektóre liczniki mogą być taktowane z częstotliwością 144 MHz

# Po wyzerowaniu

- Po włączeniu zasilania lub sprzętowym albo programowym wyzerowaniu mikrokontrolera

$$f_{\mathrm{HCLK}} = f_{\mathrm{PCLK1}} = f_{\mathrm{PCLK2}} = f_{\mathrm{SYSCLK}} = f_{\mathrm{HSI}}$$

# Przykład konfiguracji HSE

- Konfigurowanie trzeba rozpocząć z wyłączonym HSE i wyłączonymi pętlami fazowymi

```
RCC->CR &= ~(RCC_CR_PLLI2SON | RCC_CR_PLLON |
             RCC_CR_HSEBYP | RCC_CR_HSEON);
```

- Włączamy oscylator HSE z rezonatorem kwarcowym

```
RCC->CR |= RCC_CR_HSEON;
```

- Albo włączamy zewnętrzny generator HSE

```
RCC->CR |= RCC_CR_HSEBYP | RCC_CR_HSEON;
```

- Czekamy, aż oscylator „zaskoczy”, czyli aż będzie spełniony warunek

```
RCC->CR & RCC_CR_HSERDY
```

- Ale nie czekamy w nieskończoność
- Jak to zrobić?
- Co zrobić, gdy oscylator nie „zaskoczy”?

# Przykład konfiguracji PLL

- Konfigurujemy sygnał wejściowy i parametry pętli fazowej

```
RCC->PLLCFGR = RCC_PLLCFGR_PLLSRC_HSE |
  M | N << 6 | ((P >> 1) - 1) << 16 | Q << 24;
```

- Włączamy oscylator pętli

```
RCC->CR |= RCC_CR_PLLON;
```

- Czekamy na zsynchronizowanie się pętli, czyli aż będzie spełniony warunek

```
RCC->CR & RCC_CR_PLLRDY
```

- Ale oczywiście nie czekamy w nieskończoność
- Co zrobić, gdy pętla się nie zsynchronizuje?

# Przykład konfiguracji Flash

- Pamięć Flash jest za wolna
- Trzeba skonfigurować dodatkowe takty oczekiwania (ang. *latency*)
- Żeby nie spowalniać pracy mikrokontrolera, trzeba włączyć pmięci podręczne

```
FLASH->ACR = FLASH_ACR_DCEN | /* data cache */
             FLASH_ACR_ICEN | /* instr. cache */
             FLASH_ACR_PRFTEN | /* prefetch */
             latency;
```

- Wartość parametru `latency` zależy od napięcia zasilania
  - zakładamy, że jest większe niż 2,7 V
  - dla innych napięć trzeba sprawdzić w dokumentacji

$$\text{latency} = \left\lfloor \frac{f_{\text{HCLK}}\ [\text{Hz}]}{30 \cdot 10^6} \right\rfloor$$

# Przykład konfiguracji, taktowanie szyn

- Zmienna pomocnicza

```
uint32_t reg;
```

- Konfigurujemy podzielnik dla $f_{\mathrm{HCLK}}$

```
reg = RCC->CFGR;
reg &= ~RCC_CFGR_HPRE;
reg |= RCC_CFGR_HPRE_DIV1;
RCC->CFGR = reg;
```

- Konfigurujemy podzielnik dla $f_{\mathrm{PCLK1}}$

```
reg = RCC->CFGR;
reg &= ~RCC_CFGR_PPRE1;
reg |= RCC_CFGR_PPRE1_DIV2;
RCC->CFGR = reg;
```

- Konfigurujemy podzielnik dla $f_{\mathrm{PCLK2}}$

```
reg = RCC->CFGR;
reg &= ~RCC_CFGR_PPRE2;
reg |= RCC_CFGR_PPRE2_DIV1;
RCC->CFGR = reg;
```

# Przykład konfiguracji, włączenie całości

- Ustawiamy $f_{\mathrm{SYSCLK}} = f_{\mathrm{PLL\_OUT}}$

```
reg = RCC->CFGR;
reg &= ~RCC_CFGR_SW;
reg |= RCC_CFGR_SW_PLL;
RCC->CFGR = reg;
```

- To chwilę trwa, czekamy na spełnienie warunku

```
(RCC->CFGR & RCC_CFGR_SWS) == RCC_CFGR_SWS_PLL
```

- Ale oczywiście nie czekamy w nieskończoność

# Podsumowanie

- Mikrokontroler w zestawie laboratoryjnym może być taktowany
  - wewnętrznym generatorem RC HSI o częstotliwości 16 MHz
  - zewnętrznym generatorem kwarcowym HSE o częstotliwości 8 MHz – źródłem sygnału jest ST-LINK/V2-1
- Za pomocą PLL można z tych sygnałów zegarowych wytworzyć sygnały taktujące o potrzebnych częstotliwościach

# Włączanie taktowania peryferii, przypomnienie i podsumowanie

- Peryferie podłączone do szyny AHB1 (DMA1, DMA2, CRC, GPIOx) włączamy za pomocą rejestru AHB1ENR, np.

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

- Peryferie podłączone do szyny AHB2 (USB) włączamy za pomocą rejestru AHB2ENR, np.

  ```
  RCC->AHB2ENR |= RCC_AHB2ENR_OTGFSEN;
  ```

- Peryferie podłączone do szyny APB1 (TIM2, TIM3, TIM4, TIM5, WWDG, SPI2, SPI3, USART2, I2C1, I2C2, I2C3) włączamy za pomocą rejestru APB1ENR, np.

  ```
  RCC->APB1ENR |= RCC_APB1ENR_USART2EN;
  ```

- Peryferie podłączone do szyny APB2 (TIM1, TIM9, TIM10, TIM11, USART1, USART6, ADC1, SDIO, SPI1, SPI4, SPI5, SYSCFG) włączamy za pomocą rejestru APB2ENR, np.

  ```
  RCC->APB2ENR |= RCC_APB2ENR_USART1EN;
  ```