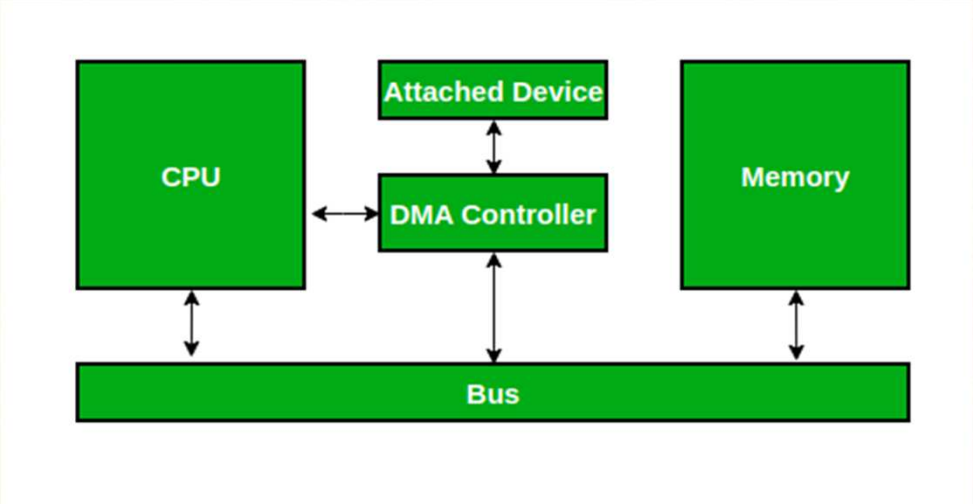
PŘÍMÝ PŘÍSTUP DO PAMĚTI DMA - OBECNĚ

Přenos dat realizovaný procesorem bude minimálně zatížen čtením, dekódováním a vykonáním instrukcí pro čtení z V/V, uložení hodnoty do RAM, výpočtem nové adresy a počtem přenášených hodnot.

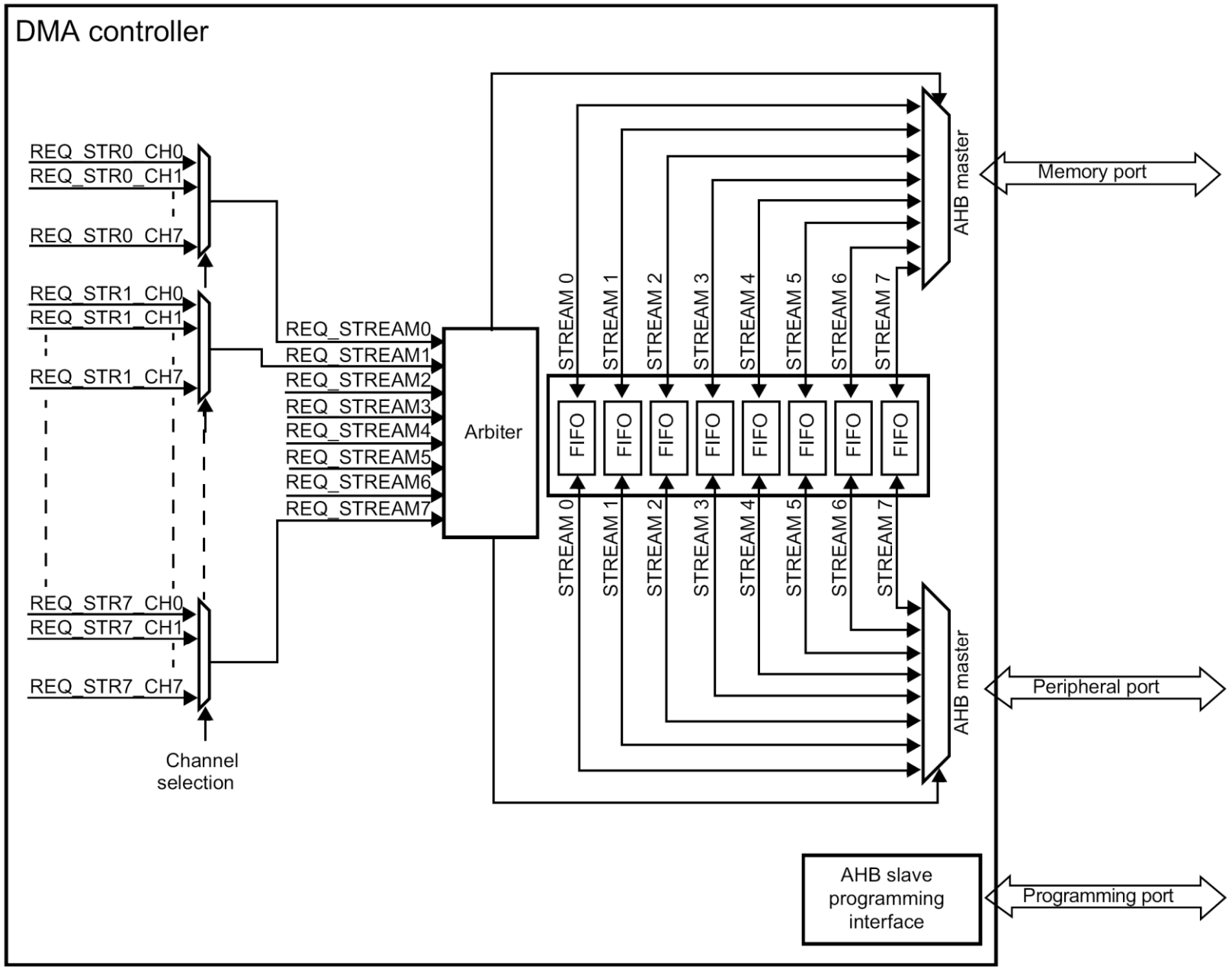
- DMA zařizuje přenos bez účasti procesoru pomocí hardwarové podpory střídající pouze operaci čtení dat z V/V zařízení a zápisu na adresu paměti RAM s její následnou inkrementací.
- Uvedené funkce zajišťuje naprogramovaný řadič DMA.
- Řadič DMA je logický sekvenční obvod, generující sekvenci pro čtení V/V zařízení, její dočasné uložení a sekvenci pro zápis do RAM následované inkrementací adresy zápisu a dekrementací počtu přenášených hodnot.

- **Metody DMA**
  - Zastavením procesoru z důvodu uvolnění sběrnic procesoru pro přenos řízený DMA
  - Zastavování cyklů procesoru nebo nezávislý přenos DMA
  - Sdílení paměti - přenos v části strojového cyklu, kdy procesor nekomunikuje po sběrnici
  - Vytvořením speciálních sběrnic pro tyto přenosy bez omezení činnosti procesoru



- **Výhody a nevýhody**
  - Velká rychlost přenosu hodnot dosažená odstraněním neproduktivních činností spojených se čtením a zpracováním instrukcí vyjma jejich výkonných fází.
  - Data při přenosu nelze nijak upravovat
  - Zpomalení nebo zastavení činnosti procesoru při uvolňování sběrnice do stavu vysoké impedance při některých metodách DMA
  - Nutnost dalšího technického vybavení

VLASTNOSTI DMA NA PROCESORU M4



VLASTNOSTI DMA NA PROCESORU M4

DMA – umožňuje rychlý přenos hodnot mezi periferií a pamětí nebo paměti a pamětí bez účasti CPU.

- Architektura sběrnice AHB je zdvojena, jedna pro zajištění přístupu do paměti a druhá pro přístup do periferií. V procesoru jsou dva DMA kontroléry, každý disponuje 8 streamy s až 8 požadavky (kanály) přenosu.
- Každý stream disponuje čtyř úrovňovou FIFO pamětí o šířce 32 bitů. Řadič DMA zajišťuje skládáním a rozdělováním slov při nestejné šířce slova zdroje a cíle s programovatelnou prahovou úrovní ¼, ½ a ¾. Přenos může být realizován přímo nebo přes FIFO.
- Priority mezi jednotlivými požadavky (kanály) streamu DMA jsou programovatelné do čtyřech úrovní (velmi vysoké, vysoké, střední, nízké).

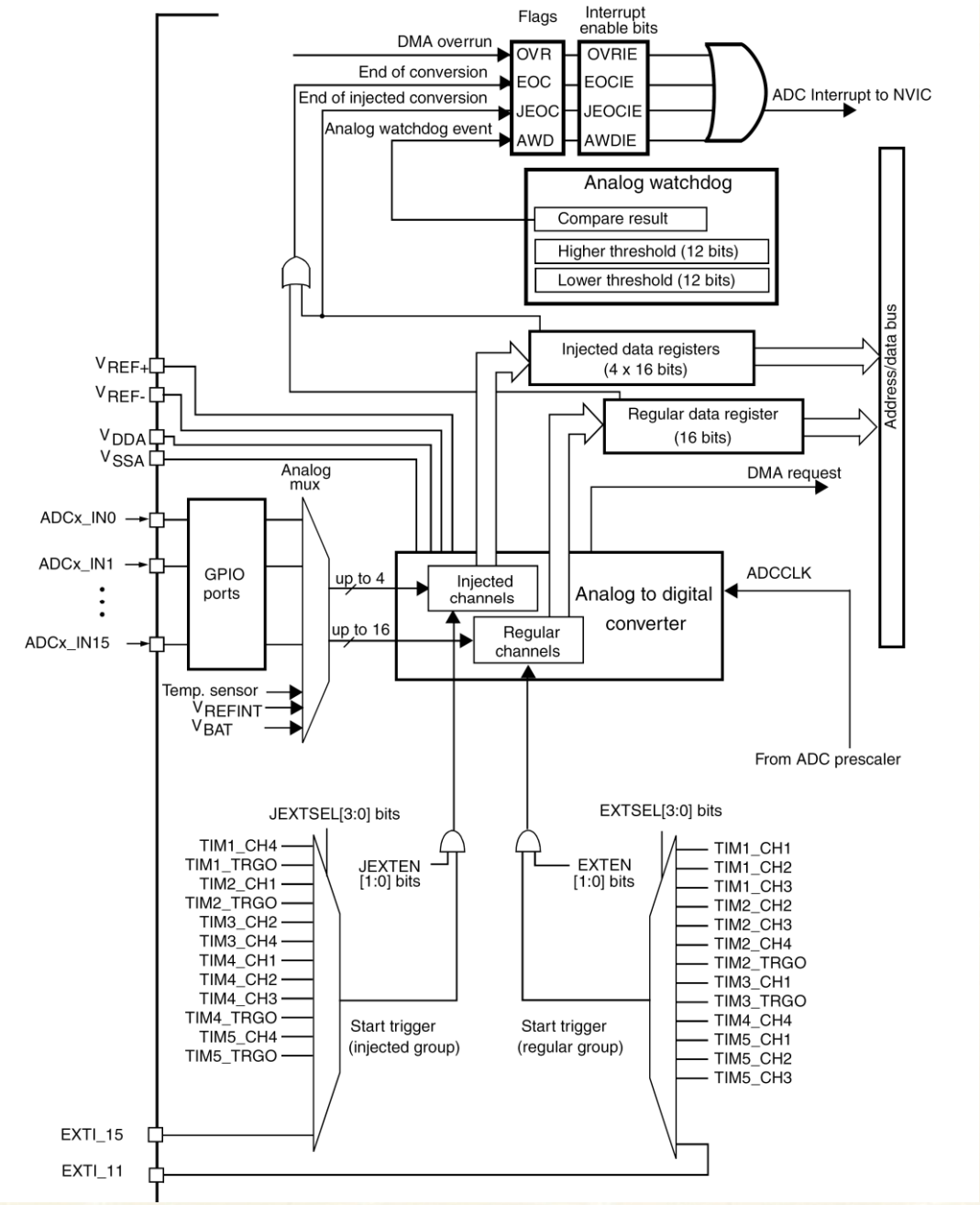
VLASTNOSTI DMA NA PROCESORU M4

- V případě rovnosti priority u jednotlivých požadavků (kanálů), pak kanál 0 má přednost před kanálem 1 atd.).
- Každý stream podporuje práci s kruhovým buffrem.
- DMA generuje 5 příznaků
  - DMA Half Transfer
  - DMA Transfer complete
  - DMA Transfer Error
  - DMA FIFO Error
  - Direct Mode Error

INICIALIZACE PŘEVODNÍKU PRO JOYSTICK HORIZONTÁLNÍ SMĚR - PA4

```
//   DMA Stream 0 kanál 0 bude realizovat přenos periferie(A/D)-paměť RAM (ADC_Buffer)

void Inicializace_DMA(void){
     RCC->AHB1ENR |= RCC_AHB1ENR_DMA2EN;          // Povoleni hodin pro DMA2
//     ADC1->CR2|= ADC_CR2_DMA;                     // DMA mód povolen v inicializaci ADC1
     DMA2_Stream0->CR &= ~DMA_SxCR_EN;            // Zakázání Streamu
     while((DMA2_Stream0->CR &0x0001)!=0) ;       // Čekej na ukončeni DMA2_Stream0

     DMA2_Stream0->PAR = (uint32_t)&ADC1->DR;     // Nastaveni adresy DR registru ADC1
     DMA2_Stream0->M0AR = (uint32_t)&ADC_Buffer;  // Nastaveni adresy ADC_Buffer

     DMA2_Stream0->CR |= DMA_SxCR_CIRC;           // Povolení kruhového módu (kruhový bufer)
     DMA2_Stream0->CR |= DMA_SxCR_MINC;           // Adresa do buffru je inkrementována
     DMA2_Stream0->CR |= DMA_SxCR_PSIZE_0;        // Z ADR1 se přenáší se slova o 16 bitech
                                                  // (half word)
     DMA2_Stream0->CR |= DMA_SxCR_MSIZE_0;        // Do paměti se přenáší slova o 16 bitech
     DMA2_Stream0->CR &= ~DMA_SxCR_CHSEL;         // Přenos kanálem 0
     DMA2_Stream0->NDTR |= 16;                    // Přenáší se 16 hodnot
     DMA2_Stream0->CR |= DMA_SxCR_EN;             // Zapnutí DMA2 Stream0 kanál 0
}

void DMA2_Stream0_IRQHandler() {
     if ((DMA2->LISR & DMA_LISR_TCIF0 )!=0)
     {   DMA2->LIFCR |= DMA_LIFCR_CTCIF0;          // Nuluj návěští naplnění bufferu
         DMA2->LIFCR |= DMA_LIFCR_CHTIF0;          // Nuluj flag naplněni poloviny bufferu

//   Nutno znovu obnovit přenos přes DMA, pokud nebylo použito ADC1->CR2|= ADC_CR2_DDS;
         ADC1->CR2&= ~ADC_CR2_DMA;                 // DMA mód zakázán
         ADC1->CR2|= ADC_CR2_DMA;                  // DMA mód povolen - Nové měření
                                                   // Nemusí být umístěno v přerušení
     }
}
```

A/D PŘEVODNÍK NASTAVENÍ



- Hodinový signál pro A/D převodník je vyvedený přímo z oscilátoru HSI o kmitočtu 16MHz. Povolený hodinový signál pro převodník, může být vydělen hodnotou 1, 2 nebo 4 v registru **ADC_CCR** bity **ADCPRE.**

- **Převodník** může být spuštěn programově nebo obvodově (viz. Obrázek) z vpravo dole uvedených periferií.

INICIALIZACE PŘEVODNÍKU PRO JOYSTICK HORIZONTÁLNÍ SMĚR - PA4

```
//   A/D převodník bude spouštěn signálem TRGO

void Inicializace_ADC(void){
     RCC->APB2ENR |= RCC_APB2ENR_ADC1EN;       // Povoleni hodin pro ADC1
//     ADC->CCR |= ADC_CCR_TSVREFE;              // Povoleni teplotního senzoru a Uref
     ADC->CCR |= ADC_CCR_ADCPRE_1;             // Předdělič ADC = HSI/4
     ADC1->CR2 |= ADC_CR2_EXTEN_0;             // Spouštění převodu na náběžnou hranu
     ADC1->CR2 |= ADC_CR2_EXTSEL_2|ADC_CR2_EXTSEL_1;  // ADC spouštěn událostí TIM2 TRGO
     ADC1->SQR3 &= ~ADC_SQR1_L;                // Jeden ADC převod - nuluj L
//     ADC1->SQR1 |= ADC_SQR1_L_0;               // Dva převody
     ADC1->SQR3 &= ~ADC_SQR3_SQ1;              // Smazání všech kanálu pro první převod
     ADC1->SQR3 |= ADC_SQR3_SQ1_2;             // První převod z kanálu 4 (tj. PA_4)
     ADC1->SMPR2 |= ADC_SMPR2_SMP4;            // Vzorkování kanálu 4 - 480 cyklů
     ADC1->CR1 &= ~ADC_CR1_RES;                // 12 bitová konfigurace (Tcon=15 ADCCLK cyklů)
     ADC1->CR2 &= ~ADC_CR2_ALIGN;              // Zarovnání doprava
//     ADC1->CR1|= ADC_CR1_SCAN;                 // Nastavení skenovacího režimu
     ADC1->CR2|= ADC_CR2_DMA;                  // Povolení DMA módu
//     ADC1->CR2|= ADC_CR2_DDS;                  // Přenos DMA po každém ADC převodu
                                               // Nastavením můžeme zajistit kontinuální
                                               // přenos naměřených hodnot od počátečního
                                               // spuštění
     ADC1->CR2|= ADC_CR2_EOCS;                 // Nastaveni EOC po každém převodu
     ADC1->CR2 |= ADC_CR2_ADON;                // Spuštění ADC převodníku
}
```

INICIALIZACE ČASOVAČE PRO SPOUŠTĚNÍ A/D PŘEVODNÍKU

```
//   TIM2 je připojen ke sběrnici APB1, nastavenou na kmitočet 24MHz

void TIM2_Inicializace(void)
{    RCC->APB1ENR |= RCC_APB1ENR_TIM2EN;    // Povolení hodinového signálu pro TIM2
     TIM2->PSC = 24;                        // Nastavení předděliče čítače
     TIM2->ARR = 1000;                      // Nastavení maximální hodnoty čítače
     TIM2->CR2 = TIM_CR2_MMS_1;             // Bity MMS[2:0]=010, Update generuje signál TRGO
     TIM2->DIER |= TIM_DIER_UIE;            // Povolení přerušení od TIM2
     NVIC_EnableIRQ(TIM2_IRQn);             // Povolení přerušení od TIM2 v kontroléru NVIC
     TIM2->CR1 |= TIM_CR1_CEN;              // Spuštění časovače TIM2
     TIM2->SR |= TIM_SR_UIF;                // Nulovaní příznaku události časovače TIM2
}

void TIM2_IRQHandler()
{    TIM2->SR &= ~TIM_SR_UIF;               // Nulovaní příznaku události od časovače TIM2

//   Možná realizace zobrazení na dynamickém displeji

     togglebit(GPIOC->ODR, 10);             // Identifikace funkce přerušení časovače
                                            // Měření periody generované TIM2
}
```