Neposreden dostop do pomnilnika (DMA) v STM32F4

Na zadnji letošnji vaji se bomo spoznali z napravo, ki omogoča prenos podatkov med perifernimi napravami in pomnilnikom brez posredovanja centralne procesne enote. Te naprave poznamo pod kratico DMA, ki izhaja iz angleškega izraza direct memory access. Na predavanjih ste spoznali, da obstaja več različnih implementacij neposrednega dostopa do pomnilnika, za potrebe vaj bomo spoznali način izvedbe DMA prenosov v krimilnikih družine STM32F4.

Primer uporabe DMA krmilnika je sprejem večjega števila bajtov z up naprave USART1. Pri klasičnem pristopu, brez DMA krmilnika, bi to storili tako, da bi krmilnik sprejel vsak bajt posebej (s prekinitvami ali brez) in ga shranil na svoje mesto v polju ustrezne velikosti. Z uporabo DMA krmilnika lahko isto nalogo opravimo brez posredovanja CPE. S tem CPE razbremenimo izvajanja enostavnih pomnilniških operacij, pri katerih so same operacije trivialne, CPE pa večino časa zgolj čaka, da so podatki na voljo. V tem času lahko CPE izvaja pomembnejša opravila oziroma "počiva", saj CPE med svojim običajnim delovanjem porabi precej električne energije.

DMA streami in kanali

Mikrokrmilnik STM32F407 ima dva DMA krmilnika, oba lahko hkrati upravljata z osmimi tokovi podatkov (angl. stream). Podatkovni tokovi so označeni z oznakami Stream 0 do Stream 7. Podatkovni tok je lahko na primer zgoraj omenjen primer shranjevanja podatkov iz naprave USART1 v pomnilnik. Za vsakega izmed osmih podatkovnih tokov ima DMA krmilnik nabor osmih možnosti, ki določajo kaj in v kateri smeri se bo prenašalo. Te možnosti imenujemo kanali (angl. Channels). Sliki 1 in 2 prikazujeta tabeli kanalov za posamezne podatkovne tokove. Stream 3 naprave DMA1 tako lahko neposredno v pomnilnik zapisuje podatke, ki jih prejme naprava SPI2 ali naprava UART7. Lahko pa tudi neposredno iz pomnilnika pošilja podatke napravi USART3.

Vsaka DMA naprava upravlja z do osmimi podatkovnimi tokovi naenkrat. V danem trenutku je aktiven zgolj en podatkovni tok, ostali pa takrat čakajo. Izbiranje aktivnega podatkovnega toka izvaja arbiter na podlagi prioritete posameznega toka. Prioriteto podatkovnega toka določimo pro-

Slika 1: Streami in kanali DMA1.

Table 42. DMA1 request mapping

Peripheral requests	Stream 0	Stream 1	Stream 2	Stream 3	Stream 4	Stream 5	Stream 6	Stream 7
Channel 0	SPI3_RX	-	SPI3_RX	SPI2_RX	SPI2_TX	SPI3_TX	-	SPI3_TX
Channel 1	I2C1_RX	-	TIM7_UP	-	TIM7_UP	I2C1_RX	I2C1_TX	I2C1_TX
Channel 2	TIM4_CH1	-	I2S3_EXT_RX	TIM4_CH2	I2S2_EXT_TX	I2S3_EXT_TX	TIM4_UP	TIM4_CH3
Channel 3	I2S3_EXT_RX	TIM2_UP TIM2_CH3	I2C3_RX	I2S2_EXT_RX	I2C3_TX	TIM2_CH1	TIM2_CH2 TIM2_CH4	TIM2_UP TIM2_CH4
Channel 4	UART5_RX	USART3_RX	UART4_RX	USART3_TX	UART4_TX	USART2_RX	USART2_TX	UART5_TX
Channel 5	UART8_TX(1)	UART7_TX(1)	TIM3_CH4 TIM3_UP	UART7_RX(1)	TIM3_CH1 TIM3_TRIG	TIM3_CH2	UART8_RX(1)	TIM3_CH3
Channel 6	TIM5_CH3 TIM5_UP	TIM5_CH4 TIM5_TRIG	TIM5_CH1	TIM5_CH4 TIM5_TRIG	TIM5_CH2	-	TIM5_UP	-
Channel 7	-	TIM6_UP	I2C2_RX	I2C2_RX	USART3_TX	DAC1	DAC2	I2C2_TX

Slika 2: Streami in kanali DMA2.

Table 43. DMA2 request mapping

Peripheral requests	Stream 0	Stream 1	Stream 2	Stream 3	Stream 4	Stream 5	Stream 6	Stream 7
Channel 0	ADC1	SAI1_A(1)	TIM8_CH1 TIM8_CH2 TIM8_CH3	SAI1_A(1)	ADC1	SAI1_B(1)	TIM1_CH1 TIM1_CH2 TIM1_CH3	-
Channel 1	-	DCMI	ADC2	ADC2	SAI1_B(1)	SPI6_TX(1)	SPI6_RX(1)	DCMI
Channel 2	ADC3	ADC3	-	SPI5_RX(1)	SPI5_TX(1)	CRYP_OUT	CRYP_IN	HASH_IN
Channel 3	SPI1_RX	-	SPI1_RX	SPI1_TX	-	SPI1_TX	-	-
Channel 4	SPI4_RX(1)	SPI4_TX(1)	USART1_RX	SDIO	-	USART1_RX	SDIO	USART1_TX
Channel 5	-	USART6_RX	USART6_RX	SPI4_RX(1)	SPI4_TX(1)	-	USART6_TX	USART6_TX
Channel 6	TIM1_TRIG	TIM1_CH1	TIM1_CH2	TIM1_CH1	TIM1_CH4 TIM1_TRIG TIM1_COM	TIM1_UP	TIM1_CH3	-
Channel 7	-	TIM8_UP	TIM8_CH1	TIM8_CH2	TIM8_CH3	SPI5_RX(1)	SPI5_TX(1)	TIM8_CH4 TIM8_TRIG TIM8_COM

1. These requests are available on STM32F42xxx and STM32F43xxx.

gramsko, izbiramo lahko med štirimi nivoji prioritete: nizka, srednja, visoka in zelo visoka.

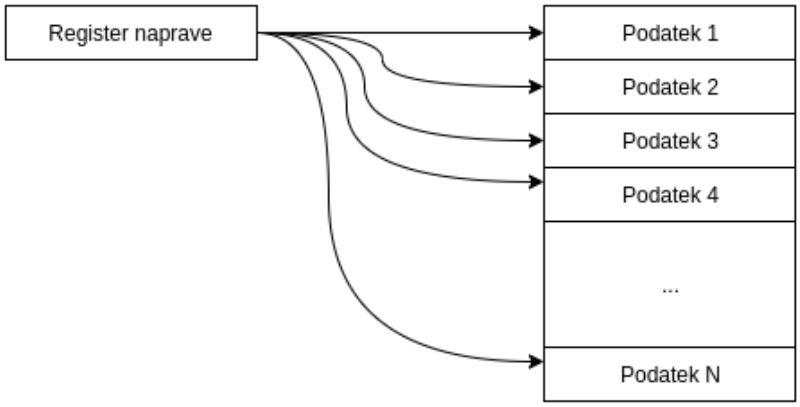
Nastavitve DMA prenosa

DMA prenosu moramo določiti več lastnosti. Prva je smer prenosa, ki je lahko *od periferne naprave do pomnilnika* (angl. peripheral to memory), *od pomnilnika do periferne naprave* (angl. memory to peripheral) ali pa *od*

pomnilnika do pomnilnika (angl. memory to memory). Slednjega je sposobna samo naprava DMA2. To lahko uporabimo, ko želimo na primer prekopirati vsebino daljšega polja na drug naslov.

Poleg smeri moramo določiti tudi izvorni (angl. source) in ciljni oz. ponorni (angl. destination) naslov DMA prenosa. V primeru prenosa od periferne naprave do pomnilnika je izvorni naslov register naprave, ciljni naslov pa naslov spremenljivke ali polja v pomnilniku. V primeru obratne smeri prenosa je ravno obratno. Določiti moramo tudi ali se izvorni oziroma ciljni naslov med prenosom povečujeta (angl. increment). Če pošiljamo podatke iz polja ali podatke sprejemamo v polje je namreč potrebno izvorni ali ciljni naslov po vsakem prenosu povečati. V nasprotnem primeru bi vsak podatek zapisali na isto mesto. Prikaz povečevanja ciljnega naslova je prikazan na sliki 3.

Slika 3: Povečevanje ciljnega naslov pri DMA prenosu.



Določiti je potrebno tudi dolžino prenosa (označeno s črko N na sliki 3) in velikost posameznega podatka. Podatek je lahko bajt (angl. byte), dva bajta (angl. half-word) ali štiri bajti (angl. word).

Izbiramo lahko med dvema tipoma prenosa: normalnim in krožnim. Pri prvem se izvrši prenos N podatkov, nato pa se DMA prenos konča. Pri krožnem prenosu pa se po prenosu N podatkov izvirni in ciljni naslov nastavita na začetne vrednosti, prenos pa se nadaljuje. Na primeru iz slike 3 se polje z N podatki tako ves čas prepisuje.

Dvojni-ciljni naslov

V primeru, da uporabljamo krožni prenos, DMA omogoča tudi uporabo dveh ciljnih naslovov. Po zaključku prenosa N podatkov, ko se pri običajnem krožnem prenosu ciljni naslov nastavi na začetno vrednost, se v tem primeru kot ciljni naslov nastavi drugi ciljni naslov. Po prenosu naslednjih N podatkov se ciljni naslov zopet nastavi na vrednost prvega ciljnega naslova. Na ta način dobimo dve polji, ki ju izmenično polnimo. V času, ko se prenašajo podatkov v drugo polje, CPE lahko obdela podatke v prvem polju in obratno.

DMA FIFO

Vsakemu podatkovnemu toku pripada šestnajst bajtni FIFO medpomnilnik (angl. buffer). V FIFO medpomnilnik se lahko začasno shranjujejo podatki iz izvora, preden se prenesejo na ciljni naslov. V primeru, da FIFO medpomnilnika ne uporabljamo, DMA deluje v neposrednem načinu (angl. direct mode). Če uporabljamo FIFO način, pa moramo določiti ali bomo uporabili celoten FIFO medpomnilnik ali zgolj četrtino, polovico oziroma tri četrtine. Hkrati s FIFO medpomnilnikom lahko uporabimo tudi eksplozijski prenos, ki je lahko v dolžini štirih, osmih ali šestnajstih bajtov.

Programski vmesnik DMA v STM32 HAL

Kot običajno moramo naše delo začeti z vklopom ure, za kar uporabimo funkcijo `__DMA1_CLK_ENABLE()` ali `__DMA2_CLK_ENABLE()`. Temu sledi inicializacija podatkovnega toka, kjer določimo vse nastavitve DMA prenosa. Podobno kot pri ostalih napravah za inicializacijo uporabimo posebno strukturo, v primeru DMA je to `DMA_HandleTypeDef`. Pri tej strukturi sta ključna dva elementa – `Instance` in `Init`.

Z elementom `Instance` določimo napravo, ki jo želimo uporabljati ter določimo podatkovni tok, na primer `DMA2_Stream2`, `DMA1_Stream4` ... Element `Init` je struktura, ki hrani vse ostale nastavitve DMA prenosa. Posamezni elementi te strukture so opisani v nadaljevanju. Ko določimo vse nastavitve prenosa, s klicem funkcije `HAL_DMA_Init(DMA_HandleTypeDef*)` napravo dejansko inicializiramo.

Channel

Nastavitev `Channel` določa kanal podatkovnega toka. Uporabimo vrednosti `DMA_CHANNEL_x`, kjer je x oznaka kanala.

Mode

Nastavitev `Mode` določa način DMA prenosa, ta je lahko normalen ali krožen. Za prvo možnost uporabimo `DMA_NORMAL`, za drugo pa `DMA_CIRCULAR`.

Priority

Nastavitev `Priority` določa prioriteto podatkovnega toka, ki je lahko:

- nizka (`DMA_PRIORITY_LOW`),
- srednja (`DMA_PRIORITY_MEDIUM`),
- visoka (`DMA_PRIORITY_HIGH`)
- ali zelo visoka (`DMA_PRIORITY_VERY_HIGH`).

Direction

Nastavitev `Direction` določa smer prenosa. Smer je lahko od periferne naprave do pomnilnika (`DMA_PERIPH_TO_MEMORY`), od pomnilnika do periferne naprave (`DMA_MEMORY_TO_PERIPH`) ali pa prenos od pomnilnika do pomnilnika (`DMA_MEMORY_TO_MEMORY`).

FIFOMode

Z nastavitvijo `FIFOMode` določimo ali želimo uporabiti FIFO medpomnilnik. V našem primeru ga ne bomo uporabljali, zato bomo uporabili nastavitev `DMA_FIFOMODE_DISABLE`.

PeriphInc in MemInc

Nastavitvi `PeriphInc` in `MemInc` določata ali želimo, da se izvorni oziroma ciljni naslov med prenosom povečujeta. Za vklop povečevanja naslova uporabimo vrednosti `DMA_PINC_ENABLE` in `DMA_MINC_ENABLE`. Za izklop namesto `ENABLE` uporabimo `DISABLE`.

Vklop

Vklop DMA prenosa moramo omogočiti na obeh straneh – na strani periferne naprave ter na strani DMA naprave. V našem primeru bomo DMA uporabili v kombinaciji z napravo USART. Za vklop sprejemanja s pomočjo DMA uporabite ukaz `SET_BIT(USART1->CR3, USART_CR3_DMAR)`, kjer namesto USART1 uporabite vašo ciljno UART napravo. Za vklop pošiljanja z DMA uporabite `SET_BIT(USART1->CR3, USART_CR3_DMAT)`.

DMA prenos nato zaženete s funkcijo `HAL_DMA_Start`, ki ji podate kazalec do inicializacijske DMA strukture, izvorni in ciljni kazalec ter dolžino prenosa. Na strani UART naprave bo naslov vedno `&USARTx->DR`, saj register `DR` UART naprava uporablja tako za sprejemanje kot pošiljanje. Primer uporabe pri sprejemanju je prikazan spodaj.

```
1 HAL_DMA_Start(&dma, (uint32_t)&USART1->DR, (uint32_t)buff, 10);
```

Kjer `buff` predstavlja spremenljivko (polje) v katero želimo shraniti podatke iz USART1, `dma` pa inicializacijsko strukturo (`DMA_HandleTypeDef`),

Naloga

Tokratna naloga je tretja in zadnja naloga za oddajo. Rok za oddajo je nedelja, 16.1. ob 23:59

Napišite program, ki bo počel sledeče:

- vsakih 50 milisekund prebere naklon po x in y-osi senzorja gibanja in ga shrani na mesto v poljih Ax in Ay, v obeh je prostora za 80 meritev. Za ustvarjanje časovnega intervala morate uporabiti enega izmed časovnikov TIMx. Ne smete uporabiti funkcije `HAL_Delay()` ali se zanašati na milisekudne prekinitve SysTick.

- vsake 4 sekunde sproži DMA prenos, ki prenese vsebino polj Ax in Ay v večje polje B. Polje B ima prostora za 5 polj Ax in Ay (skupno prostora za 800 meritev). Po petih prenosih naj bo tako v polju B vseh 800 meritev opravljenih v zadnjih dvajsetih sekundah. Prenose polj v polje B je obvezno opraviti s pomočjo DMA krmilnika

- po zaključku petih parov prenosa v polje B izračunajte koliko se je razvojna plošča premikala v zadnjih dvajsetih sekundah. Določite 2 stopnji nagibov: nizko in visoko. V primeru, da ploščo močno nagibate, naj program po dvajsetih sekundah zazna visoko stopnjo, sicer pa nizko. Mejo med nizko in visoko stopnjo določite sami.

- v primeru visoke stopnje naslednjih 20 sekund (do naslednjega preverjanja) utripajte z vsemi štirimi LED diodami.

  Namig: za zaznavanje stopnje nagibov najprej izračunajte povprečje meritev za vsako os ter nato koliko meritve odstopajo od izračunanega povprečja (t.i. varianca).