STM32 – interfejs QuadSPI

Niemal wszystkie nowe modele mikrokontrolerów serii STM32L4 i STM32F4 są wyposażone w interfejs QSPI, przeznaczony do współpracy z szybkimi pamięciami z interfejsem szeregowym. Interfejs QSPI (QuadSPI) stanowi rozwinięcie znanego interfejsu SPI. Główną różnicą pomiędzy SPI i QSPI jest zwiększona liczba i zmodyfikowana funkcjonalność linii danych.

Interfejs SPI jest synchronicznym, dwukierunkowym interfejsem szeegowym. Korzysta on z czterech linii:

- NSS – linia uaktywnienia urządzenia podrzędnego – opadające zbocze rozpoczyna transakcję SPI, a narastające kończy ją.
- SCLK – linia zegara transmisji – zbocza zegara wyznaczają chwile zmiany stanów linii danych oraz ich próbkowania.
- MOSI (Master Out, Slave In) – linia danych transmitowanych z urządzenia nadrzędnego do podrzędnego.
- MISO (Master In, Slave Out) – linia danych transmitowanych z urządzenia podrzędnego do nadrzędnego.

Transmisja danych odbywa się równocześnie w obu kierunkach. Przyjmuje się, że zmiana stanu linii danych przez oba urządzenia zachodzi na przeciwnych zboczach zegara. Interfejs SPI ma dwie opcje konfiguracji:

- CPOL – określa stan linii SCK podczas bezczynności.
- CPHA – określa, czy dane są próbkowane na nieparzystych (0), czy na parzystych (1) zboczach przebiegu zegarowego.

Poszczególne układy wyposażone w interfejs SPI mogą wymagać różnych konfiguracji interfejsu. Większość współczesnych układów może działać w dwóch konfiguracjach: CPOL=0 i CPHA=0 (tzw. tryb 0) lub CPOL=1 i CPHA=1 (tzw. tryb 3).

QuadSPI

W interfejsie SPI występują dwie jednokierunkowe linie danych, MOSI i MISO. Interfejs QSPI może pracować w trzech trybach, różniących się użyciem linii danych:

- W trybie zgodności z SPI logika interfejsu jest taka sama, jak w SPI.
- W trybie dual obie linie danych są dwukierunkowe i noszą oznaczenia IO0 i IO1; w każdym takcie zegara SCLK mogą być przesyłane równolegle w tym samym kierunku dwa bity danych.
- W trybie quad interfejs używa czterech linii (dodatkowe linie są oznaczone IO2 i IO3), a w każdym cyklu zegara mogą być transmitowane cztery bity danych.

Rozwojowe wersje interfejsu mogą również umożliwiać transmisję dwuzboczową oraz zwiększać liczbę linii danych do ośmiu (OctoSPI).

Przełączanie trybów pracy

Interfejs QSPI rozpoczyna działanie w trybie SPI. Zmiana trybu następuje po przesłaniu do układu podrzędnego polecenia przełączenia trybu pracy. Przełączenie to może mieć skutek chwilowy (do końca transakcji SPI) lub trwały. W tym drugim przypadku kolejne transakcje SPI (po kolejnym uaktywnieniu linii NSS) będą rozpoczynane w trybie Quad i cała transmisja będzie używała czterech linii danych.

Pamięci QSPI Flash serii 25Q

Pamięci z interfejsem QSPI są wytwarzane przez wielu producentów. Typowe pamięci NOR Flash, zawierające w oznaczeniu typu znaki 25Q, mają pojemności do 32 MB (256 MB) mają obudowy o ośmiu wyprowadzeniach. Dwa wyprowadzenia służą do zasilania układu, kolejne cztery udostępniają interfejs SPI/DualSPI. Pozostałe dwa wyprowadzenia mogą funkcjonować jako linie IO2 i IO3 interfejsu QSPI, jednak ich domyślne funkcje przed ustawieniem trybu Quad są inne – są one wejściami sterującymi:

- –WP (IO2) – poziom niski powoduje wyłączenie możliwości zapisu pamięci,
- –HOLD (IO3) – poziom niski powoduje dezaktywację interfejsu SPI.

Projektując urządzenia z pamięciami QSPI należy zwrócić uwagę na poprawne sterowanie tych linii w czasie, gdy interfejs działa w trybie SPI lub DualSPI.

Każda transakcja dotycząca pamięci rozpoczyna się od przesłania do pamięci 8-bitowego polecenia. Dalszy przebieg transakcji zależy od jej rodzaju. W przypadku transakcji operacji odczytu, zapisu lub kasowania pamięci, po poleceniu jest przesyłany adres, a następnie dane. W zależności od typu pamięci i samego polecenia, pomiędzy poleceniem i adresem lub pomiędzy adresem i danymi mogą być wymagane dodatkowe przerwy – transmitowane są wtedy dodatkowe bajty, które w niektórych pamięciach mogą zawierać znaczniki modyfikacji sposobu wykonania polecenia.

Przedstawione poniżej fragmenty oprogramowania zapewniają wykonania podstawowych operacji na pamięci GD25Q32, produkowanej przez GigaDevice. Dla podstawowej współpracy z pamięcią są w tym przypadku używane tylko 4 polecenia, przedstawione w **tabeli 1**.

Interfejs QuadSPI w STM32

Interfejs QUADSPI zastosowany w mikrokontrolerach serii STM32F4 i STM32L4 umożliwia łatwą współpracę z pamięciami Flash wyposażonymi w interfejs QSPI. Oprócz łatwiejszej niż przy użyciu zwykłego interfejsu SPI wymianie danych z pamięciami udostępnia on dodatkową, bardzo pożyteczną funkcjonalność: umożliwia

Tabela 1. Wybrane polecenia pamięci Flash GD25Q32

Bajt polecenia	Argumenty	Opis
0x06	brak	Zezwolenie na zapis danych i zmianę trybu pracy
0x31	brak	Włączenie trybu Quad
0x05	<odczytany stan – 1 B>	Odczyt stanu pamięci (znacznika niegotowości)
0x20	<adres – 3 B>	Kasowanie sektora 4 KiB
0x32	<adres – 3 B><tryb – 1 B><odstęp><dane>	Zapis danych w trybie Quad
0xeb	<adres – 3 B><dane>	Odczyt danych w trybie Quad

Poprzednie części kursu i dodatkowe materiały dostępne są na stronie www.media.avt.pl

Listing 1. Konfigurowanie QUADSPI

```
                                                                    // QSPI - GD25Q32 connected as block2
QUADSPI->DCR = 21 << QUADSPI_DCR_FSIZE_Pos | 7 << QUADSPI_DCR_CSHT_Pos;   // 22 addr bits, 8 idle cycles between accesses
QUADSPI->CR = QUADSPI_CR_APMS | 0 << QUADSPI_CR_PRESCALER_Pos | QUADSPI_CR_FSEL | QUADSPI_CR_EN;
QUADSPI->PIR = 100;
QUADSPI->PSMKR = 1;                                                 //status match mask - write in progress bit
                                                                    //QUADSPI->PSMAR = 0;
                                                                    // status match pattern - write in progress = 0 (default val)
QUADSPI->LPTR = 200;                                                // memory mapped mode timeout
```

on bezpośredni, przezroczysty odczyt dowolnych lokacji pamięci, odwzorowanych w przestrzeni adresowej procesora.

Peryferiale QUADSPI występujące w poszczególnych modelach mikrokontrolerów rożnią się nieco bardziej zaawansowanymi możliwościami; tutaj przedstawiona zostanie podstawowa funkcjonalność bloku QSPI, dostępna we wszystkich mikrokontrolerach.

Interfejs ma trzy tryby działania, służące do różnych celów:

1. W trybie podstawowym (*indirect*) umożliwia on ułatwione przesyłanie dowolnych poleceń do pamięci, wraz z transmisją adresów i danych. W trybie tym współpraca oprogramowania z pamięcią jest podobna, jak przy użyciu zwykłego interfejsu SPI, ale znacznie wygodniejsza, gdyż interfejs samoczynnie przesyła do pamięci adresy.
2. W trybie odpytywania (*status polling*) interfejs może samoczynnie powtarzać transmisję polecenia odczytu stanu pamięci, aż do odczytania zadanej wartości bajtu stanu. Tryb ten służy do oczekiwania na zakończenie operacji kasowania lub programowania bez aktywnego udziału oprogramowania.
3. Najciekawszym dla projektanta urządzeń trybem pracy jest trzeci tryb – przeźroczysty (*memory-mapped*), w którym pamięć zostaje odwzorowana w przestrzeni adresowej procesora Cortex-M, dzięki czemu może ona być odczytywana tak samo, jak pamięci Flash i RAM zawarte w mikrokontrolerze. W trybie tym, umożliwiającym najszybszy dostęp do pamięci, może ona być używana do pobierania programu lub dowolnych danych używanych przez oprogramowanie, np. czcionek, obrazów lub dźwięków. Tryb przezroczysty może być również użyty do realizacji urządzenia pamięci masowej z bezpośrednim dostępem do odczytu plików.

Tryb przeźroczysty

Tryb przeźroczysty może być używany wyłącznie przy odczycie pamięci. Do zapisu danych niezbędne jest przełączenie interfejsu QSPI w tryb podstawowy i jawne przesyłanie poleceń kasowania i zapisu. Po zaprogramowaniu trybu przezroczystego pamięć SPI staje się widoczna w przestrzeni adresowej mikrokontrolera począwszy od adresu 0x90000000. Dostępne okno ma rozmiar 256 MiB. Jeżeli pojemność pamięci jest większa – jej pozostała część nie jest dostępna w trybie przeźroczystym. Przy próbie odczytu pamięci przez procesor interfejs QSPI automatycznie wysyła do pamięci polecenie odczytu z odpowiednim adresem. W tym czasie procesor zostaje wstrzymany do zakończenia zleconej transakcji odczytu. Odczytywany jest zawsze dłuższy blok, a odczytana zawartość pamięci jest przechowywana w buforze podręcznym, więc odczyty kolejnych lokacji pamięci wykonywane w pewnych odstępach czasu nie powodują kolejnych wstrzymywań pracy procesora. W pamięci mogą być przechowywane zarówno dane, jak i program, a zastosowane mechanizmy buforowania powodują, że różnica w szybkości wykonania programu pomiędzy wewnętrzną pamięcią Flash i pamięcią QSPI nie jest na ogół znacząca.

Listing 2. Procedury dla trybu podstawowego

```
static _Bool fl_writeop, fl_mmread;

static void sf_waitrdy(void)
{
    while (QUADSPI->SR & QUADSPI_SR_BUSY);
    QUADSPI->FCR = QUADSPI_FCR_CTOF | QUADSPI_FCR_CSMF | QUADSPI_FCR_CTCF | QUADSPI_FCR_CTEF;
}
                                                                    // enable write access
static void sf_wren(void)
{
    sf_waitrdy();
    QUADSPI->CCR = QUADSPI_CCR_FMODE_IW | QUADSPI_CCR_IMODE_0 | 0x06;
    sf_waitrdy();
}
                                                                    // enable quad mode
void sfll_enquad(void)
{
    sf_wren();
    QUADSPI->DLR = 0;
    QUADSPI->CCR = QUADSPI_CCR_FMODE_IW | QUADSPI_CCR_DMODE_0 | QUADSPI_CCR_IMODE_0 | 0x31;
    QUADSPI_DRB = 0x02;
    sf_waitrdy();
}
                                                                    // write memory page
static void sf_mbwrite(uint32_t addr, const uint8_t *data, uint32_t dlen)
{
    if (!dlen) return;
    sf_wren();
                                                                    // send command, then send and receive data from/to data buffer
    QUADSPI->DLR = dlen - 1;
                                                                    // quad page program
    QUADSPI->CCR = QUADSPI_CCR_FMODE_IW | QUADSPI_CCR_DMODE4        // 4 lines
        | QUADSPI_CCR_ADSIZE24 | QUADSPI_CCR_ADMODE1                // 1 line
        | QUADSPI_CCR_IMODE_0 | 0x32;
    QUADSPI->AR = addr;
    while (dlen --)
    {
        while (~QUADSPI->SR & QUADSPI_SR_FTF);
        QUADSPI_DRB = *data++;
    }
    fl_writeop = 1;
}
                                                                    // erase sector
void sfll_mserase(uint32_t addr)
{
   sf_wren();
    QUADSPI->CCR = QUADSPI_CCR_FMODE_IW
        | QUADSPI_CCR_ADMODE_0 | QUADSPI_CCR_ADSIZE24
        | QUADSPI_CCR_IMODE_0 | 0x20;
    QUADSPI->AR = addr;
    fl_writeop = 1;
}
```

Listing 3. Odpytywanie – oczekiwanie na gotowość pamięci po zapisie lub kasowaniu

```
static void sf_waitwrdy(void)
{
    sf_waitrdy();
    if (fl_writeop)
   {
        QUADSPI->DLR = 0;                                                                    // 1 byte
        QUADSPI->CCR = QUADSPI_CCR_FMODE_AP | QUADSPI_CCR_DMODE_0 | QUADSPI_CCR_IMODE_0 | 0x05;
        while ((QUADSPI->SR & QUADSPI_SR_SMF) == 0);
        QUADSPI->FCR = QUADSPI_FCR_CTOF | QUADSPI_FCR_CSMF | QUADSPI_FCR_CTCF | QUADSPI_FCR_CTEF;
        fl_writeop = 0;
    }
}
```

Do programowania lub zapisu pamięci niezbędne jest zakończenie działania interfejsu QUADSPI w trybie przezroczystym.

Programowanie interfejsu QUADSPI

Do zaprogramowania interfejsu QUADSPI niezbędna jest znajomość używanej w urządzeniu pamięci Flash. Poszczególne typy pamięci mogą się nieco różnić szczegółami poleceń, dlatego zawsze należy zapoznać się z dokumentacją układu pamięci.

Należy również zwrócić uwagę na to, że sam interfejs QUADSPI w mikrokontrolerach STM32 ma pewne niezgodności z dokumentacją. Przy jego programowaniu należy uwzględnić nie tylko informacje z *Reference Manual*, ale również te zawarte w dokumencie *Errata sheet*, w którym znajduje się opis rozbieżności pomiędzy założonym i rzeczywistym działaniem bloku QUADSPI.

Dalej zaprezentowano przykłady programowania interfejsu QSPI mikrokontrolera STM32L496 do współpracy z pamięcią GD25Q32.

Przygotowanie do pracy

Pierwszym krokiem potrzebnym do uruchomienia interfejsu QUADSPI jest włączenie bloków GPIO i QSPI oraz konfiguracja linii portów mikrokontrolera. Należy przy tym:

- Ustawić największą szybkość dla wszystkich linii QUADSPI (rejestr GPIOx→OSPEEDR).
- Włączyć podciąganie linii IO2 i IO3, niezbędne do działania interfejsu w zwykłym trybie SPI.
- Wybrać funkcję QUADSPI w rejestrach GPIOx->AFR.
- Włączyć tryb AF dla wyprowadzeń QUADSPI.

Konfigurowanie QUADSPI

W celu skonfigurowania bloku QUADSPI należy zaprogramować jego podstawowe parametry:

- W rejestrze DCR – rozmiar pamięci, liczbę cykli zegarowych pomiędzy transakcjami, tryb SPI (0/3).
- W rejestrze CR – dzielnik częstotliwości przebiegu zegarowego pamięci, wybór banku pamięci, automatyczne zakończenie odpytywania.
- W rejestrze PIR – odstęp pomiędzy odczytami stanu w trybie odpytywania.
- W rejestrze PMSKR – maskę bitu gotowości w bajcie stanu pamięci dla operacji odpytywania.
- W rejestrze PSMAR – wzorzec informacji o gotowości w bajcie stanu pamięci.
- W rejestrze LPTR – czas, po którym następuje dezaktywacja pamięci przy braku dostępu w trybie przeźroczystym.

Programowanie QUADSPI dla pamięci podłączonej jako bank 2 pokazano na **listingu 1**. Większość pamięci QSPI wymaga jawnego włączenia możliwości użycia trybu QuadSPI. Po skonfigurowaniu interfejsu QUADSPI należy więc jednorazowo wysłać do pamięci odpowiednie polecenie.

Programowanie QUADSPI w trybie podstawowym

Pojedyncza transakcja QUADSPI w trybie podstawowym może składać się z następujących faz:

- przesłania do pamięci bajtu polecenia,
- przesłania do pamięci adresu,
- odstępu w postaci określonej liczby cykli zegarowych,
- transmisji danych do lub z pamięci.

Każda z tych faz jest opcjonalna Najprostsza transakcja polega na przesłaniu tylko bajtu polecenia. Interfejs QUADSPI umożliwia zaprogramowanie sposobu wykonania każdej fazy. Służy do tego rejestr CCR, do którego wpisuje się parametry transakcji, w tym:

- wartość bajtu plecenia,
- długość adresu,
- liczbę bajtów dodatkowych przesyłanych przed danymi,
- liczbę cykli odstępu pomiędzy bajtami dodatkowymi i początkiem transmisji danych,
- sposób transmisji w poszczególnych fazach transakcji (SPI, DualSPI, QUADSPI),
- tryb pracy QUADSPI i rodzaj operacji.

Wartości poszczególnych parametrów, w tym tryby pracy interfejsu w poszczególnych fazach transakcji i wartości bajtów dodatkowych, muszą być zgodne z wymaganiami zastosowanego układu pamięci Flash. Typowo bajt polecenia jest przesyłany w trybie SPI, a dane – w trybie Quad. Sposób przesyłania adresu do pamięci zależy od polecenia; zwykle w poleceniach odczytu jest to tryb Quad, a w pozostałych – tryb SPI.

Należy również zwrócić uwagę na błędne działanie modułu QSPI opisane w dokumencie Errata sheet [2]. Jeśli pamięć wymaga dodatkowych cykli odstępu przed danymi zapisywanymi do pamięci, nie należy programować cykli odstępu, a zamiast tego należy wprowadzić bajty dodatkowe lub odpowiednio zwiększyć ich liczbę.

Pole FMODE rejestru CCR służy do programowania trybu pracy QUADSPI. Przyjmuje ono jedną z czterech wartości, odpowiadających czterem trybom:

- odczytu zwykłego (używany również dla poleceń nie wymagających zapisu danych),
- zapisu danych,
- automatycznego odpytywania,
- odczytu przezroczystego.

Listing 4. Programowanie odczytu w trybie przezroczystym

```
void sfll_startmm(void)
{
    sf_waitwrdy();
    sf_waitrdy();

    QUADSPI->CCR = QUADSPI_CCR_FMODE_MM | QUADSPI_CCR_DMODE4                  // quad io fast read - ok @ 80 MHz FAST
        | 4 << QUADSPI_CCR_DCYC_Pos                                           // 4 lines
        | QUADSPI_CCR_ABMODE4                                                 // 4 clocks
        | QUADSPI_CCR_ADSIZE24 | QUADSPI_CCR_ADMODE4                         // 4 lines, 1 byte
        | QUADSPI_CCR_IMODE_0 | 0xeb;                                         // 4 lines
    fl_mmread = 1;
}
```

Adres, który ma być przesłany do pamięci, jest zapisywany do rejestru AR. Wartości bajtów dodatkowych należy wpisać do rejestru ABR. Przy transakcjach zwykłego zapisu lub odczytu danych zapisuje się do rejestru DLR liczbę bajtów danych pomniejszoną o jeden. Wpisanie polecenia do rejestru CCR powoduje zainicjowanie transakcji. Jeżeli transakcja obejmuje również transmisję danych, są one przesyłane przez rejestr DR. Podczas transmitowania danych do i z pamięci w trybie podstawowym należy pamiętać o rozmiarze dostępu do rejestru danych QUADSPI. Jeśli przesyłamy dane w jednostkach 8-bitowych, dostęp do rejestru QUADSPI→DR musi być 8-bitowy. Procedury włączenia trybu QuadSPI oraz kasowania sektora i zapisu danych przedstawiono na **listingu 2**.

Programowanie odpytywania

Po wcześniejszym jednorazowym skonfigurowaniu parametrów odpytywania, samo użycie trybu odpytywania sprowadza się do zaprogramowania polecenia odpytywania i włączenia trybu odpytywania. Odczytanie przez QUADSPI pamięci bajtu stanu o wartości odpowiadającej stanowi gotowości powoduje ustawienie znacznika SMF w rejestrze QUADSPI→SR.

Programowanie odczytu w trybie przezroczystym

Programowanie przezroczystego dostępu do pamięci jest podobne, jak programowanie odczytu w trybie podstawowym. Nie ma tu jednak potrzeby podawania adresu ani długości danych. Do rejestru CCR wpisuje się polecenie odczytu i ustawia się tryb przezroczysty. Po jednorazowym zaprogramowaniu interfejs QUADSPI będzie samoczynnie generował polecenia odczytu pamięci odpowiadające odczytom adresów okna pamięci QSPI inicjowanym przez procesor.

Przed zainicjowaniem dostępu przezroczystego należy zaczekać na zakończenie ewentualnych operacji zapisu lub kasowania.

Grzegorz Mazur

Bibliografia

1. RM0351 Reference manual, STM32L4x5 and STM32L4x6 advanced ARM^{q}-based 32-bit MCUs, ST Microelectronics 03'2017
2. ES0335 STM32L496xx STM32L4A6xx Errata sheet, ST Microelectronics 02'2018
3. GD25Q32C DATASHEET, GigaDevice 2016