MSP430 w przykładach (8)

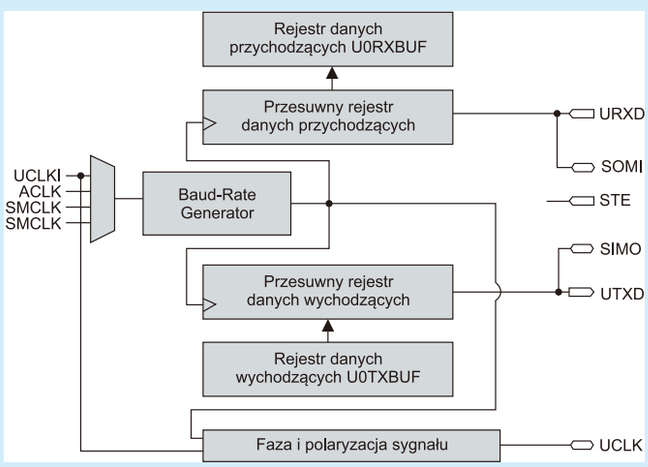
Transmisja szeregowa UART, SPI

W artykule omówimy obsługę interfejsów komunikacyjnych UART oraz SPI w mikrokontrolerze MSP430f1232. Przy okazji podamy sporą porcję informacji teoretycznych. Omówimy również praktyczne przykłady obsługi interfejsów.

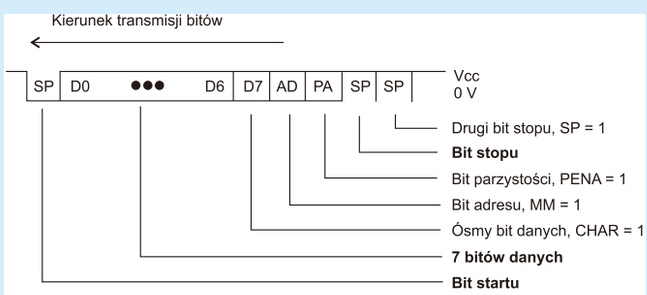
Zainstalowany w module „Komputerek" mikrokontroler MSP430f1232 ma wbudowany moduł transmisji szeregowej USART (**rysunek 1**). Korzystając z modułu USART możemy zrealizować obsługę jednego z interfejsów komunikacyjnych: UART, albo SPI. Rodzaj obsługiwanego interfejsu konfiguruje bit SYNC z rejestru U0CTL (wartość 0 - interfejs UART, wartość 1 – interfejs SPI).

Moduł USART. Interfejs asynchroniczny UART

Interfejs UART jest przeznaczony do asynchronicznej, szeregowej transmisji danych. Do jej realizacji są używane linie I/O mikrokontrolera: linia nadawcza UTXD oraz linia odbiorcza URXD. Dane transmitowane są w „ramkach". Prędkość transmisji danych ustala programista. Schemat blokowy modułu USART w trybie obsługi interfejsu UART (bit SYNC = 0) pokazano na **rysunku 2**. Podstawowe elementy modułu to: rejestry odbiornika (przesuwny i odbiorczy), rejestry nadajnika (przesuwny i nadawczy), linie transmisyjne (URXD – odbiorcza, UTXD – nadawcza), oraz generator podstawy czasu transmisji.



Rysunek 1. Schemat blokowy modułu USART. Obsługa interfejsów UART / SPI



Rysunek 2. Moduł USART. Interfejs UART. Szarym kolorem zaznaczono bloki funkcjonalne odpowiedzialne za obsługę interfejsu SPI

Dodatkowe materiały na CD/FTP:
ftp://ep.com.pl, user: 75282, pass: 852sjb64

Ramka transmisyjna. Podstawowa ramka transmisyjna rozpoczyna się bitem startu, zawiera 7 bitów danych i kończy 1 bitem stopu. Dodatkowo, w ramce może być przesłany ósmy bit danych, bit adresu ramki, bit kontroli parzystości oraz drugi bit stopu. Wygląd ramki transmisyjnej pokazano na **rysunku 3**.

Wysyłanie danych. Dane wysyłane są w ramkach transmisyjnych. W pojedynczej ramce transmisyjnej bity wysyłane są w kolejności od najmniej do najbardziej znaczącego. Nadawanie inicjuje wpisanie danych (7 albo 8 bitów) do rejestru nadajnika U0TXBUF. Wówczas jest formowana ramka transmisyjna. Następnie kolejne bity danych poprzez przesuwny rejestr nadajnika „wypychane są" na linię nadawczą UTXD. Po wysłaniu danych jest ustawiana flaga przerwania UTXIFG0.

Odbieranie danych. Mikrokontroler próbkuje linię odbiorczą URXD. W momencie wykrycia początku ramki transmisyjnej (bit startu), odczytywane są kolejne bity transmisji. Odebrane bity danych są wprowadzane do rejestru przesuwnego odbiornika, a gdy zostanie wykryty koniec ramki (bit stopu), to dane (7 albo 8 bitów) przepisywane są do rejestru odbiornika U0RXBUF. Wówczas jest ustawiana flaga przerwania URXIFG0.

Generator taktujący. Moduł USART wyposażono w generator taktujący transmisję szeregową (*Baud Rate Generator*). Jego zadaniem jest wytworzenie sygnału zegarowego BITCLK używanego do taktowania transmisji UART lub SPI. Schemat blokowy układu pokazano na **rysunku 4.** Podstawowe elementy generatora taktującego to: 16-bitowy licznik, komparator, dwa rejestry dzielnika BRCLK, modulator oraz rejestr modulatora. Na wejście układu jest podawany sygnał BRCLK. Na wyjściu jest generowany sygnał BITCLK.

Źródło sygnału BRCLK konfiguruje bit SSELx z rejestru U0TCTL. Sygnał wejściowy BRCLK może być taktowany jednym z sygnałów zegarowych ACLK, SMLCK, albo przez zewnętrzny sygnał UCLKI doprowadzony do wejścia UCLK mikrokontrolera. Częstotliwość sygnału wyjściowego BITCLK (bez modulacji) określa **wzór 8.1**.

(8.1) $$f_{BITCLK} = f_{BRCLK}/N$$

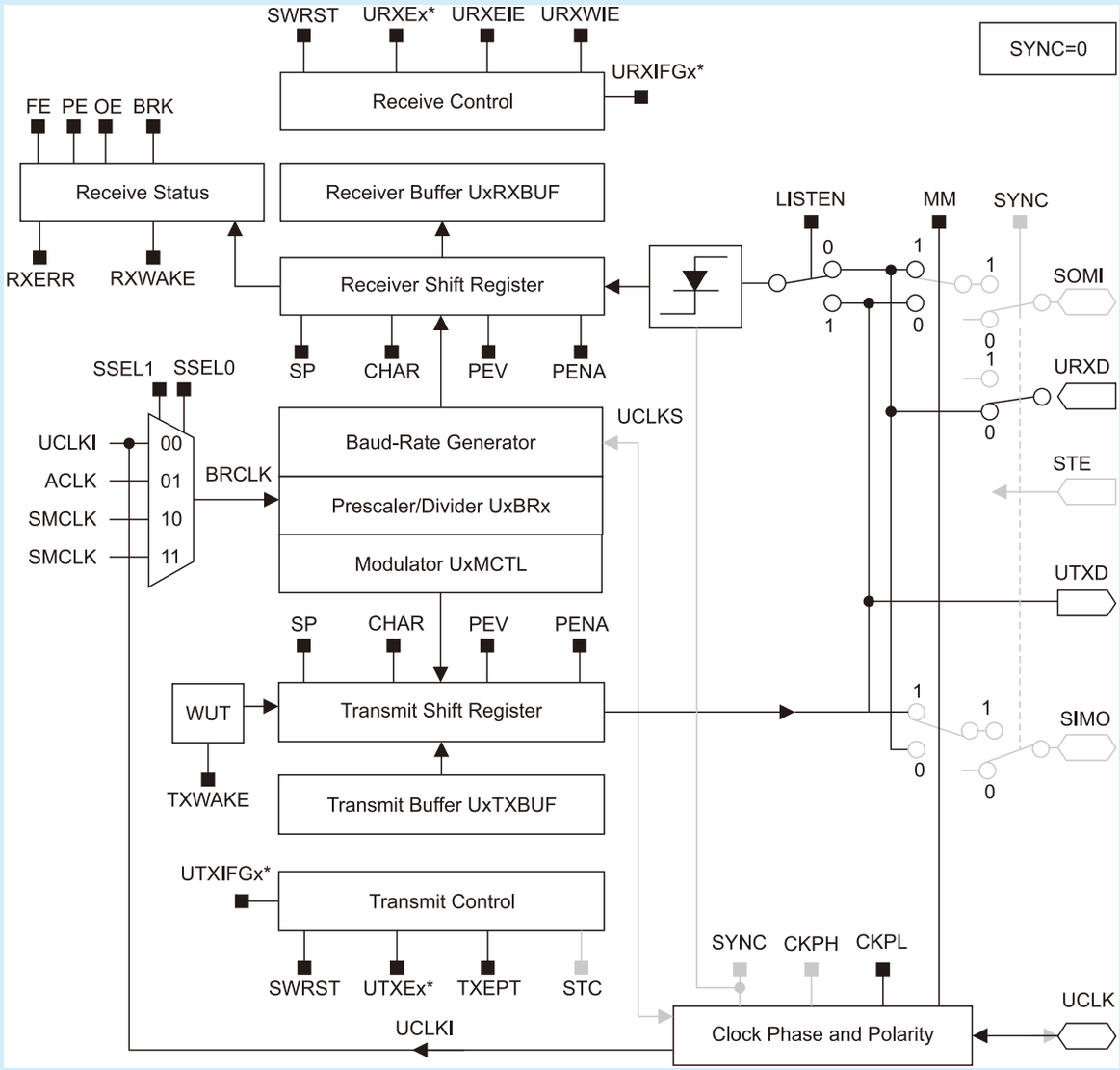
(8.2) $$N = f_{BRCLK}/f_{BITCLK}$$

gdzie:

fBITCLK – częstotliwość sygnału wyjściowego BITCLK [Hz]

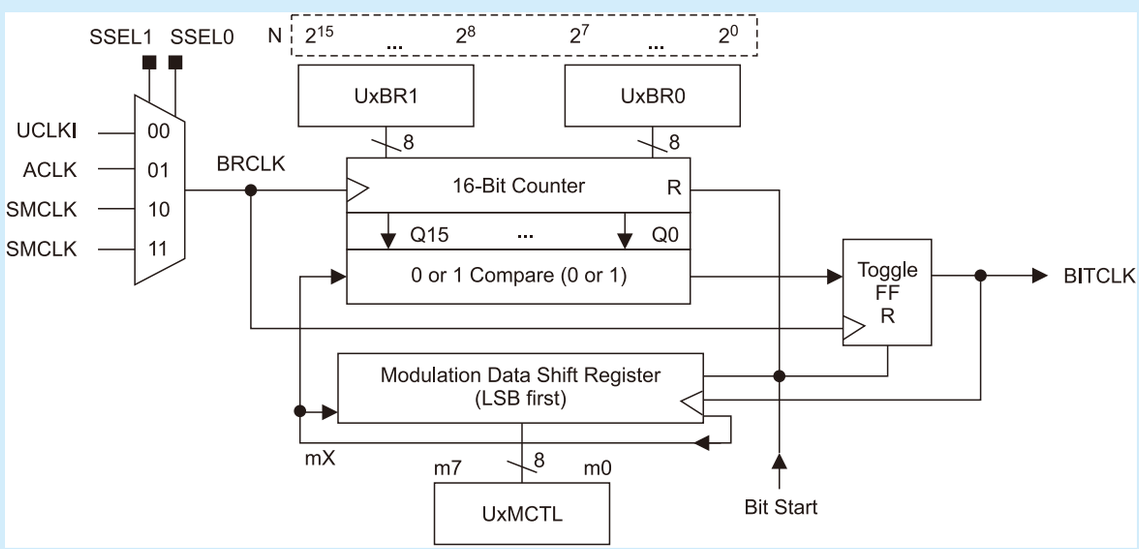
fBRCLK – częstotliwość sygnału wejściowego BRCLK [Hz]

N – dzielnik częstotliwości BRCLK

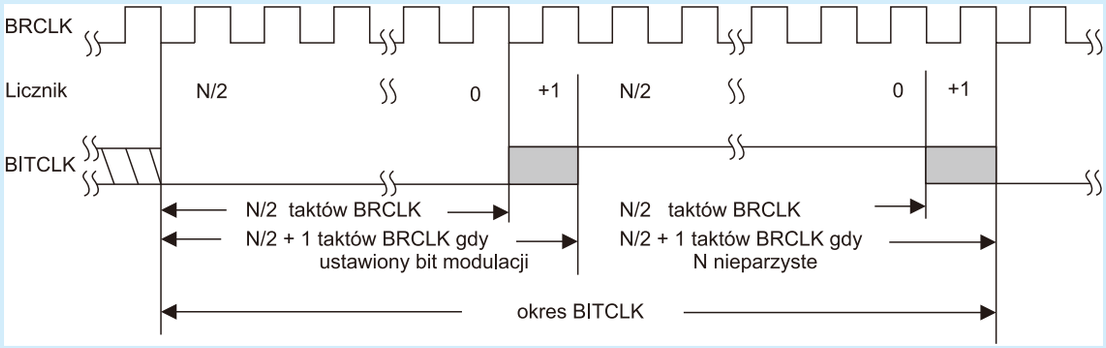


Rysunek 3. Interfejs UART. Ramka transmisyjna. Podstawowe pola to: bit startu, 7 bitów danych, bit stopu

Okres sygnału wyjściowego trwa N taktów sygnału wejściowego. Takty sygnału wejściowego zlicza 16-bitowy licznik wbudowany w układ generatora. Jeśli N jest liczbą parzystą, to dwukrotnie jest odliczane po N/2 taktów sygnału. Jeśli N jest liczbą nieparzystą, to najpierw jest odliczane N/2 taktów sygnału, a następnie odliczane są pozostałe takty sygnału (N/2+1) wejściowego. Dodatkowo, za pomocą BITCLK jest możliwe modulowa-



Rysunek 4. Schemat blokowy generatora taktującego



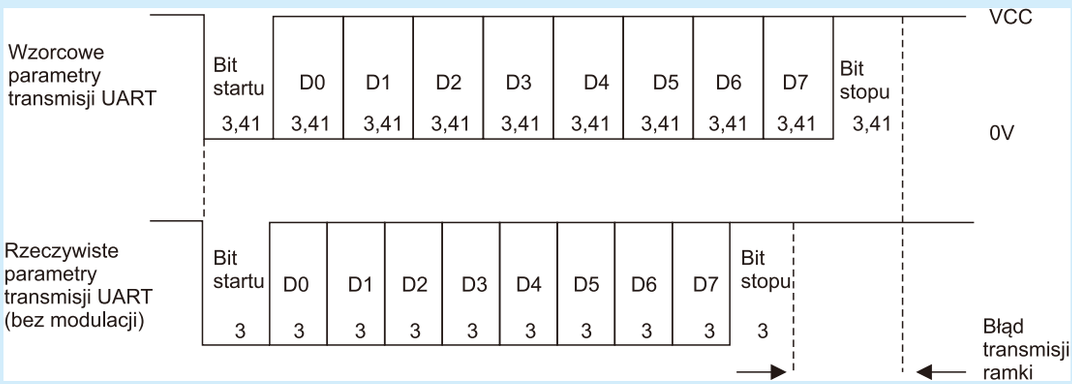
Rysunek 5 Generator taktujący: przebiegi czasowe sygnałów BRCLK, BITCLK

nie częstotliwości sygnału wyjściowego. Gdy generator taktujący wykryje ustawiony bit modulacji, to najpierw odlicza N/2+1, a następnie N/2 (N parzyste) albo N/2+1 (N nieparzyste) taktów sygnału wejściowego. Przebiegi czasowe sygnałów pokazano na **rysunku 5.**

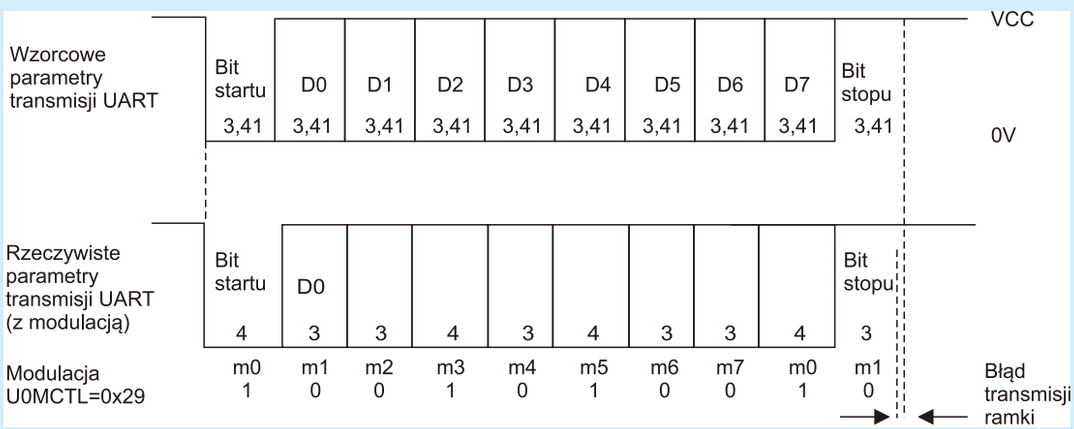
Prędkość transmisji. Transmisja danych taktowana jest sygnałem BITCLK wytwarzanym na wyjściu układu generatora taktującego (*Baud Rate Generator*). Prędkość transmisji danych wyrażona w bitach na sekundę (b/s) jest równa częstotliwości sygnału BITCLK (wzór 8.1).

Aby ustalić prędkość transmisji, konfigurujemy parametry pracy generatora. Ustawiamy źródło sygnału wejściowego BRCLK. Obliczamy wartość dzielnika częstotliwości BRCLK (wzór 8.2). Obliczoną wartość dzielnika sygnału wejściowego BRCLK wpisujemy do rejestrów U0BR1, U0BR0 (bardziej znaczący bajt do rejestru U0BR1, mniej znaczący do rejestru U0BR0). W mikrokontrolerach MSP430 możemy konfigurować praktycznie dowolną prędkość transmisji danych UART. Jedyne ograniczenie jest takie, że maksymalna prędkość transmisji danych (częstotliwość sygnału wyjściowego) nie może być wyższa niż 1/3 częstotliwości sygnału wejściowego BRCLK (wartość N musi być większa bądź równa 3).

Obliczona ze wzoru 8.2 wartość N może być liczbą zmiennoprzecinkową. Przykładowo, przy taktowaniu sygnału wejściowego BRCLK sygnałem o częstotliwości 32768 Hz (kwarc zegarkowy) i prędkości transmisji 9600 b/s, wartość N wynosi 3,41. Ponieważ do rejestrów U0BR1 i U0BR0 można wprowadzać tylko liczby stałoprzecinkowe, to w tym konkretnym przypadku do rejestru U0BR1 należy wprowadzić wartość 0, a do rejestru U0BR0 wartość 3. Prędkość transmisji danych (częstotliwość sygnału wyjściowego BITCLK) wyniesie 10923 b/s (32768 Hz/3). Różnica pomiędzy prędkością, którą chce-



Rysunek 6. Transmisja UART (brak modulacji). Przykładowa ramka danych (BRCLK = 32768 Hz, definiowana prędkość transmisji 9600 b/s). Każdy bit jest transmitowany z błędem 14%. Sumaryczny błąd (przesunięcie o 140%, czyli 1,4 bitu) uniemożliwia poprawną transmisję danych



Rysunek 7. Transmisja UART (modulacja 0x29). Przykładowa ramka transmisji danych (BRCLK = 32768 Hz, prędkość transmisji 9600 b/s). Maksymalny błąd transmisji bitu to 17%. Sumaryczny błąd (przesunięcie o 0,039 bitu) nie wpływa na poprawność transmisji danych

Ramka 8.1 Konfiguracja kontrolera USART. Interfejs UART

1. Włącz tryb restartu kontrolera. (ustaw bit SWRST w rejestrze U0CTL).
2. Ustal rejestry konfiguracyjne. (U0CTL, U0TCTL, U0RCTL oraz U0BR1, U0BR0, U0MCTL).
3. Włącz moduł odbiorczy lub nadawczy, ewentualnie oba jednocześnie. (ustaw bit/bity UTXE0/URXE0 w rejestrze ME2).
4. Wyłącz tryb restartu kontrolera. (wyzeruj bit SWRST w rejestrze U0CTL).
5. Opcjonalnie włącz obsługę przerwań danych przychodzących/wychodzących (ustaw bit/bity URXIE0, UTXIE0 w rejestrze IE2).

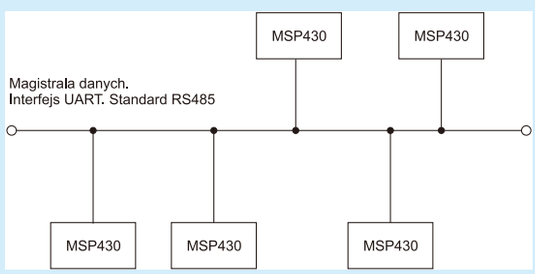
my uzyskać, a faktycznie uzyskaną wyniesie blisko 14%. Pierwszy bit w ramce transmisyjnej (bit startu) przesunie się o 14% swojego czasu trwania, kolejny o 28%, a ostatni, dziesiąty bit ramki transmisyjnej (ramka: 1 bit startu + 8 bitów danych + 1 bit stopu) aż o 140%. Przesunięcie bitów spowoduje, że dane będą wysyłane/odbierane niepoprawnie. Omówiony przypadek pokazano na **rysunku 6**.

Aby rozwiązać „problem" liczb zmiennoprzecinkowych i poprawić dokładność sygnału wyjściowego BITCLK, trzeba użyć zainstalowanego w generatorze taktującym (*Baud Rate Generator*) modulatora sygnału BITCLK. Dopasowuje on częstotliwość sygnału wyjściowego BITCLK (prędkość transmisji) do wartości optymalnej. W momencie, gdy czas trwania bitu jest zbyt krótki, to modulator wydłuża sygnał wyjściowy BITCLK o jeden takt sygnału wejściowego BRCLK. Efekt pracy modulatora pokazano na **rysunku 7**.

Żeby transmisja danych z włączonym modulatorem wykonywała się poprawnie należy obliczyć sekwencję modulującą, a obliczoną wartość wpisać do rejestru U0MCTL. W materiałach dodatkowych zamieszczamy program „Kalkulator", który służy do obliczania parametrów transmisji UART.

Konfigurowanie interfejsu UART i mikrokontrolera. Linie wejścia-wyjścia mikrokontrolera, do których dołączone są sygnały transmisyjne UTXD i URXD, ustawiamy w tryb pracy funkcyjny oraz definiujemy kierunek linii (UTXD wyjście, URXD wejście). Następnie konfigurujemy rejestry sterujące pracą kontrolera USART. Szablon procedury konfiguracyjnej pokazano w **ramce 8.1**. Dokumenty opisujące rejestry modułu zamieszczamy na płycie CD, oraz serwerze FTP.

Przerwania. Obsługę przerwań od danych odebranych RX włącza/wyłącza bit URXIE z rejestru IE2. Flaga przerwania URXIFG0 (rejestr IFG2) informuje o odebraniu danych. Jest ona ustawiana w momencie przepisania danych z rejestru przesuwnego do rejestru odbiornika U0RXBUF. Flaga jest zerowana w trakcie odczytu danych z rejestru U0RXBUF oraz przy wejściu do procedury obsługi przerwania (za wyjątkiem trybu wykrywania początku transmisji danych).



Rysunek 8. Przykład podłączenia MSP430, do magistrali danych

Obsługę przerwań danych wychodzących TX włącza/wyłącza bit UTXIE z rejestru IE2. Flaga przerwania UTXIFG0 znajduje się w rejestrze IFG2. Flaga przerwania jest ustawiana w momencie, gdy moduł transmisyjny jest gotowy do wysłania danych (pusty rejestr nadajnika U0TXBUF). Dodatkowo, w rejestrze U0TCTL umieszczono bit TXEPT, który jest ustawiany, gdy rejestr przesuwny nadajnika zostanie opróżniony.

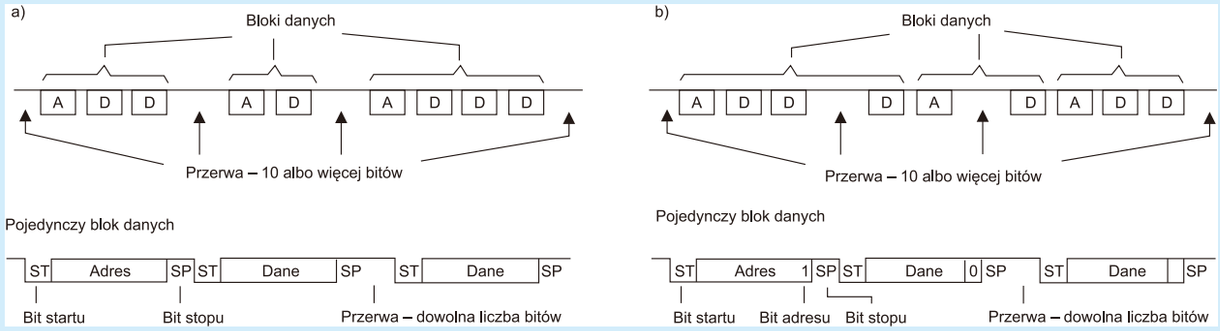
Tryb pracy wieloprocesorowej. Zazwyczaj interfejs UART jest używany do komunikacji pomiędzy dwoma urządzeniami. W praktyce budowane są także systemy, w których jest łączona ze sobą większa liczba urządzeń. Przykładem takiego systemu może być sieć czujników pomiarowych prezentowana na **rysunku 8**.

Aby urządzenia połączone w sieć mogły się ze sobą komunikować, każde z nich musi mieć unikalny adres. Dane wysyłane do urządzenia (pakiet/blok danych) muszą być zaadresowane. W mikrokontrolerach MSP430 interfejs UART wyposażono w mechanizm adresowania bloków danych. Jest to tzw. wieloprocesorowy tryb pracy interfejsu UART. W tym trybie dostępne są dwa protokoły komunikacji urządzeń: protokół *idle-line* oraz protokół z bitem adresowym. Oba konfigurowane są za pomocą bitu MM z rejestru U0CTL. (bit wyzerowany – *idle-line*, bit ustawiony – protokół z bitem adresowym). W obu protokołach minimalna wielkość bloku danych to dwie ramki transmisyjne UART. W pierwszej ramce jest umieszczany adres urządzenia, do którego kierowane są dane. W kolejnej (bądź kolejnych) są transmitowane dane. W protokole *idle-line* kolejne bloki danych oddzielane są przerwami w transmisji (trwającymi przez czas co najmniej 10 bitów) o wysokim poziomie logicznym, nadanymi po pierwszym bicie stopu. W pierwszej ramce bloku jest przesyłany adres, w pozostałych dane.

W protokole z bitem adresowym, do ramki transmisyjnej UART jest dodawany bit adresu ramki, którego wartość określa czy w ramce jest wysyłany adres, czy też dane. Jeśli bit adresu jest ustawiony (pierwsza ramka w bloku), to w ramce UART jest przesyłany adres, gdy bit jest wyzerowany, to dane. Na **rysunku 9** pokazano transmisję wieloprocesorową interfejsu UART w trybach obsługi protokołu *idle-line* oraz protokołu z bitem adresowym.

Wysyłając blok danych musimy go zaadresować. W trybie protokołu *idle-line* adres jest ustalany w dwóch krokach. Najpierw w rejestrze U0TCTL ustawiamy bit TXWAKE, a do rejestru nadajnika U0TXBUF wpisujemy dowolną wartość. Następnie, do rejestru nadajnika wpisujemy adres wysyłanego bloku. Dane wpisane do rejestru nadajnika w pierwszym kroku, nie są wysyłane, jest jedynie generowana przerwa na linii transmisyjnej. Pierwszą wysłaną ramką transmisyjną jest ramka z adresem bloku danych – w kolejnych możemy wysyłać dane. W trybie protokołu z bitem adresowym do ramki transmisyjnej jest dodawany bit adresu, który definiuje czy w ramce transmisyjnej wysyłany jest adres, czy dane (1 – adres, 0 – dane). Bit adresu ustawiamy za pomocą bitu TXWAKE. W trakcie wysyłania ramki danych bit TXWAKE jest przepisywany w pole bitu adresu.

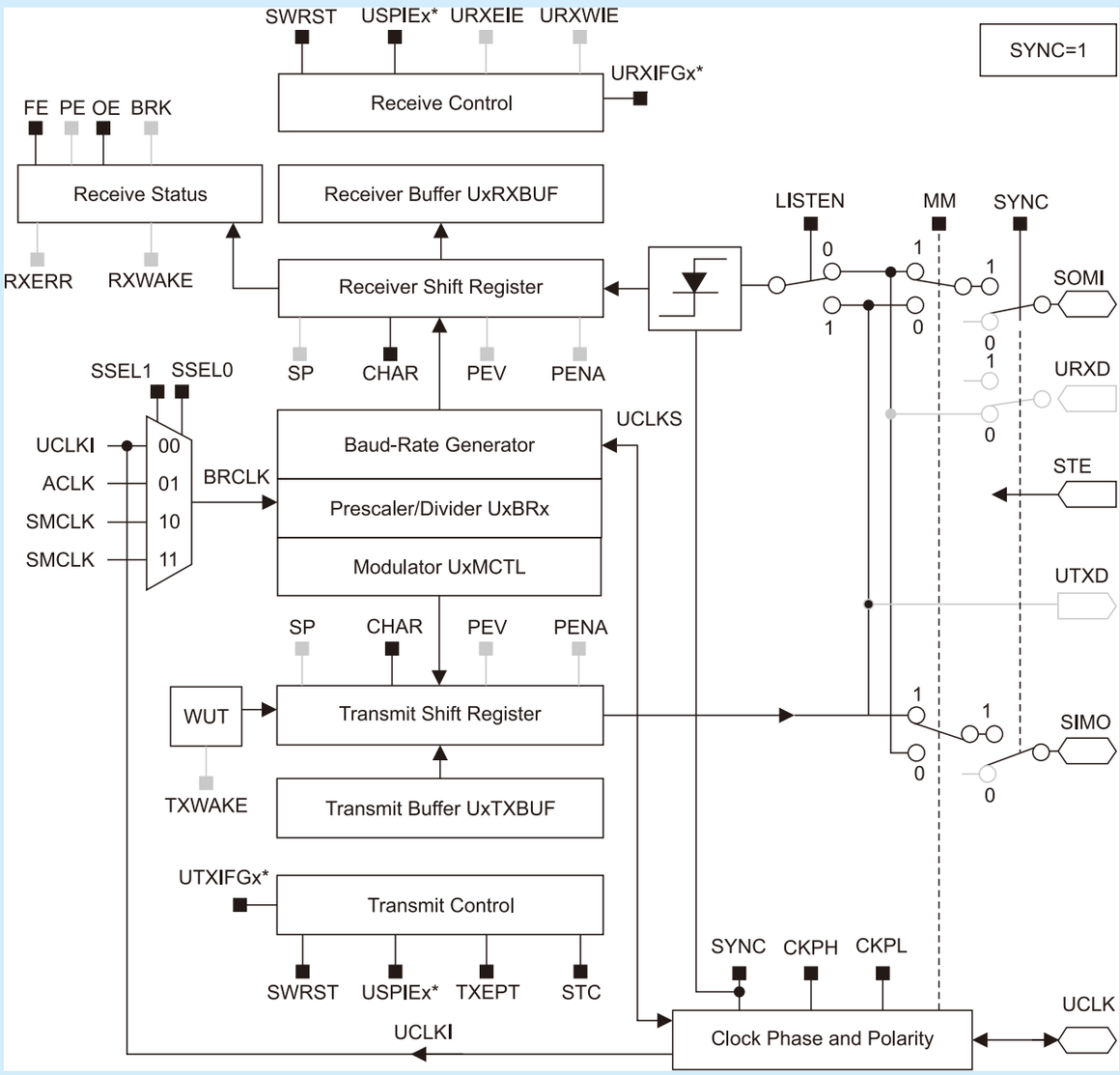
Odbieranie bloków danych jest identyczne dla obu protokołów. Przed odebraniem bloku danych należy



Rysunek 9. Transmisja wieloprocesorowa. Protokoły a) idle-line, b) z bitem adresowym

ustawić bit URXWIE w rejestrze U0RCTL. Wówczas wszystkie ramki transmisyjne, które nie będą zawierały adresu zostaną odrzucane przez MSP430. Gdy zostanie odebrana ramka z adresem (początek bloku danych), to w rejestrze U0RCTL zostanie ustawiany bit RXWAKE. Wówczas odczytujemy adres bloku danych i jeśli dane są adresowane do nas, to zerujemy bit URXWIE i odbieramy dane. Po odebraniu całego bloku danych ponownie ustawiamy bit URXWIE.

Wykrywanie początku transmisji. Moduł UART obsługuje mechanizm pozwalający na wykrycie początku transmisji danych przychodzących (bit URXSE w rejestrze U0TCTL). Pojawienie się na linii odbiorczej RXD ramki transmisyjnej powoduje ustawienie wewnętrznego sygnał URXS oraz wywołanie procedury obsługi przerwania RX (flaga przerwania URXIFG0 nie jest ustawiana). W procedurze obsługi przerwania sprawdzamy stan flagi URXIFG0. Gdy flaga jest ustawiona to oznacza, że zostały odebrane dane (ewentualnie błąd BRK). W przeciwnym razie wejście do procedury obsługi przerwania zostało wymuszone wykryciem początku transmisji danych przychodzących. W drugim wypadku zerujemy



Rysunek 10 Moduł USART. Interfejs SPI. Szarym kolorem zaznaczono bloki funkcjonalne odpowiedzialne za obsługę interfejsu UART

Tabela 8.1. Interfejs UART błędy danych przychodzących

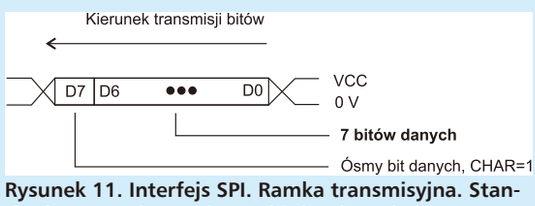
Typ błędu.	Opis błędu.
FE (błąd ramki)	Błąd ramki jest wykrywany, gdy bit stopu jest zerem. Jeśli ramka ma dwa bity stopu, to jest sprawdzany tylko pierwszy z nich.
OE (błąd przepełnienia rejestru U0RXBUF)	Błąd przepełnienia rejestru U0RXBUF jest zgłaszany, gdy w rejestrze odbiornika zostaną nadpisanie dane. Taka sytuacja ma miejsce, jeśli nie odczytamy z rejestru odebranego znaku i zostanie odebrany nowy znak.
BRK (błąd przerwania odbioru danych)	Błąd przerwania odbioru danych jest zgłaszany, gdy w ramce nie zostanie odebrany bit stopu, a następnie na linii URXD zostanie odebranych co najmniej 10 bitów o niskim poziomie logicznym.
PE (błąd parzystości)	Błąd parzystości. Przed wysłaniem ramki jest obliczana liczba 1 w przesyłanej wiadomości (dane + ew. bit adresu). Jeśli wynik jest liczbą nieparzystą, to bit parzystości jest ustawiany 1. W przeciwnym wypadku, bit parzystości jest zerowany. Ustawienie bitu kontrolnego powoduje, że liczba jedynek w wiadomości (dane + ew. bit adresu + bit kontrolny) w przypadku bitu parzystości jest parzysta, a bitu nieparzystości nieparzysta. Po odebraniu ramki danych ponownie jest obliczana liczba 1 w wiadomości. Jest obliczany bit kontrolny, a jego wartość porównywana z odebraną w ramce danych. Brak zgodności oznacza, że podczas transmisji danych wystąpiło przekłamanie i jest zgłaszany błąd parzystości.

oraz ponownie ustawiamy bit URXSE. Spowoduje to wyzerowanie wewnętrznego sygnału URXS oraz ponownie włączy mechanizm wykrywania początku transmisji danych odbieranych.

Mechanizm wykrywania początku transmisji danych odbieranych znalazł zastosowanie w aplikacjach obsługujących tryby uśpienia mikrokontrolera. Możemy uśpić MSP430, oszczędzać energię i czekać transmisję danych. Po wykryciu jej początku mikrokontroler jest budzony i może zająć się dalszą obsługą UART.

Sygnalizowanie błędu odbioru danych. Układ odbiornika UART potrafi wykryć błędy odbioru danych. Automatycznie wykrywane są: błąd w strukturze ramki (FE), przepełnienie rejestru odbiornika (OE), przerwanie odbioru danych (BRK). Dodatkowo, możemy włączyć mechanizm kontroli parzystości (PE). W rejestrze U0CTL ustawimy bit PENA. Wówczas do ramki transmisyjnej jest dodawany bit kontrolny. Za pomocą bitu PEV z rejestru U0CTL definiujemy czy będziemy sprawdzać parzystość, czy też nieparzystość bitów w ramce transmisyjne (0 – nieparzystość, 1 – parzystość). Wszystkie bity błędów ustawiane są w rejestrze U0RCTL, a ich znaczenie opisano w **tabeli 8.1**.

W momencie wykrycia błędu jest ustawiany odpowiedni bit (FE, OE, BRK, PE) oraz dodatkowo, jest ustawiany bit RXERR. Podczas odczytu danych z odbiornika (rejestr U0RXBUF) wszystkie bity błędów są zerowane automatycznie i dlatego bity błędów powinny być sprawdzane zanim odczytamy rejestr odbiornika. Najpierw sprawdzamy wartość bitu RXERR. Gdy bit będzie wyzerowany, to odebrane dane są poprawne. Ustawiony bit informuje, że wystąpił błąd odbioru danych. Wówczas należy sprawdzić pozostałe bity błędów i wykryć jego rodzaj.



Rysunek 11. Interfejs SPI. Ramka transmisyjna. Standardowa ramka zawiera 7 bitów danych

Moduł USART: interfejs synchroniczny SPI

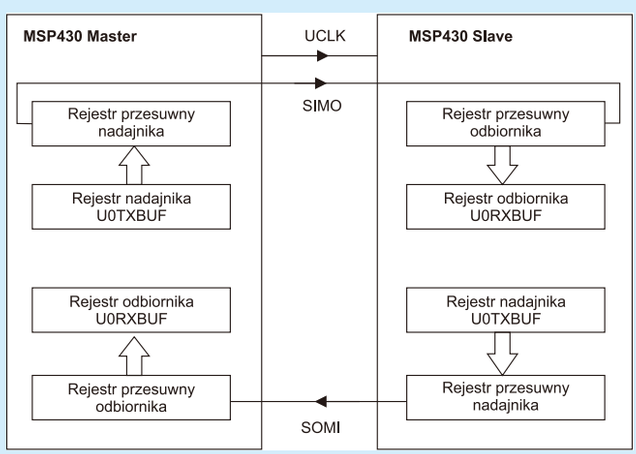
Interfejs SPI zaprojektowano do realizacji synchronicznej transmisji szeregowej. W mikrokontrolerach MSP430 może on pracować w trybie nadrzędnym (Master) lub podrzędnym (Slave). Tryb pracy interfejsu konfiguruje bit MM z rejestru U0CTL (1 – Master, 0 – Slave). Transmisja danych jest realizowana przy użyciu 3 linii: SIMO (*Slave In Master Out* – wejście Slave, wyjście Master), SOMI (*Slave Out Master In* – wyjście Slave, wejście Master), UCLK (*USART SPI Clock*). Linie SIMO i SOMI to linie danych, linia UCLK to linia sygnału zegarowego taktującego transmisję danych.

Transmisję danych zawsze inicjuje urządzenie Master, które generuje przebieg zegarowy UCLK taktujący transmisją danych. W mikrokontrolerach MSP430 możemy włączyć tryb pracy interfejsu SPI, w którym będzie używana dodatkowa, czwarta linia sterująca. Linia STE (*Slave Transmit Enable*) służy do rozwiązywania konfliktów podczas transmisji danych. Obsługę linii STE (tryb pracy 4-pinowy) włączamy zerując bit STC w rejestrze U0TCTL.

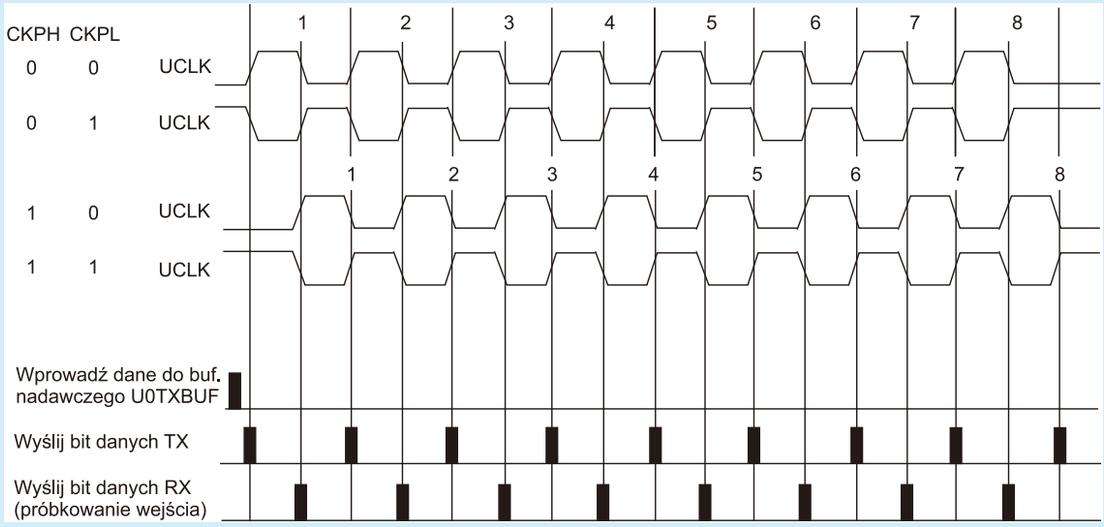
Budowa. Schemat blokowy modułu USART w trybie obsługi interfejsu SPI (bit SYNC=1) pokazano na **rysunku 10**. Podstawowe elementy modułu to: rejestry odbiornika (przesuwny i odbiorczy), rejestry nadajnika (przesuwny i nadawczy), linie interfejsowe (SIMO, SOMI, UCLK, STE).

Ramka transmisyjna. W ramce może być przesłanych 7, albo 8 bitów danych. Dane przesyłane są w kierunku od najbardziej do najmniej znaczącego bitu. Wygląd ramki transmisyjnej pokazano na **rysunku 11**.

Wysyłanie i odbieranie danych. Transmisja danych jest dwukierunkowa. Z każdym taktem sygnału taktującego transmisję (UCLK) jest wysyłany i odbierany jeden



Rysunek 12. Interfejs SPI. Transmisja danych. Z każdym taktem zegara UCLK (zbocze rosnące i opadające) wysyłany i odbierany jest jeden bit danych



Rysunek 13 Interfejs SPI. Polaryzacja i faza sygnału UCLK. Przebiegi czasowe sygnału

bit danych. Sposób przesyłanie danych zależy od tego czy mikrokontroler pracuje w trybie nadrzędnym (Master), czy podrzędnym (Slave). W sposób graficzny wysyłanie i odbieranie danych pokazano na **rysunku 12**.

W trybie Master transmisję bitów inicjuje wpisanie danych do rejestru nadajnik U0TXBUF. Po uzupełnieniu rejestru kontroler czeka, aż rejestr przesuwny nadajnika będzie pusty i gdy to nastąpi, kopiuje dane z rejestru nadajnika do rejestru przesuwnego. Wówczas jest ustawiana flaga przerwania UTXIFG0 informująca o tym, że rejestr nadajnika jest pusty i że można do niego zapisać kolejne dane do wysłania. Po załadowaniu rejestru przesuwnego na linii zegarowej mikrokontrolera pojawia się sygnał UCLK (wytwarzany przez generator taktujący), a narastające lub opadające zbocze sygnału (w zależności od wybranego trybu pracy interfejsu, o czym dalej) powoduje wysłanie bitu danych za pomocą linii wyjściowej SIMO. Przy zmianie zbocza sygnału zegarowego mikrokontroler próbkuje wejście danych odbieranych SOMI, a odczytany bit jest wpisywany do rejestru przesuwnego odbiornika. W momencie, gdy zostanie odebrana cała ramka danych (7 lub 8 bitów), to dane z rejestru przesuwnego odbiornika kopiowane są do rejestru odbiornika U0RXBUF. Wówczas jest ustawiana flaga przerwania URXIFG0 informująca o odebraniu danych.

W trybie Slave transmisję danych taktuje sygnał UCLK generowany przez dołączone urządzenie zewnętrzne. Dane po wpisaniu do rejestru nadajnika U0TXBUF przesyłane są do rejestru przesuwnego. W rejestrze przesuwnym czekają aż na linii zegarowej wystąpi sygnał UCLK. Z narastającym lub opadającym zboczem sygnału zegarowego (w wybranego trybu) bit danych jest wysyłany na linię transmisyjną SOMI. Po zmianie zbocza sygnału zegarowego mikrokontroler próbkuje linię danych przychodzących SIMO, a odczytany bit jest wpisywany do rejestru przesuwnego odbiornika. Gdy zostanie odebrana cała ramka danych (7 lub 8 bitów), to dane są kopiowane do rejestru odbiornika U0RXBUF i jest ustawiana flaga przerwania URXIFG0 informująca o odebraniu ramki danych. Jeśli nie odczytamy danych z rejestru odbiornika, to zostanie odebrana nowa ramka danych, a stare dane zostaną nadpisane, co spowoduje ustawienie flagi błędu OE w rejestrze U0RCTL.

Polaryzacja i faza sygnału zegarowego. W mikrokontrolerach MSP430 przyjęto inne niż w standardzie nazewnictwo bitów konfigurujących tryb transmisji SPI. Są to bity: CKPL, CKPH. Dodatkowo, bit CKPH (faza sygnału) działa inaczej, niż standardowo (odwrócone znaczenie). Dlatego też definiując parametry pracy sygnału zegarowego UCLK dla MSP430 nie można kierować się rutyną i doświadczeniem nabytym podczas stosowania innych mikrokontrolerów.

Bity CKPL i CKPH ustalające polaryzację i fazę sygnału UCLK umieszczono w rejestrze U0TCTL. Bit polaryzacji sygnału zegarowego określa poziom linii zegarowej w stanie spoczynku (przed rozpoczęciem transmisji):

- CKPL=0 – linia wyzerowana,
- CKPL=1 – linia ustawiona.

Bit fazy sygnału zegarowego definiuje zależność pomiędzy zboczem sygnału zegarowego, a momentem wysłania (transmisja bitu) i odebrania (próbkowanie linii wejścia) bitu danych:

- CKPH=0 – pierwsze zbocze sygnału zegarowego inicjuje wysłanie bitu danych, drugie zbocze wyznacza moment odbioru danych (próbkowanie wejścia),
- CKPH = 1 – pierwsze zbocze sygnału zegarowego wyznacza moment odbioru danych (próbkowanie wejścia), drugie zbocze inicjuje wysłanie bitu danych.

Modyfikując bity CKPL, CKPH uzyskujemy 4 tryby interfejsu SPI. Przebiegi czasowe sygnału UCLK pokazano na **rysunku 13**.

Prędkość transmisji. Transmisja danych jest taktowana sygnałem zegarowym UCLK. Prędkość transmisji danych wyrażana w bitach na sekundę (b/s) jest równa częstotliwości sygnału UCLK.

W trybie Master sygnał zegarowy UCLK jest generowany na podstawie BITCLK. Aby ustalić prędkość transmisji danych, konfigurujemy parametry pracy generatora taktującego (*Baud Rate Generator*). Ustawiamy źródło sygnału wejściowego BRCLK. Obliczamy wartość dzielnika częstotliwości BRCLK (wzór 8.2). Obliczoną wartość dzielnika sygnału wejściowego BRCLK wpisujemy do rejestrów U0BR1, U0BR0. Maksymalna prędkość transmisji danych (częstotliwość BITCLK) nie może być większa niż 1/2 częstotliwości sygnału BRCLK (wartość dzielnika musi być większa bądź równa 2). Układ modulatora nie jest używany, a rejestr U0MCTL należy wyzerować.

Ramka 8.2 Konfigurowanie kontrolera USART. Interfejs SPI
1. Włącz tryb restartu kontrolera (ustaw bit SWRST w rejestrze U0CTL).
2. Ustaw rejestry konfiguracyjne U0CTL, U0TCTL, U0RCTL. W trybie Master definiuj prędkość transmisji danych (rejestry U0BR1, U0BR0, U0MCTL = 0).
3. Włącz obsługę interfejsu SPI (ustaw bit USPIE0 w rejestrze ME2).
4. Wyłącz tryb restartu kontrolera (wyzeruj bit SWRST w rejestrze U0CTL).
5. Opcjonalnie włącz obsługę przerwań danych odbieranych/wysyłanych (ustaw bit/bity URXIE0, UTXIE0 w rejestrze IE2).

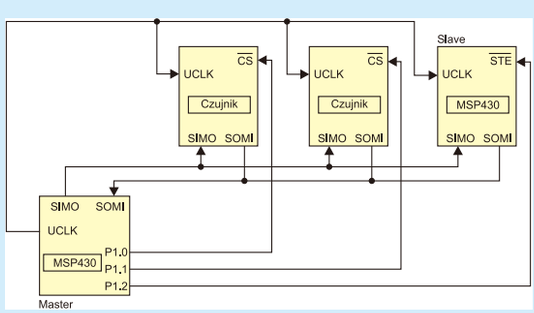
W trybie Slave sygnał zegarowy UCLK jest generowany przez urządzenie zewnętrzne i ono ustala również prędkość transmisji danych.

Konfigurowanie interfejsu SPI i mikrokontrolera. Linie I/O mikrokontrolera, do których są dołączone sygnały transmisyjne SOMI, SIMO, UCLK, STE, należy ustawić w tryb pracy funkcyjny. W trybie Master linie SIMO i UCLK konfigurujemy w kierunku wyjścia, a linię SOMI w kierunku wejścia. W trybie Slave linię SOMI przełączamy w kierunku wyjścia, a linie SIMO i UCLK w kierunku wejścia. Linie STE (tryb pracy 4-pinowy) zawsze ustawiamy w kierunku wejście. W trybie Master, jeśli linia STE jest wyzerowana, mikrokontroler ma zablokowaną transmisję SPI. W trybie Slave transmisję SPI blokuje poziom wysoki na linii STE. Graficznie pokazano to na **rysunku 14**.

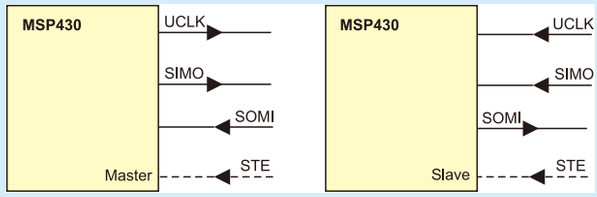
Po zakończeniu konfiguracji linii transmisyjnych konfigurujemy rejestry sterujące pracą kontrolera USART. Szablon procedury konfiguracyjnej umieszczono w **ramce 8.2**.

Budowa magistrali. Urządzenia komunikujące się przy pomocy interfejsu SPI można łączyć ze sobą tworząc magistrale komunikacyjne. Podstawowe konfiguracje to: magistrala z jednym urządzeniem Master, albo magistrala z wieloma urządzeniami Master. Częściej stosowany i prostszy w obsłudze jest wariant magistrali z jednym urządzeniem Master. Przykład takiego połączenia pokazano na **rysunku 15**. Ponieważ linie transmisyjne danych (SIMO, SOMI) są wspólne dla wszystkich urządzeń dołączonych do magistrali, to w jednej chwili mogą komunikować się tylko dwa urządzenia. (urządzenie Master i jedno z grupy urządzeń Slave). Aby zapobiec kolizjom na liniach transmisyjnych, Master wybiera urządzenie Slave, z którym chce „rozmawiać", a pozostałe wyłącza (blokuje interfejs SPI).

W MSP430 interfejs SPI włącza/wyłącza sygnał STE. W przypadku czujników sygnał sterujący pracą interfejsu SPI jest zazwyczaj w dokumentacji technicznej oznaczany CS (Chip Select). Przyjęto, że poziom niski na linii CS włącza komunikację SPI. W przykładzie prezentowanym na rys. 15 magistralę danych tworzą cztery urządzenia (dwa mikrokontrolery MSP430 oraz dwa czujniki temperatury). Układem Master jest mikrokontroler MSP430 pracujący w trybie 3-liniowym (w magistrali jest jeden Master, nie ma potrzeby stosowanie trybu 4-pinowego i użycia linii STE). Master korzysta z linii we-wy P1.0, P1.1, P1.2. Sterując wyjściami Master wybiera, z którym z trzech urządzeń Slave chce komunikować się. Poziom niski na linii sterującej włącza komunikację z wybranym urządzeniem Slave, wysoki blokuje komunikację. Żeby uniknąć kolizji na liniach danych tylko jedna z linii jest zerowana.



Rysunek 15. Interfejs SPI. Magistrala danych SPI (jeden Master)



Rysunek 14. Interfejs SPI. Kierunek linii transmisyjnych

Przerwania. W module USART Interfejs SPI korzysta z tych samych wektorów przerwań, co interfejs UART. Identycznie konfigurowana są źródła przerwań. Tak samo obsługiwane są flagi przerwań.

Przykłady

Zaprezentujemy dwa przykłady ilustrujące działanie układu USART w MSP430f1232. W przykładzie „Komunikacja mikrokontrolera z PC", korzystając z interfejsu UART, zrealizujemy wymianę danych pomiędzy MSP430 a komputerem PC. Będziemy wysłać polecenia sterujące pracą diody LED zainstalowanej w module „Komputerek". W przykładzie „Gra zręcznościowa" obsłużymy układ 3-osiowego akcelerometru LIS35DE. Z akcelerometrem będziemy komunikować się korzystając z interfejsu SPI. Dane odczytane z czujnika posłużą do sterowania ruchem samolotu w aplikacji gry zręcznościowej (akcelerometr pełni rolę joysticka – przechylenie czujnika w „lewo" oznacza skręt samolotu w lewo, przechylenie czujnika w „prawo" – skręt samolotu w prawo). Filmy ilustrujące działanie przykładów zostały zamieszczone w materiałach dodatkowych.

Przykład 1: komunikacja mikrokontrolera z PC. Program uruchamiamy korzystając z modułu „Komputerek". Zworki JP7, JP8 dołączające rezonator kwarcowy do źródła zegarowego LFXT1 należy ustawić w pozycji LF. Zworkę JP2 konfigurującą diodę należy ustawić w pozy-

Listing 8.2. Konfiguracja kontrolera USART. Interfejs SPI
```
U0CTL |= SWRST;              // włącz tryb restartu
kontrolera USART
                             // konfiguracja ramki
transmisyjnej
U0CTL  = CHAR + SYNC + MM;   // 8 bitów w ramce
                             // synchronizacja
transmisji, tryb Master
U0TCTL = CKPL + SSEL1 + SSEL0 + STC; // polaryzacja
UCLK- stan wysoki
                             // tryb pracy 3-liniowy
                             // sygnał BRCLK taktowany
zegarem SMCLK
U0BR0  = 0x02;               // ustaw prędkość transmisji
3 Mb/s
U0BR1  = 0x00;               // N = 6 MHz / 3Mb/s= 2
U0MCTL = 0x00;               // brak modulacji
ME2   |= USPIE0;             // włącz moduł transmisji
SPI
U0CTL &=~ SWRST;             // wyłącz tryb restartu
kontrolera USART
```

Tabela 8.2. „Komunikacja mikrokontrolera z PC". Instrukcje sterujące

Instrukcja (PC)	Odpowiedź (MSP430)	Opis
linia=0<CR>	<CR><LF> ustawiono 0 <CR><LF>	Ustaw poziom niski na linii P2.3 (wyłącz diodę LED)
linia=1<CR>	<CR><LF> ustawiono 1 <CR><LF>	Ustaw poziom wysoki na linii P2.3 (włącz diodę LED)
linia=?<CR>	<CR><LF> stan 0 <CR><LF> albo <CR><LF> stan 1 <CR><LF>	Odczytaj poziom linii P2.3 (niski / wysoki) (odczytaj status diody LED – wyłączona / włączona)
w przypadku niepoprawnej instrukcji zwracany jest komunikat: <CR><LF>błąd<CR><LF>		

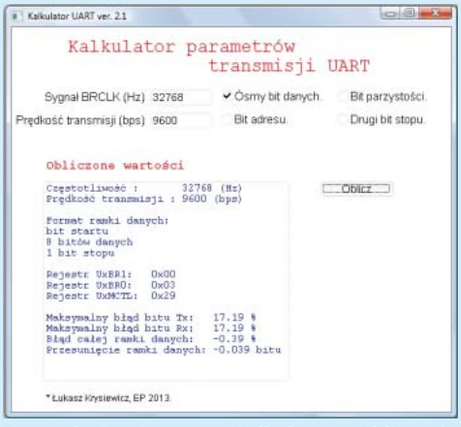
Listing 8.1. Konfiguracja kontrolera USART. Interfejs UART

```
U0CTL |= SWRST;     // włącz tryb restartu kontrolera
USART
                    // konfiguracja ramki transmisyjnej
U0CTL |=  CHAR;     // bit startu, 8 bitów danych, bit
stopu
U0TCTL =  SSEL0;    // sygnał BRCLK taktowany zegarem ACLK
                    // ustaw prędkość transmisji 9600 b/s
U0BR0   = 0x03;     // (wzór: 8.2 | aplikacja kalkulator)
U0BR1   = 0x00;     // N = 32768 Hz/9600 b/s = 3.41
U0MCTL = 0x29;      // modulacja = 0x29
ME2    |=  (UTXE0 + URXE0);  // włącz moduł nadawczy
                    // włącz moduł odbiorczy
U0CTL &=~ SWRST;    // wyłącz tryb restartu kontrolera
USART
IE2    |=  URXIE0;  // włącz obsługę przerwań
                    // danych przychodzących RX
```

cji LED. Zworkę JP12 dołączającą linię transmisji danych przychodzących interfejsu UART należy ustawić w pozycji RxD. Pozostałe zworki układu należy ustawić w pozycji IO/Off. Moduł „Komputerek" podłączamy do portu COM komputera PC.

Pliki źródłowe programu zamieszczono w materiałach dodatkowych. W programie głównym mikrokontroler oczekuje na polecenia wysyłane przez PC. Odbierane bajty danych wpisywane są do bufora cyklicznego. W momencie wykrycia znaku końca polecenia (znak CR) odebrane znaki odczytywane są z bufora. Analizowana jest treść polecenia. Jeśli instrukcja nie zawiera błędów to polecenie jest wykonywane, a MSP430 odsyła informacje o statusie wykonania polecenia. Listę obsługiwanych poleceń zamieszczono w **tabeli 8.2**.

Procedurę konfiguracji modułu USART w tryb pracy interfejsu UART pokazano na **listingu 8.1**. Modulacja (rejestr U0MCTL) i prędkość transmisji (rejestry U0BR1, U0BR0) zostały obliczone przy użyciu programu „Kalkulator". Działanie programu prezentuje **rysunek 16**.



Rysunek 16. Program „Kalkulator". Obliczenie parametrów modulacji i prędkość transmisji UART

Fotografia 17. „Gra zręcznościowa".

nych wynosi 9600 b/s. Generator taktujący (*Baud Rate Generator*) jest taktowany sygnałem zegarowym ACLK o częstotliwości 32768 Hz. W ramce transmisyjnej są przesyłane: 1 bit startu, 8 bitów danych oraz 1 bit stopu. Odbieranie danych jest obsługiwane z użyciem przerwań.

Przykład 2: gra zręcznościowa. Program „Gra zręcznościowa" uruchamiamy korzystając z modułu „Komputerek". Zworki JP7 i JP8 dołączające rezonator kwarcowy do źródła zegarowego LFXT1 należy ustawić w pozycji HF. Pozostałe zworki układu należy ustawić w pozycji IO/Off. W złączu szpilkowym Dis1 należy zamontować wyświetlacz LCD, a do złącza Con9 (SPI) podłączyć trzyosiowy akcelerometr LIS35DE (w przykładzie użyto modułu startowego KaModMEMS2 firmy Kamami). Czytelnicy, którzy nie mają akcelerometru, a chcą uruchomić grę, mogą zaprogramować moduł „Komputerek" programem w wersji ze sterowaniem ruchu samolotu za pomocą przycisków SW1 i SW2 (procedura *GraCzytajRuchSW*).

Pliki źródłowe programu zamieszczono na płycie CD i serwerze FTP. W programie głównym mikrokontroler cyklicznie odczytuje dane z akcelerometru. W zależności od położenie czujnika jest zmieniana pozycja samolotu na planszy gry. Celem gry jest omijanie przeszkód i unikanie kolizji. Przykład działania modułu „Komuterek" z uruchomioną aplikacją „Gra zręcznościowa" pokazano na **fotografii 17**. Procedurę konfiguracji modułu USART w tryb pracy interfejsu SPI zamieszczono na **listingu 8.2**.

Mikrokontroler pracuje w trybie Master (akcelerometr to urządzenie Slave). Włączono 3-pinowy tryb pracy mikrokontrolera. Linia CS urządzenia Slave sterowana jest przez MSP430 za pomocą wyjścia P3.0. W stanie bezczynności linia zegarowa UCLK pozostaje na poziomie wysokim. Prędkość transmisji danych wynosi 3 Mb/s.

Łukasz Krysiewicz, EP