Programowanie mikrokontrolerów

Magistrala I^2C

Marcin Engel Marcin Peczarski

Instytut Informatyki Uniwersytetu Warszawskiego

6 stycznia 2012

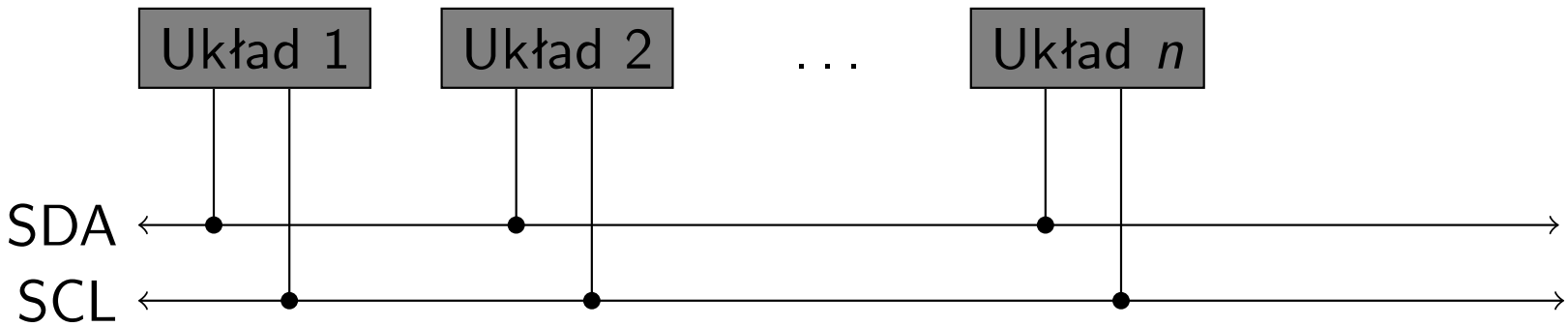
Magistrala I^2C

- Jest akronimem Inter-Intergrated Circuit.
- Została opracowana w latach osiemdziesiątych przez Philipsa.
- Magistrala I^2C składa się z dwóch dwukierunkowych linii:
  - linii danych, SDA,
  - linii zegara, SCL.
- Transmisja danych odbywa się szeregowo i synchronicznie.
- Do szyny może być przyłączonych wiele układów.
- Identyfikacja układu odbywa się za pomocą jego adresu sprzętowego.

Warianty magistrali I^2C

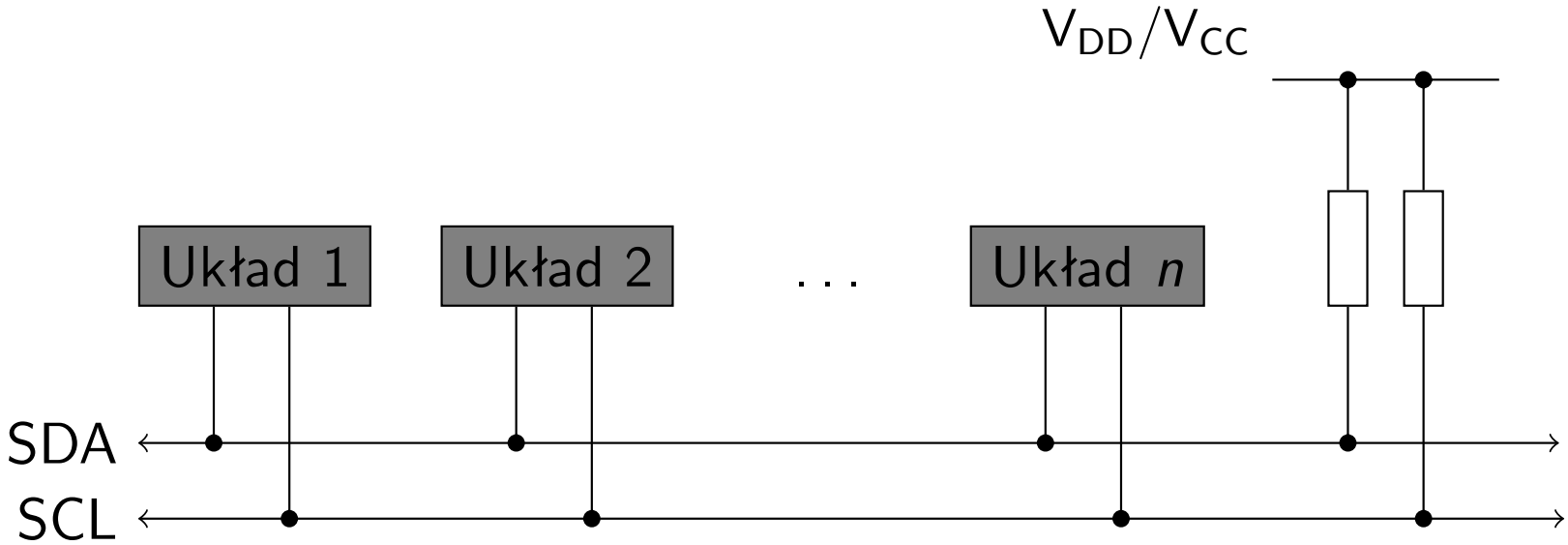
- Powstało kilka wersji I^2C:
  - wersja podstawowa z prędkością do 100 kb/s i 7-bitowymi adresami sprzętowymi (1982),
  - wersja 1.0 (fast mode) z prędkością do 400 kb/s i 10-bitową przestrzenią adresową (1992),
  - wersja 2.0 (high speed mode) z prędkością do 3,4 Mb/s i rozszerzonym zakresem napięć (1998),
  - wersja 2.1 (2000),
  - wersja Rev. 03 (2007).
- Skupimy się na wersji podstawowej.

Podłączenia elektryczne



- Wyjścia układów są typu otwarty dren/kolektor.
- Aby uzyskać na linii zero, należy podać stan niski.
- Aby uzyskać na linii jedynkę, wyjście musi być w stanie wysokiej impedancji, czyli musi być odłączone od linii.

Podłączenia elektryczne



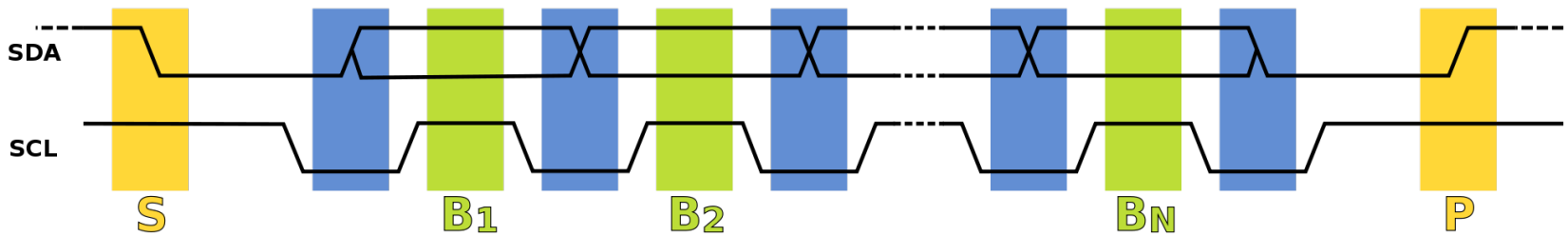
- Wyjścia układów są typu otwarty dren/kolektor.
- Aby uzyskać na linii zero, należy podać stan niski.
- Aby uzyskać na linii jedynkę, wyjście musi być w stanie wysokiej impedancji, czyli musi być odłączone od linii.
- Aby linie nie „wisiały w powietrzu”, dołącza się zewnętrzne rezystory podciągające.
- Stan linii jest iloczynem logicznym stanów wszystkich wyjść.
- Ich wartości zależą od liczby przyłączonych układów, szybkości transmisji i innych parametrów.

I^2C, tryby pracy

- Każdy układ może pracować jako:
  - master (M) — inicjuje komunikację i generuje sygnał zegara,
  - slave (S) — reaguje na dane pojawiające się na szynie.
- Każdy układ może:
  - nadawać (T) — umieszczając dane na szynie,
  - odbierać (R) — odczytując dane z szyny.
- Mamy więc cztery tryby pracy: MR, MT, SR i ST.
- W dalszym ciągu zakładamy, że tylko jeden układ pracuje w trybie master, choć protokół radzi sobie (przy pewnych dodatkowych założeniach) także z konfiguracjami z wieloma układami master.

Protokół komunikacyjny

- Tranmisja danych polega na nadaniu:
  - sygnału START,
  - 9-bitowej ramki z adresem sprzętowym układu slave,
  - pewnej liczby 9-bitowych ramek z danymi,
  - sygnału STOP.



Inicjacja transmisji

- Transmisję inicjuje master, nadając sygnał START.
- Od tej chwili szyna jest w stanie zajętości.
- Master zwalnia szynę, nadając sygnał STOP.
- Master może zainicjować kolejną transmisję bez zwalniania szyny, ponownie wysyłając sygnał START (tzw. REPEATED START).

Transmisja pojedynczej ramki

- Master generuje sygnał zegarowy na linii SCL.
- W ośmiu kolejnych cyklach nadawca przesyła kolejne bity od najstarszego do najmłodszego.
- W dziewiątym cyklu odbiorca generuje sygnał ACK lub NACK.

Format ramki z adresem

- Składa się z 9 bitów:
  - 7-bitowy sprzętowy adres urządzenia slave,
  - bit READ/WRITE:
    - 1 oznacza odczyt z urządzenia slave,
    - 0 oznacza zapis do urządzenia slave.
  - bit potwierdzenia — slave potwierdza odbiór własnego adresu, generując sygnał ACK w dziewiątym cyklu zegara (SCL).
- Za generowanie sygnału zegara odpowiada master.

Format ramki z danymi

- Składa się z 9 bitów:
  - 8 bitów danych wysyłanych przez nadawcę,
  - bit potwierdzenia generowany przez odbiorcę, przyjmujący wartości:
    - ACK, jeśli odbiorca otrzymał dane i jest gotowy na następne,
    - NACK, jeśli odbiorca nie może lub nie chce odbierać kolejnych danych.
- Za generowanie sygnału zegara odpowiada master.

Warstwa fizyczna

- Gdy szyna jest wolna, obie linie są w stanie wysokim.
- Sygnał START polega na zmianie poziomu linii SDA z wysokiego na niski, przy jednoczesnym wysokim stanie linii SCL.
- Poszczególne bity są ustawiane w fazie niskiej linii SCL i muszą pozostawać stabilne w fazie wysokiej.
- Sygnał ACK polega na ustawieniu niskiego poziomu linii SDA w czasie dziewiątego cyklu zegara.
- Sygnał NACK polega na pozostawieniu wysokiego poziomu linii SDA w czasie dziewiątego cyklu zegara.
- Sygnał STOP polega na zmianie poziomu linii SDA z niskiego na wysoki, przy jednoczesnym wysokim stanie linii SCL.

I^2C w ATmega16 i ATmega32

- Mikrokontrolery ATmega16 i ATmega32 mają wbudowany moduł Two-Wire Serial Interface (TWI), który:
  - realizuje komunikację po I^2C,
  - zwalnia programistę z konieczności implementacji tej komunikacji na poziomie sygnałów na liniach SCL i SDA.
- Gdy moduł TWI jest włączony, sygnał SCL jest wyprowadzony na pin PC0, a sygnał SDA jest wyprowadzony na pin PC1.
- Można aktywować (rejestr PORTC) wewnętrzne rezystory podciągające na nogach PC0 i PC1 i w pewnych warunkach używać ich zamiast rezystorów zewnętrznych.

Sygnał zegara

- Jeśli mikrokontroler pracuje w trybie slave:
  - nie generuje sygnału na SCL,
  - częstotliwość zegara systemowego musi być co najmniej 16 razy większa niż częstotliwość sygnałów na linii SCL.
- Jeśli mikrokontroler pracuje w trybie master:
  - częstotliwość na linii SCL wynosi

$$\frac{\text{clk}}{16 + 2 \cdot \text{TWBR} \cdot 4^{\text{TWPS}}},$$

  - clk – częstotliwość zegara systemowego,
  - TWBR – wartość rejestru TWBR,
  - TWPS – wartości bitów preskalera.

Rejestr TWCR

Two-Wire Interface Control Register

7	6	5	4	3	2	1	0
TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	–	TWIE

- Bit 7:
  - jest ustawiany sprzętowo po zakończeniu operacji na magistrali I^2C (ale nie po sygnale STOP!),
  - nigdy nie jest automatycznie zerowany,
  - jego wyzerowanie (przez wpisanie jedynki) inicjuje kolejną operację na magistrali.
- Bit 6 — ustawienie na jeden powoduje automatyczne generowanie sygnału ACK, gdy:
  - urządzenie odczyta własny adres w trybie slave,
  - urządzenie otrzyma bajt danych w trybie odbioru.

Rejestr TWCR, cd.

Two-Wire Interface Control Register

7	6	5	4	3	2	1	0
TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	–	TWIE

- Bit 5 — ustawienie na jeden powoduje przełączenie w tryb master:
  - dopóki szyna jest zajęta, mikrokontroler czeka na sygnał STOP,
  - następnie generuje sygnał START,
  - Bit musi zostać wyzerowany programowo (w zwykły sposób).
- Bit 4:
  - w trybie master ustawienie na jeden powoduje wygenerowanie sygnału STOP i automatyczne wyzerowanie bitu,
  - w trybie slave przełącza SCL i SDA do stanu wysokiej rezystancji.

Rejestr TWCR, cd.

Two-Wire Interface Control Register

7	6	5	4	3	2	1	0
TWINT	TWEA	TWSTA	TWSTO	TWWC	TWEN	–	TWIE

- Bit 3 — ustawiany przy próbie zapisu do TWDR, gdy TWINT jest w stanie niskim, zerowany po zapisie do TWDR, gdy TWINT jest w stanie wysokim.
- Bit 2 — ustawienie powoduje uaktywnienie interfejsu I^2C i odłączenie wyprowadzeń PC0, PC1 od portu C.
- Bit 0 — włącza (przy ustawionym znaczniku I w SREG) przerwanie o adresie symbolicznym `TWIaddr`. Przerwanie jest aktywne tak długo, jak długo TWINT jest ustawiony.

Rejestr TWSR

Two-Wire Interface Status Register

7	6	5	4	3	2	1	0
TWS					–	TWPS1	TWPS0

- Bity 7 do 3 — kod błędu, zależny od trybu pracy, szczegóły w dokumentacji producenta.
- Bity 2 — zarezerwowany.
- Bity 1, 0 — bity preskalera.

Rejestr TWDR

- Two-Wire Data Register
- W trybie nadawania zawiera następne dane (lub adres) do wysłania.
- W trybie odbioru zawiera ostatnio odebrane dane (lub adres).
- Można do niego pisać, gdy TWINT zostanie sprzętowo ustawiony na jedynkę.
- Zatem nie można go zainicjować przed rozpoczęciem transmisji po magistrali!

Rejestr TWAR

Two-Wire Address Register

7	6	5	4	3	2	1	0
TWA							TWGCE

- Jeśli mikrokontroler pracuje w trybie slave zawiera adres mikrokontrolera.
- Adresy postaci 0 i 1111xxx są zarezerwowane i nie należy ich używać.
- Bit 0 jest ustawiany, gdy mikrokontroler ma reagować na tzw. General Call, czyli adres 0.

Typowa sekwencja — mikrokontroler w trybie MT

- Włączamy I^2C, ustawiamy TWSTA i inicjujemy transmisję, zerując TWINT — zostanie wysłany sygnał START:

```
ldi r16, 1<<TWINT | 1<<TWSTA | 1<<TWEN
out TWCR, r16
```

- Czekamy na gotowość:

```
czekaj1:
  in r16, TWCR
  sbrs r16, TWINT
  rjmp czekaj1
```

- Sprawdzamy poprawność:

```
in r16, TWSR
andi r16, 0xF8
cpi r16, MT_START; 0x08
brne ERROR
```

Typowa sekwencja, cd.

- Wysyłamy adres sprzętowy (SLA_W):

```
ldi r16, SLA_W
out TWDR, r16
ldi r16, 1<<TWINT | 1<<TWEN
out TWCR, r16
```

- Czekamy na gotowość:

```
czekaj2:
  in r16, TWCR
  sbrs r16, TWINT
  rjmp czekaj2
```

- Sprawdzamy poprawność (czy slave wygenerował ACK):

```
in r16, TWSR
andi r16, 0xF8
cpi r16, MT_SLA_ACK; 0x18
brne ERROR
```

Typowa sekwencja, cd.

- Wysyłamy bajt danych (`DATA`):

```
ldi r16, DATA
out TWDR, r16
ldi r16, 1<<TWINT | 1<<TWEN
out TWCR, r16
```

- Czekamy na gotowość:

```
czekaj3:
  in r16, TWCR
  sbrs r16, TWINT
  rjmp czekaj3
```

- Sprawdzamy poprawność (czy slave wygenerował ACK):

```
in r16, TWSR
andi r16, 0xF8
cpi r16, MT_DATA_ACK; 0x28
brne ERROR
```

Typowa sekwencja, cd.

- Generujemy sygnał STOP:

```
ldi r16, 1<<TWINT | 1<<TWEN | 1 << TWSTO
out TWCR, r16
```

Przykłady układów komunikujących się po I^2C

- różne rodzaje pamięci EEPROM,
- zegary czasu rzeczywistego (np. DS1307, PCF8583, . . .),
- ekspandery wejść-wyjść (np. PCF8574),
- konwertery a/c i c/a (np. PCF8591),
- . . .

I^2C w zestawach

- Wyprowadzenia PC0 i PC1 są połączone ze złączem I^2C.
- Za pomocą zworek JSDA i JSCL można dołączyć zewnętrzne rezystory podciągające 4,7 kΩ.
- Na płycie znajduje się układ DS1307 (zegar czasu rzeczywistego):
  - taktowany kwarcem 32,768 kHz,
  - podtrzymywany akumulatorkiem 3,6 V, doładowywanym przy włączonym zasilaniu i założonej zworce LOAD,
  - wyprowadzenia SDA, SCL i FT wyprowadzone na szpilki w grupie MISC i podciągane zewnętrznymi rezystorami 4,7 kΩ.

Zegar czasu rzeczywistego DS1307

- Zlicza sekundy, minuty, godziny, dni miesiąca, miesiące, dni tygodnia, lata (z uwzględnieniem lat przestępnych do 2100).
- Ma 56 bajtową pamięć.
- Dostarcza sygnału prostokątnego o programowalnej częstotliwości (1 Hz, 4 kHz, 8 kHz lub 32 kHz).
- Zużywa 500 nA przy podtrzymywaniu bateryjnym.
- Zalecane napięcie zasilania: 4,5 V – 5,5 V.
- Zalecane napięcie podtrzymania: 2,0 V – 3,5 V.
- Częstotliwość zegara na linii SCL: do 100 kHz.

Pamięć RAM

Adres	b7	b6	b5	b4	b3	b2	b1	b0
00	CH	sek. dz.			sek. j.			
01	0	min. dz.			min. j.			
02	0	12/24	PM/AM	godz. dz.	godz. j.			
03	0	0	0	0	0	dzień. tyg.		
04	0	0	dzień mies. dz.		dzień mies. j.			
05	0	0	0	miesiąc dz.	mies. j.			
06	rok dz.				rok j.			
07	OUT	0	0	SQWE	0	0	RS1	RS0
08–3F	RAM							

Rejestry RTC

- Po włączeniu zasilania są w nieokreślonym stanie.
- Są w formacie BCD.
- Dzień tygodnia zwiększa się o północy.
- Próba ustawienia niepoprawnego czasu lub daty daje nieokreślony wynik.
- Ustawienie bitu CH wyłącza oscylator.
- Ustawienie bitu 12/24 włącza tryb 12 godzinny.
- W trybie 12 godzinnym ustawiony bit PM/AM oznacza PM; w trybie 24 godzinnym jest wykorzystywany do kodowania cyfry dziesiątek godziny.
- Po zmianie formatu należy ponownie ustawić godzinę.
- Data i czas są odczytywane z rejestrów pomocniczych, synchronizowanych z rzeczywistymi po sygnale START.
- DS1307 ma wewnętrzny rejestr pamiętający adres ostatniej operacji.

Rejestr sterujący

- Jeśli SQWE (bit 4) jest ustawiony, na wyprowadzeniu FT jest generowany przebieg prostokątny, którego częstotliwość zależy od bitów RS1 i RS0.
- Jeśli SQWE jest wyzerowany i OUT jest ustawiony, na wyprowadzeniu FT jest stan wysoki.
- Jeśli SQWE jest wyzerowany i OUT jest wyzerowany, na wyprowadzeniu FT jest stan niski.
- Stan wyprowadzenia FT jest określony poniższą tabelą:

RS1	RS0	SQWE	OUT	FT
0	0	1	X	1 Hz
0	1	1	X	4096 Hz
1	0	1	X	8192 Hz
1	1	1	X	32768 Hz
X	X	0	0	0
X	X	0	1	1

DS1307 w trybie Slave Receive — zapis do RTC

- Master (mikrokontroler) generuje sygnał START.
- Master wysyła adres sprzętowy układu DS1307, który wynosi 0b1101000 uzupełniony do ośmiu bitów zerem (zapis).
- W dziewiątym cyklu zegara SCL układ DS1307 generuje ACK.
- Master wysyła 8-bitowy adres komórki pamięci DS1307.
- W dziewiątym cyklu zegara SCL układ DS1307 generuje ACK.
- Master wysyła dowolną liczbę bajtów:
  - kolejne bajty są zapisywane w kolejnych komórkach pamięci DS1307, począwszy od przesłanego adresu,
  - wewnętrzny rejestr adresu jest zwiększany automatycznie po każdym zapisie,
  - DS1307 generuje ACK po odebraniu każdego bajtu.
- Master generuje sygnał STOP.

DS1307 w trybie Slave Transmitter — odczyt RTC

- Master (mikrokontroler) generuje sygnał START.
- Master wysyła adres sprzętowy układu DS1307, który wynosi 0b1101000 uzupełniony do ośmiu bitów jedynką (odczyt).
- W dziewiątym cyklu zegara SCL układ DS1307 generuje ACK.
- DS1307 wysyła 8-bitową wartość spod adresu pamiętanego w rejestrze adresu.
- W dziewiątym cyklu zegara master generuje ACK, jeśli chce otrzymać kolejne dane lub NACK, jeśli chce zakończyć transmisję.
- Jeśli master wygenerował ACK, to DS1307 zwiększa rejestr adresu i przesyła kolejny bajt.
- W przeciwnym przypadku master musi wygenerować sygnał STOP.

Odczyt danych spod konkretnego adresu

- Master generuje sygnał START.
- Następnie wysyła adres sprzętowy DS1307 uzupełniony zerem oraz adres komórki pamięci — DS1307 pracuje w trybie Slave Receiver.
- Master generuje sygnał REPEATED START.
- Następnie wysyła adres sprzętowy DS1307 uzupełniony jedynką — DS1307 pracuje w trybie Slave Transmitter.
- Następnie protokół odbywa się jak z DS1307 w trybie Slave Transmitter.
- Na koniec jest tylko jeden sygnał STOP!

I^2C w VMLAB

- W pliku projektu można umieścić monitor I^2C:

```
X1  I2C(100k 104) PC1 PC0
```

- Trzeba umieścić rezystory podciągające:

```
R1 PC0 VDD 4700
R2 PC1 VDD 4700
```

- Monitor I^2C analizuje i wyświetla dane napływające po magistrali I^2C, może też wysyłać wpisane bajty.