

Andrzej Pawluczuk

Mikrokontrolery AVR

*techniczne
aspekty
programowania*

Białystok, 2004

Mikrokontrolery rodziny AVR integrują w swojej strukturze między innymi nieulotną pamięć przeznaczoną na program (pamięć FLASH) oraz nieulotną pamięć EEPROM do dowolnego zastosowania przez autora programu. Po napisaniu programu powstaje problem związany z umieszczeniem jego kodu wewnątrz pamięci mikrokontrolera. Producent (firma ATMEL) przewidział w ogólnym przypadku dwa rozwiązania dotyczące sposobu programowania pamięci FLASH oraz EEPROM (w niektórych przypadkach, dotyczy to mikrokontrolerów ATMEGA, istnieje trzecia możliwość związana z mechanizmem BOOT LOADER). Programowanie mikrokontrolerów może odbywać się:

1. w programatorze,
2. w układzie.

W pierwszym przypadku proces programowania wymaga włożenia programowanego mikrokontrolera do programatora, zaprogramowaniu układu i ponownego przeniesienia mikrokontrolera do układu, w którym ma on pracować. Takie rozwiązanie ma niedogodność związaną ciągłym przenoszeniem mikrokontrolera między programatorem a docelowym systemem. Drugą poważną niedogodnością tego rozwiązania jest to, że z praktycznego punktu widzenia nie istnieje możliwość użycia wybranych modeli mikrokontrolerów (ponieważ przykładowo ATMEGA103, ATMEGA128 i inne są produkowane wyłącznie w obudowach do montażu powierzchniowego).

W drugim przypadku proces programowania wymaga umieszczenia w docelowym systemie odpowiedniego złącza, przez które będzie odbywać się programowanie (kasowanie, odczytywanie pamięci, weryfikacja itp.). Metoda ta nazywa się programowaniem w trybie szeregowym i jest tematem tych rozważań. W tym trybie dostępne są operacje kasowania, programowania i odczytu pamięci FLASH i EEPROM oraz istnieje możliwość modyfikacji bitów zabezpieczeń i bitów konfiguracji (ang. fuse).

Firma ATMEL zaproponowała standard złącza programującego w dwóch wariantach, jako złącze 6-stykowe oraz 10-stykowe (przykładowo zestaw uruchomieniowy STK500 posługuje się takim złączem). Oba rozwiązania funkcjonalnie są identyczne.



Złącza programujące są następujące (rysunek 1):

gdzie:

MOSI, MISO, SCK i RESET są sygnałami służącymi do

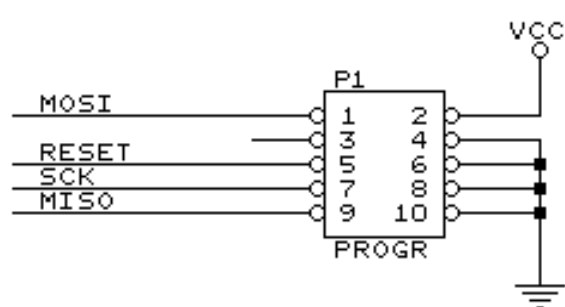
Rysunek 1. Złącze programujące

programowania (są opisane niżej), GND jest sygnałem masy oraz VTG jest napięciem zasilającym.

Z praktycznego punktu widzenia sprowadza się to do umieszczenia na płycie drukowanej docelowego systemu złącza pinowego (najczęściej są to dwurzędowe listy pinowe o rozstawie 100 mils (2.54mm)). Do połączenia z programatorem używa się kabla płaskiego o zaciśniętą na końcu złączką nakładaną na piny będące w płycie drukowanej systemu. Ponieważ na naszym rynku bez problemów są dostępne złącza 10-pinowe dalsze rozważania będą dotyczyć tego wariantu.

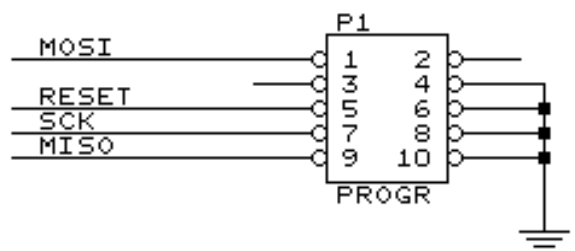
Występujące w złączu do szeregowego programowania sygnały to:

- RESET - sygnał zerowania mikrokontrolera,
- MOSI (od ang. **M**aster **O**utput **S**lave **I**nput) – dane transmitowane z programatora do mikrokontrolera,
- MISO (od ang. **M**aster **I**nput **S**lave **O**utput) – dane transmitowane z mikrokontrolera do programatora,
- SCK – sygnał zegarowy do programowania,
- GND – masa zasilania.



Rysunek 2. Schemat złącza programującego

W ogólnym przypadku złącze programujące wygląda następująco (rysunek 2 oraz rysunek 3).

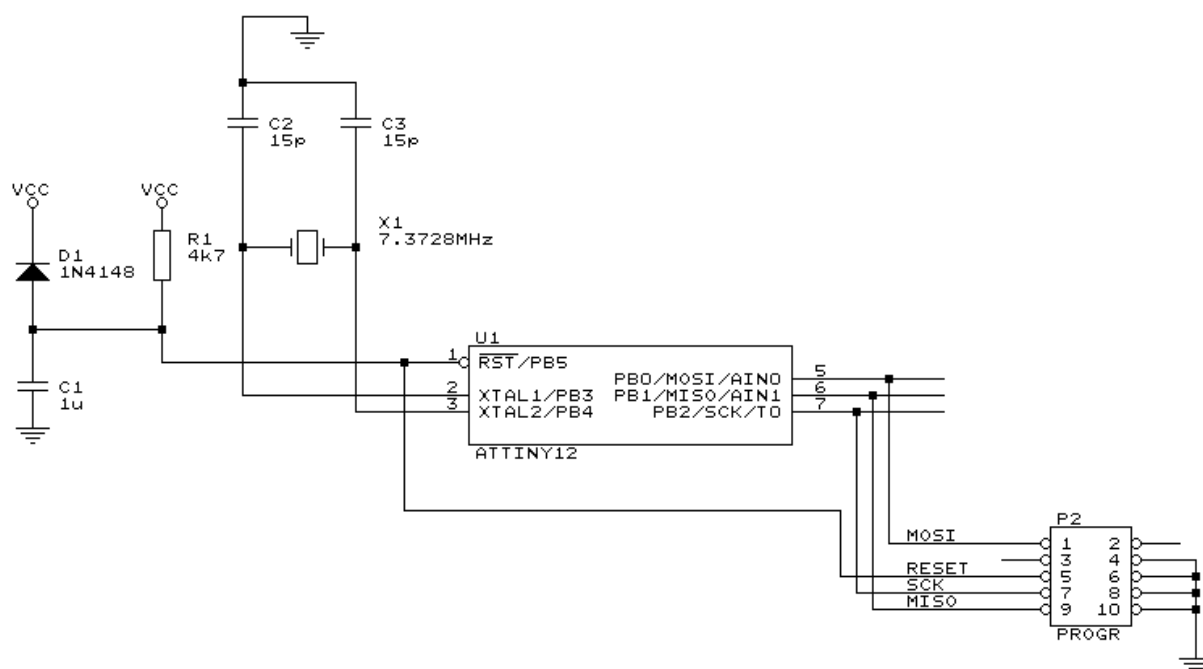


Rysunek 3. Schemat złącza programującego - wariant uproszczony

Występujące tu połączenie złączki z napięciem V_{cc} pozwala na zasilanie się układu z programatora lub programatora z układu. W wielu rozwiązaniach to połączenie może być pominięte.

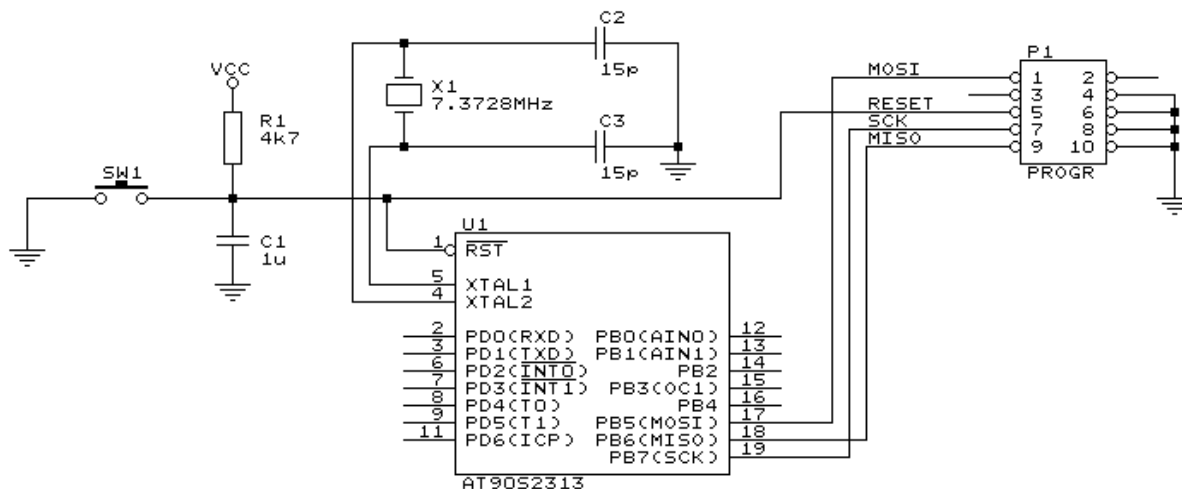
Przykładowe rozwiązanie przyłącza do programatora może być następujące (jak na rysunku 4 dla mikrokontrolera ATTINY12). Mikrokontroler ma wyprowadzenia sygnałów, które są niezbędne do przyłączenia programatora do programowania w układzie w trybie szeregowym.

Na identycznej zasadzie realizowane jest przyłączenie mikrokontrolera do



Rysunek 4. Przyłączenie złącza programującego do mikrokontrolera ATTINY12

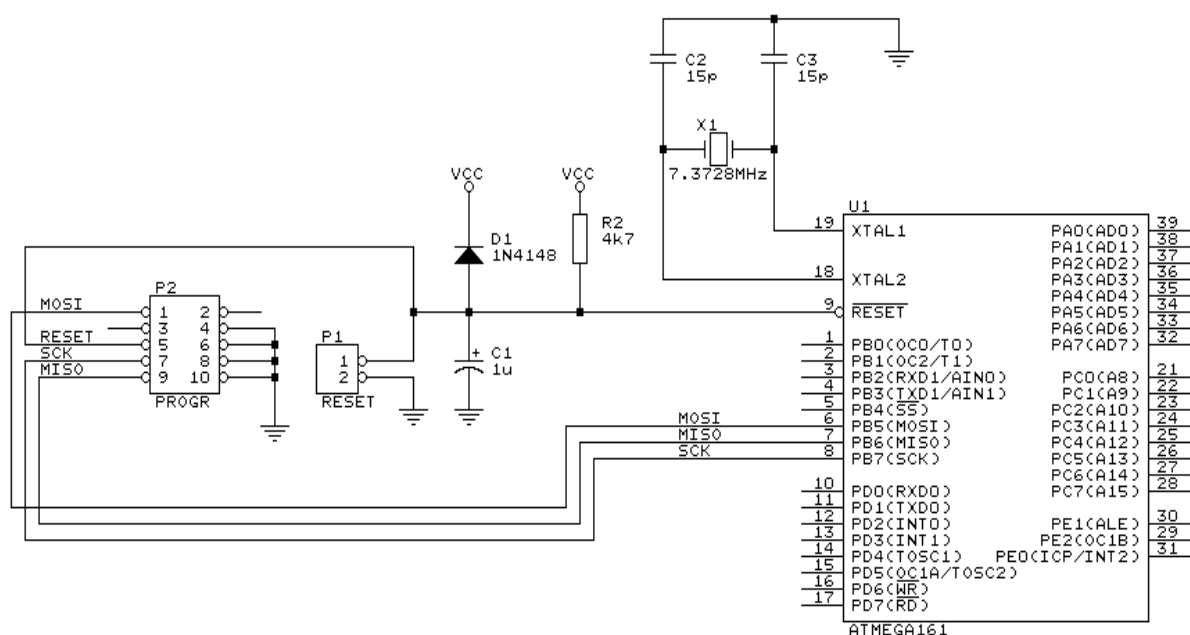
programowania w układzie w przypadku innych modeli mikrokontrolera. Przykład z użyciem bardziej złożonego mikrokontrolera (AT90S2313)



Rysunek 5. Przyłączenie złącza programującego do mikrokontrolera AT90S2313

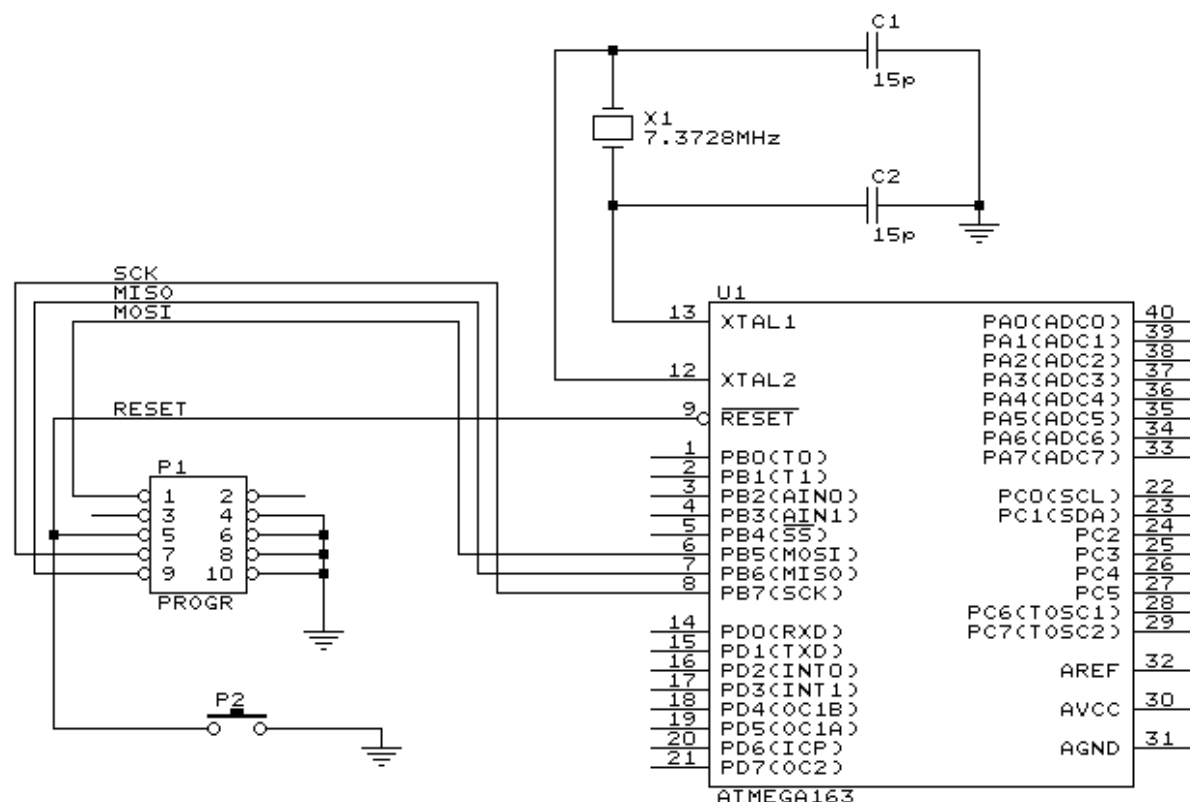
przedstawia rysunek 5.

Mikrokontrolery z rodziny ATMEGA też nie odbiegają w swoich rozwiązaniach od powyższego modelu. Przykład z użyciem mikrokontrolera ATMEGA161 pokazany jest na rysunku 6.



Rysunek 6. Przyłączenie złącza programującego do mikrokontrolera ATMEGA161

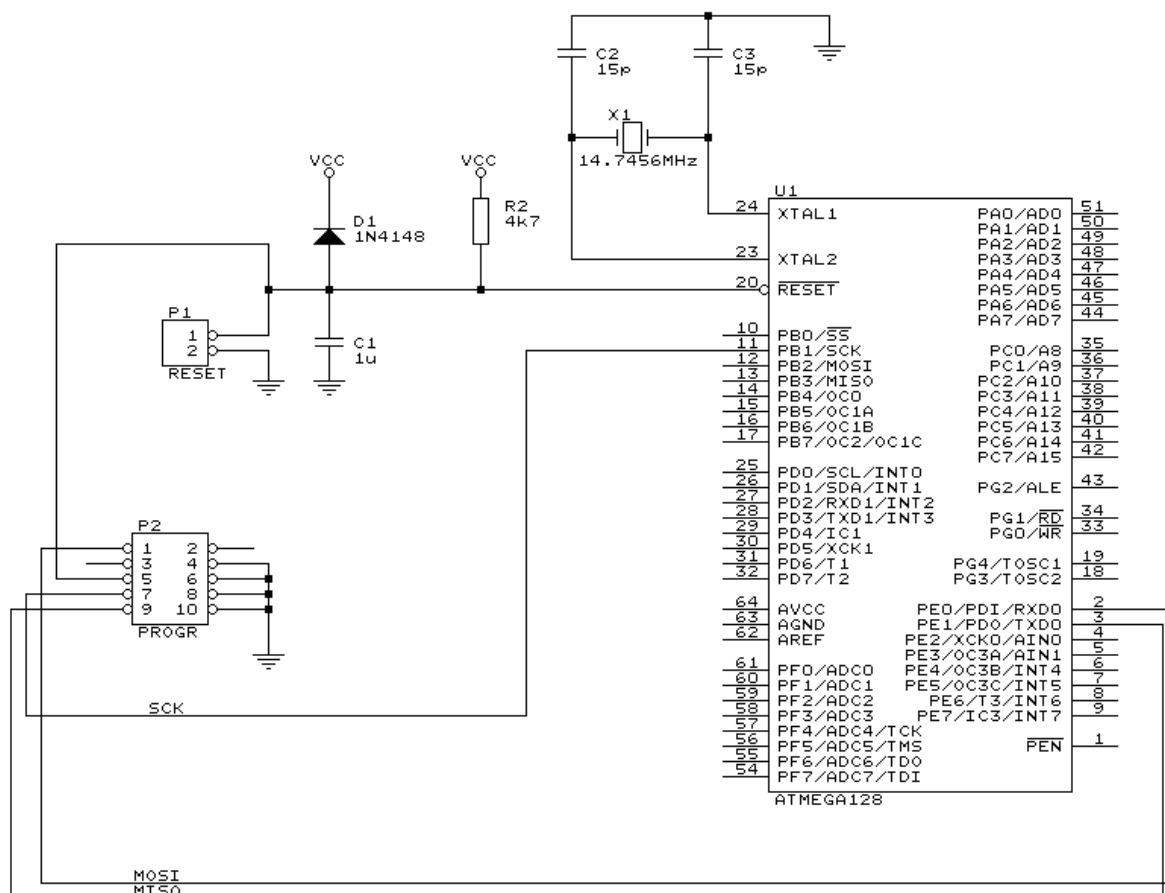
Przykład z użyciem mikrokontrolera ATMEGA163 jest pokazany na rysunku 7.



Rysunek 7. Przyłączenie złącza programującego do mikrokontrolera ATMEGA163

W większości przypadków sposób przyłączenia programatora pracującego w trybie programowania szeregowego nie odbiega od przedstawionej koncepcji, ale od tej reguły są wyjątki. Przykładem mikrokontrolera wymagającego innego

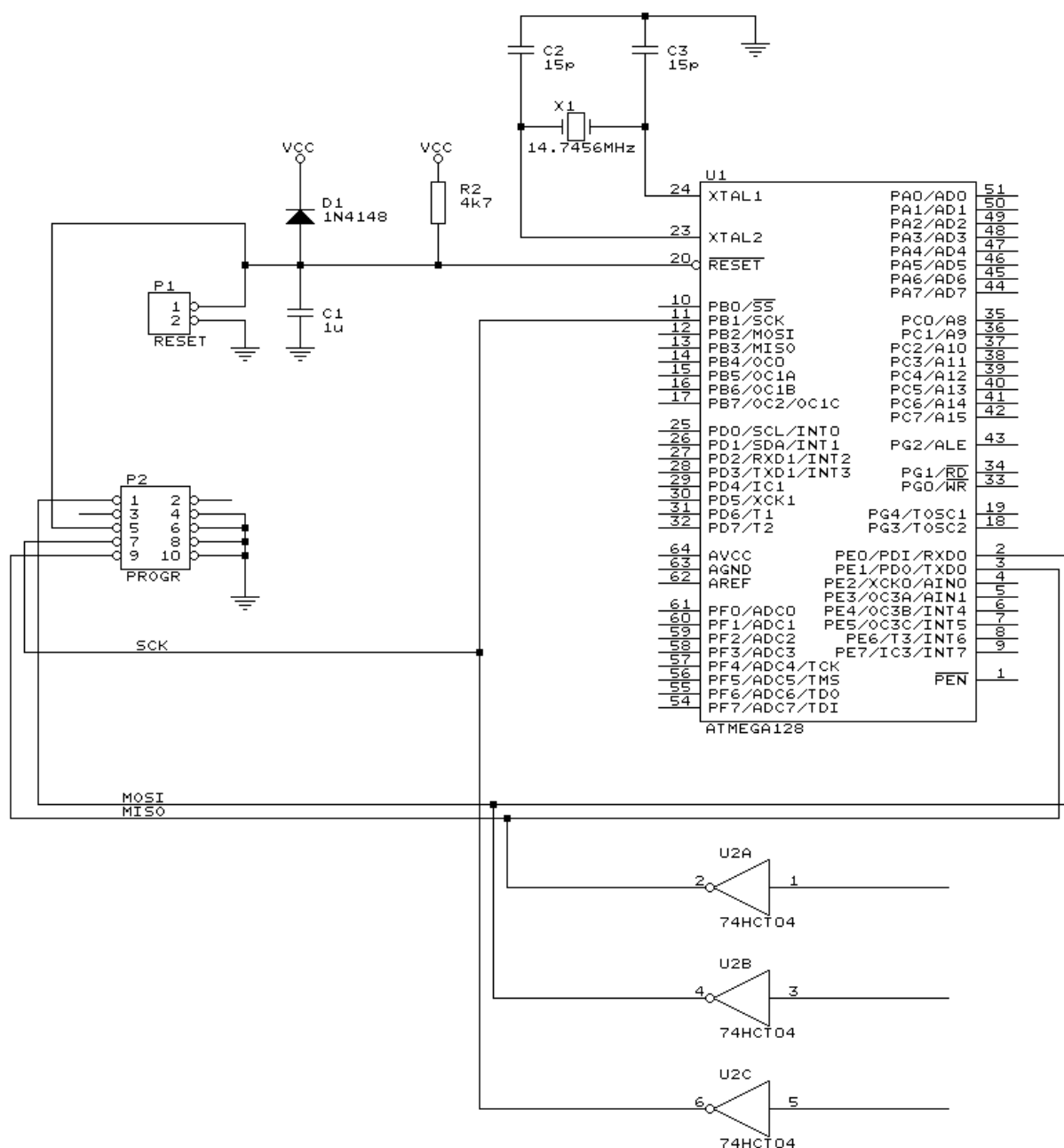
przyłączenia jest ATMEGA128. Tu sygnały MISO i MOSI są przyłączone w mikrokontrolerze w innym miejscu. Przedstawia to rysunek 8.



Rysunek 8. Przyłączenie złącza programującego do mikrokontrolera ATMEGA128

W każdym przypadku należy sprawdzić w oryginalnej dokumentacji producenta szczegóły związane z sposobem przyłączenia danego modelu mikrokontrolera do programatora szeregowego. Zaniechanie tej czynności może doprowadzić do późniejszych zmian w połączeniach na płycie drukowanej.

We wszystkich przedstawionych przykładach jest pominięty pewien problem, który może okazać się w konkretnym rozwiązaniu dość istotny. Użycie do szeregowego programowania sygnałów SCK, MOSI i MISO nie oznacza, że te sygnały nie mogą być używane w docelowym systemie do innych celów. Jeżeli z punktu widzenia mikrokontrolera te piny są wyjściami (mikrokontroler za pomocą tych pinów steruje jakimś blokiem cyfrowym) nie ma żadnego problemu. W przeciwnym wypadku (jeżeli któryś z tych pinów jest wejściem dla mikrokontrolera) mogą wystąpić problemy związane z programowaniem. Sytuacja taka jest przedstawiona na rysunku 9.



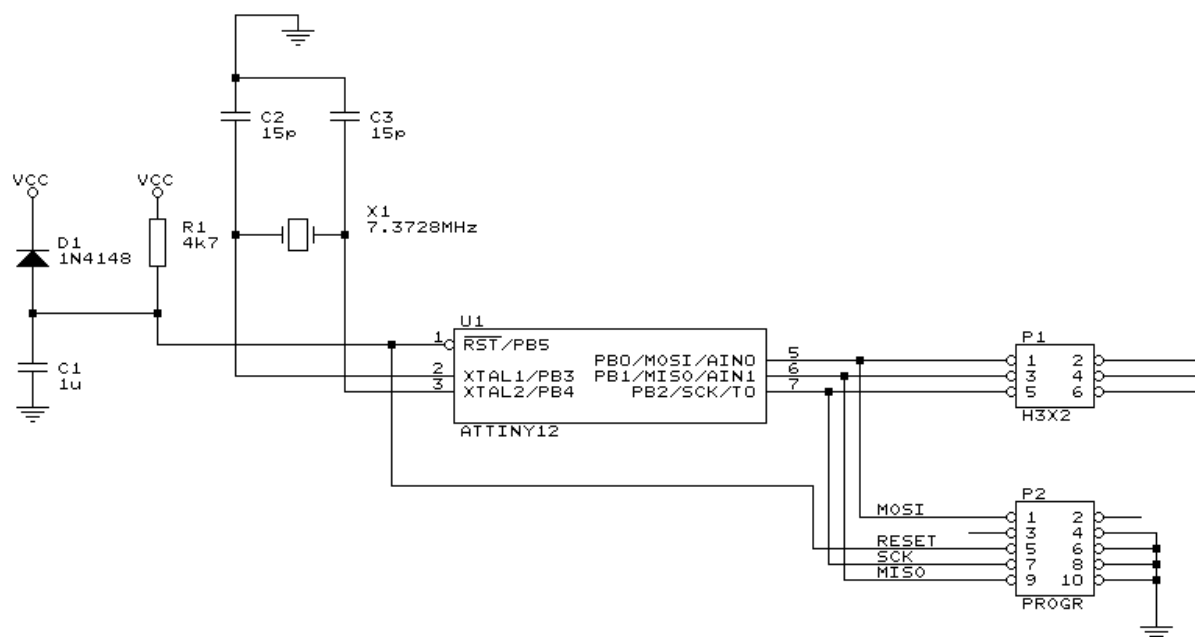
Rysunek 9. Przyczyny konfliktów logicznych

W trakcie programowania na każdym z trzech sygnałów używanych do programowania mikrokontrolera występuje logiczny konflikt, w obwodzie są dwa źródła sygnału:

- wyjście negatora (74HCT04) i sygnał SCK (programator jest źródłem sygnału),
- wyjście negatora (74HCT04) i sygnał MOSI (programator jest źródłem sygnału),
- wyjście negatora (74HCT04) i sygnał MISO (mikrokontroler jest źródłem sygnału).

Może to doprowadzić do niemożności zaprogramowania mikrokontrolera, gdyż wyjście negatorów może zakłócać przebieg sygnałów programujących.

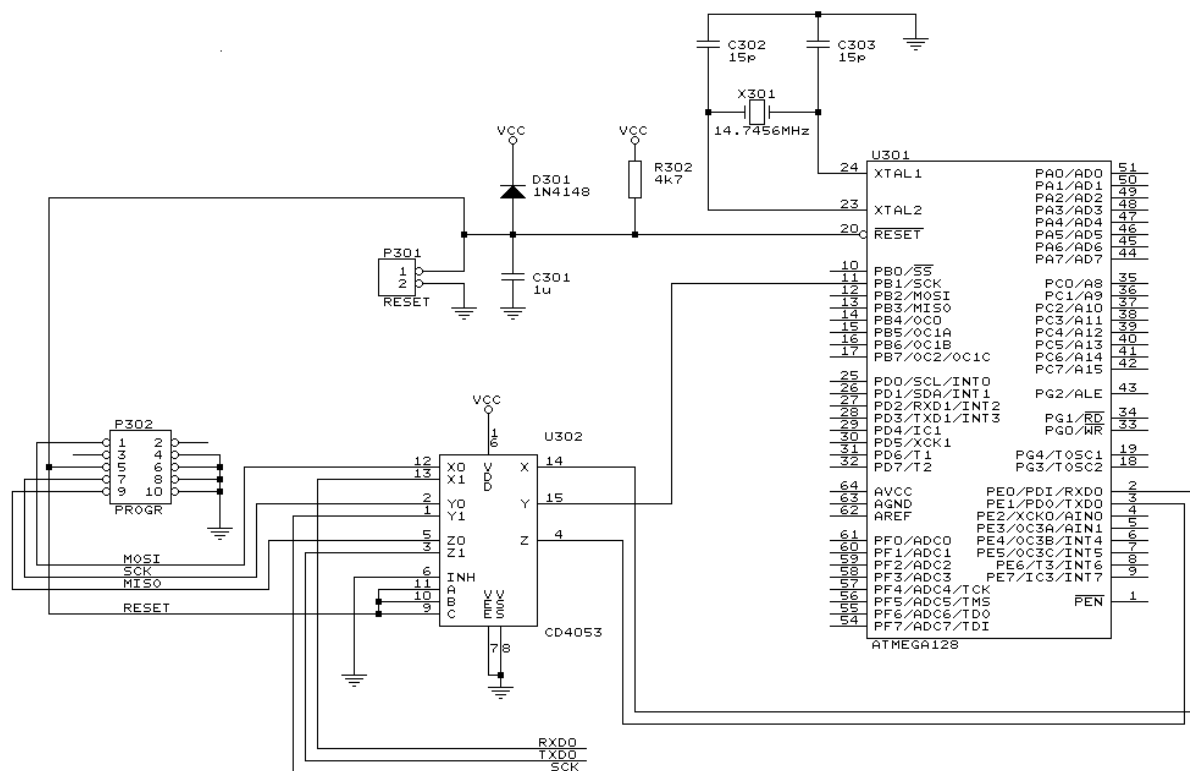
W niektórych rozwiązaniach układowych można problemu uniknąć, jeżeli te linie będą wyjściowymi (z punktu widzenia mikrokontrolera), czyli przyłączone do wejść w blokach cyfrowych, którymi steruje mikrokontroler. Niestety nie zawsze takie rozwiązanie jest możliwe. W takiej sytuacji można zaproponować dwa rozwiązania:



Rysunek 10. Eliminacja konfliktów logicznych

Pierwszy przypadek, pokazany na rysunku 10, jest fizycznym rozłączeniem poprzez użycie listy pinowej i zworek (jumperków). Na czas programowania zworki byłyby usuwane, co tworzyłoby przerwę w danym połączeniu.

Drugim rozwiązaniem jest elektroniczny przełącznik konfliktowych połączeń. Ilustruje to następujący rysunek 11.



Rysunek 11. Zautomatyzowana metoda eliminacji konfliktów

Układ CD4053 przełącza w odpowiedni sposób sygnały używane do programowania w trybie szeregowym. Sterowanie przełączaniem wejść/wyjść multipleksera jest oparte o sygnał RESET generowany przez programator w trakcie programowania. W przypadku, jeżeli mikrokontroler nie jest przyłączony do programatora, to rezystor występujący w obwodzie zerowania mikrokontrolera ustala sygnał na poziomie logicznej jedynki. W efekcie w trakcie programowania, układ CD4053 łączy mikrokontroler ze złączem programatora. W trakcie normalnej pracy, wymieniony multiplekser łączy mikrokontroler z blokiem cyfrowym.

Spis ilustracji

Rysunek 1. Złącze programujące	2
Rysunek 2. Schemat złącza programującego	3
Rysunek 3. Schemat złącza programującego - wariant uproszczony	3
Rysunek 4. Przyłączenie złącza programującego do mikrokontrolera ATTINY12	4
Rysunek 5. Przyłączenie złącza programującego do mikrokontrolera AT90S2313	4
Rysunek 6. Przyłączenie złącza programującego do mikrokontrolera ATMEGA161	5
Rysunek 7. Przyłączenie złącza programującego do mikrokontrolera ATMEGA163	5
Rysunek 8. Przyłączenie złącza programującego do mikrokontrolera ATMEGA128	6
Rysunek 9. Przyczyny konfliktów logicznych	7
Rysunek 10. Eliminacja konfliktów logicznych	8
Rysunek 11. Zautomatyzowana metoda eliminacji konfliktów	9