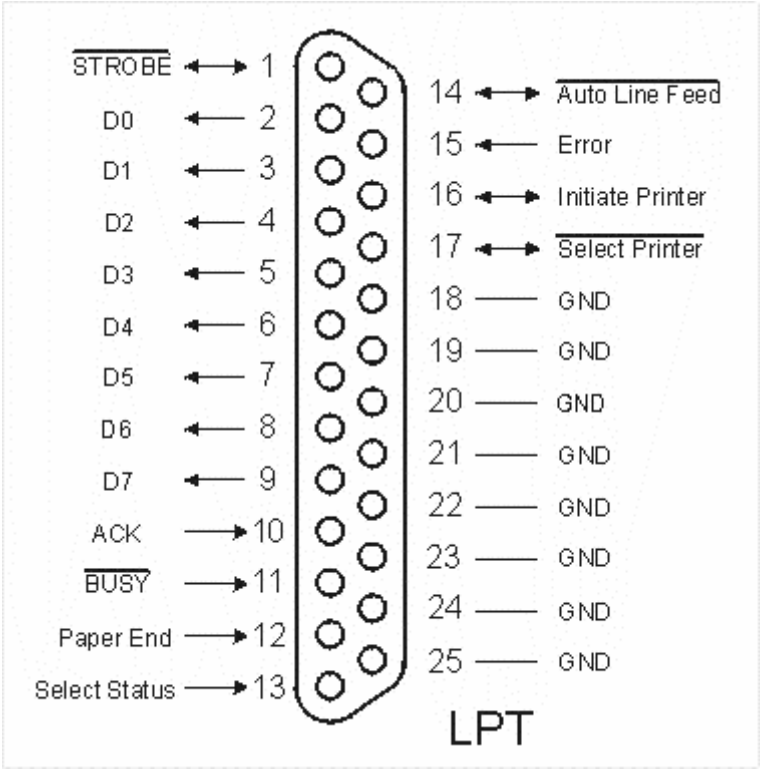
Sterowanie portem LPT

Zapewne większość użytkowników komputerów zna port równoległy potocznie zwany LPT a specjalistycznie **Interfejs IEEE 1284.** Swojego czasu służył on do współpracy z drukarkami , skanerami i tego typu urządzeniami biurowymi. Aktualnie zastąpiono go interfejsem USB.

Jednak dla elektroników hobbystów może on posłużyć jako wygodne narzędzie do sterowania swoimi układami elektronicznymi , np. do współpracy z mikrokontrolerami itd. Należy oczywiście pamiętać , że ma on ograniczoną wydajność prądową oraz napięciową dlatego najlepiej robić to za pomocą układu sprzęgającego np. opartego o transoptory. W Internecie można znaleźć wiele takich rozwiązań służących zabezpieczeniu i odizolowaniu portu LPT od układu elektronicznego.

Celem tego artykułu jest przedstawienie sposobu na wysterowanie odpowiednich pinów danych portu LPT przy pomocy języka programowania C++ wraz z użyciem asemblera oraz programiku UserPort (dotyczy WinXP i rodziny NT). Magistrala tego interfejsu składa się z: 8 linii danych, 4 linii sterujących i 5 linii statusu. My zajmiemy się tylko 8 liniami danych.



Rys 1. widok portu LPT

Linie danych dotyczą pinów od 2 do 9. Podpisanych odpowiednio D0-D7. Tymi nazwami będziemy dalej operować. Sygnały pojawiające się na pinach D0-D7 odpowiadają poziomom TTL.

Najczęściej adres bazowy portu LPT to 378_{hex}. Może się jednak zdarzyć, że będzie on pod innym adresem! Domyślną wartość można na przykład zmienić w BIOSie. Poniższa tabelka przedstawia rozpiskę standardowych adresów wyrażonych w typie HEX. My będziemy korzystać ze standardowego adresu bazowego wyrażonego w HEX.

Adres (hex)	**Znaczenie**	**Standardowy adres bazowy (hex)**	**Standardowy adres bazowy - z kartą Hercules**
0040:0008	adres bazowy LPT1	**378**	3BC
0040:000A	adres bazowy LPT2	**278**	378
0040:000C	adres bazowy LPT3	**3BC**	278
0040:000E	adres bazowy LPT4	**2BC**	2BC

Jak łatwo się domyśleć na port wysyłany jest bajt danych czyli ustawiane jest 8 bitów. Te osiem bitów to właśnie symbolicznie ustawienie logicznych 0 i 1 na pinach od D0 do D7.

Przykładowy bajt danych : 00001111 ustawi piny D0-D3 w stan wysoki czyli ok. 5V , natomiast piny D4-D7 będą miały poziom napięcia ok. 0 V. Jako, że w programie posługiwać się będziemy zapisem szesnastkowym to wyżej przedstawiony przykład można opisać tak, że na port została wysłana liczba 0x0F.

Zakładam , że czytelnik zna podstawy programowania. Zamieszczam poniżej funkcje odpowiedzialną za ustawienie odpowiedniej wartości portu LPT.

```
void outport(int,int)
{
    asm(".intel_syntax noprefix");          //assembler - INTEL
    asm("mov dx, [ebp + 8]");               //do rej DX podajemy adres portu
    asm("mov al, [ebp + 12]");              // do rej AL podajemy bajt danych
    asm("out dx, al");                      // wysyłanie
    asm(".att_syntax prefix");              //żeby nie było błędu kompilacji.
};
```

Funkcja wystawiająca odpowiedni bajt danych na port LPT nie zwracająca nic.

Przykład użycia tej funkcji to :

outport(0x378, 0x0F);

Tym sposobem wystawimy bajt danych 00001111 jak opisany powyżej .

Kod całego programu jest dostępny do ściągnięcia na stronie
http://www.stepel.elektroda.eu.

W paczce na stronie dostępny jest skompilowany program mojego autorstwa do sterowania portem, UserPort i plik CZYTAJ!!! który opisuje jak użyć UserPortu (jednorazowe użycie załatwia sprawę odblokowania dostępu do portu LPT). Dotyczy to systemu WinXP i innych z rodziny NT. Po wykonaniu czynności można cieszyć się dowolnym sterowaniem danymi portu LPT.

autor: Wojtek Stępień