Układy pośredniczące -> układy wejścia/wyjścia -> umieszczane na kartach rozszerzeniowych lub na płycie głównej kompa

procesor nie steruje urządzeiami bezpośttednio tylko za pomocą ukladow wejsica/wyjscia

Układy wejścia/wyjścia umożliwiają testowanie stanu (gotowości) urządzenia, wysyłanie poleceń do urządzenia oraz wysyłanie i przyjmowanie danych.

komunikacja odbywa sie poprzez zapis/odczyt rejestrow umeiszczonych w ukladach wejscia/wyjscia

DMA -> direct memory access -> metoda przesylania danych z pamięci operacyjnej do urządzenia (lub odwrotnie) z pominięciem procesora

4 rejestry w ukladach wejscia/wyjscia:

1) rejestr stanu -> czy urządzenie jest zajete/czy dane sa gotowe do odczytania/ czy wystapil błąd
2) rejestr sterujacy -> przyjmuje polecenia które urzadzenie ma wykonać
3)rejestr danych wysyłanych do urzadzenia
4) rejestr danych odbieranych z urzadzenia

2 metody dostepu do rejestrow ukladu wejscia/wyjscia

1) rejestry jako zwykłe komorki pamieci -> wspoladresowalne układy wejścia/wyjscia
2) rejestry dostepne w odrebnej przestrzeni adresowej -> przestrzen adresowa wejscia/wyjscia / przestrzen adresowa portow / izolowane wejscie/wyjscie

sterowanie pracą urządzeń zewnętrznych komputera może być wykonywanie jedynie przez system operacyjny

W trybie tekstowym sterownika graficznego (karty graficznej) znaki wyświetlane na ekranie stanowią odwzorowanie zawartości obszaru pamięci od adresu fizycznego B8000H – pamięć ta należy do przestrzeni adresowej procesora, ale zainstalowana jest na karcie sterownika.

tryb tekstowy jest ladowany przez BIOS przed zaladowaniem glownego systemu operacyjnego

Każdy znak wyświetlany na ekranie jest opisywany przez dwa bajty w pamięci ekranu: bajt o adresie parzystym zawiera kod ASCII znaku, natomiast

następny bajt zawiera opis sposobu wyświetlania,
nazywany atrybutem znaku.

mov byte ptr es:[4], 'E' -> zapis do pamieci ekranu

adres fizyczny -> es*16+zaw.pola adresowego (tutaj 4)

dość prosty tryb graficzny oznaczony nr 13h -> 320x200 pikseli -> kazdy piksel moze byc wyswietlany w jednym z 256 kolorow (64 000 bajtów)

10 -> kolor jasnozielony

w trybie 13H pamiec ekranu jest umieszczona od addresu fizycznego A0000h

INT 10h / AH=0 / AL = 3 mozna powrocic do trybu tekstowego

```
mov ah, 0 -> funkcja 0 ustawia tryb sterownika
mov al, 13h -> nr trybu
int 10h -> wywolanie funkcji systemu BIOS
mov ax, 0A000h -> adres pamieci ekranu
mov es, ax
mov cx, 200 -> liczba linii na ekranie
mov bx, 160 -> adres poczatkowy
ptl_lin:
 mov byte ptr es:[bx], 10 -> kolor jasnozielony
add bx, 320
loop ptl_lin
```

ten kod wyswietli na srodku ekranu w dosboxie pionową linie

w architekturze x86 stosuje sie rozkazy IN i OUT oraz ich rozszerzenia do zapisu i odczyutu danych w przestrzeni adresowej portow

```
in al, 60h -> przesłanie zawartosci portu o nr 60h do rejestru al
out 64h, al -> przeslanie zawartosic rejestru al do portu 64h
```

zmiana palety -> wpisanie kodu koloru do portu 3C8h a nastepnie przeslanie składowych RGB <0;63> do portu 3C9h

```
mov dx, 3C8H
mov al, 10 ; kod koloru
out dx, al
mov dx, 3C9H
mov al, 63 ; składowa czerwona (R)
out dx, al
mov al, 63 ; składowa zielona (G)
out dx, al
```

```
mov al, 0 ; składowa niebieska (B)
out dx, al
```

teraz pod kodem 10 zamiast jasnozielonego będzie zółty
in/out nr portu mozna podac natychmiastowo jesli miesci sie on na 8 bitach, w przeciwnym wypadku tak jak wyzej przez DX

kazdy przycisk klawiatury ma ustalony 8 bitowy kod nazywany kodem pozycji lub nr klawisza (scan code)

Po naciśnięciu lub zwolnieniu dowolnego klawisza mikrokontroler (mikroprocesor) klawiatury formuje kod naciśnięcia (ang. make code) lub kod zwolnienia (ang. break code) klawisza – kod ten zostaje przesłany szeregowo do układów płyty głównej komputera

Kod zwolnienia klawisza zawiera kod naciśnięcia poprzedzony bajtem F0H

Pojawienie się zera na i-tej kolumnie po podaniu zera na j-ty wiersz oznacza, że został naciśnięty klawisz leżący na skrzyżowaniu i-tej kolumny j-tego wiersza.

Naciśnięcie klawisza powoduje połączenie linii wiersza z linią kolumny (8 linii kolumn doprowadzono do multipleksera).
 W celu zidentyfikowania naciśniętego klawisza procesor sterujący generuje na swoich wyjściach liczby adresujące dekoder i multiplekser.
 Część adresu podawana na dekoder uaktywnia jedną z jego linii (wprowadzając ją w stan zero).
 Przy ustalonej wartości na dekoderze zmienia się wartość liczb podawanych na multiplekser (są to bowiem najmłodsze bity).

jesli zaden klaiwsz nie jest nacisniety to na wyjsicu bedzie stan wysoki

W przypadku wciśniętego klawisza na wyjściu multipleksera pojawi się stan zero, ale tylko dla kombinacji powodującej połączenie wyjścia multipleksera z linią, do której dołączony jest wciśnięty klawisz

odtworzony kod pozycji w porcie 60h

Zarówno mikrokontroler na płycie głównej jak też

mikrokontroler klawiatury mogą być programowane za pomocą rozkazów wysyłanych przez główny procesor poprzez porty 60H i 64H.

```
; zablokowanie przerwań z klawiatury
mov al, 2
out 21H, al
czekaj:
in al, 64H ; odczyt rejestru stanu
; klawiat.
; sprawdzenie czy kod pozycji dostępny jest
; w buforze wyjściowym
test al, 1
jz czekaj ; oczekiwanie w pętli
; odczytywanie kodu pozycji naciśniętego
klawisza
in al, 60H
```

odczytywanie nr nacisnietego klawisza

Zlecenie by urządzenie zewnętrzne dołączone do komputera wykonało pewną operację wymaga podjęcia następujących działań:
- sprawdzenie stanu urządzenia;
- wysłanie odpowiednich poleceń do urządzenia, o ile znajduje się ono w stanie gotowości;
- przesłanie (lub odczytanie) danych;
- oczekiwanie na zakończenie operacji.

w przypadku wielu urzadzen przesylanie danych moze nastapic tylko wtedy gdy:
1)urzadzenie jest juz gotowe do przyjecia danej
2)dana jest juz przygorowana do udostepnienia komputerowi

jesli odp jest pozytywna nastepuje przeslanie danej, jesli nie operacja sprawdzania jest powtarzana wielokrotnie -> metoda aktywnego oczekiwania/odpytywania -> nieefektywna, jałowo pochłania czas pracy procesora

metoda przerwaniowa -> procesor po otrzymaniu sygnału przerwania przerywa wykonywanie obecnego programu i rozpoczyna wykonywanie innego

Zalety obsługi urządzeń przy użyciu przerwań są szczególnie dobrze widoczne w systemach wielozadaniowych (wieloprocesowych), w których w trakcie oczekiwania na gotowość urządzenia procesor może wykonywać inne czynności (inne programy).

 Przy szybkiej transmisji danych i dużej liczbie zgłaszanych przerwań może wystąpić znaczne obciążenie procesora spowodowane koniecznością

przełączania kontekstu (program użytkowy / system operacyjny) – w takich przypadkach lepszym rozwiązaniem może być metoda aktywnego oczekiwania.

metoda przerwaniowa jest odpowiednia dla niezbyt szybkich urzadzen -> do kilu tysiecy przerwan na sekunde

aby po przerwaniu procesor mógł wrócic do wykonywania poprzedniego programu zapisywany na stosie jest tzw ślad, zawierający adres rozkazu, który miał być wykonany jako nastepny, ale nie byl, gdyż nastąpiło przerwanie

Sygnał przerwania, poprzez jedną z linii IRQ 0, IRQ 1, ... kierowany jest do układu APIC, który wspomaga procesor w obsłudze przerwań.

Na podstawie numeru linii IRQ wyznaczany jest odpowiedni wiersz w tablicy adresowej nazywanej tablicą deskryptorów przerwań.

w linuxie IRQ 1 -> sygnały przerwań z klawiatury -> wiersz nr 33 w tablicy deskryptorow przerwań

każdy deskryptor zajmuje 8 bajtów

Adres zawarty w deskryptorze wskazuje położenie podprogramu obsługi przerwania – adres ten wpisywany jest do rejestru EIP.

Ponadto zerowany jest znacznik IF, co blokuje przyjmowanie innych przerwań.

Na końcu podprogramu obsługi przerwania umieszczony jest rozkaz IRET, który pobiera ślad wcześniej zapamiętany na stosie i wpisuje go do rejestru EIP – w rezultacie następuje wznowienie wykonywania przerwanego programu.

Zazwyczaj sygnały przerwań dochodzące z urządzeń szybkich mają wyższy priorytet.

W architekturze x86 przyjęcie przerwania powoduje wyzerowanie znacznika IF, co blokuje przyjmowanie dalszych przerwań.

Podprogram obsługi przerwań może jednak ustawić znacznik IF w stan 1 (za pomocą rozkazu STI), co otwiera możliwość przerwania podprogramu obsługi przez przerwanie o wyższym priorytecie

te przerwania mogą byc blokowane poprzez wyzerowanie znacznika IF -> przerwania maskowalne

przerwania niemaskowalne -> nie mogą byc blokowane(NMI - Non maskable interrupt)

stosuje sie je do sygnalizacji zdarzen wymagajacych natychmiastowej obsługi niezależnie od stanu systemu

Sterowniki DMA (ang. Direct Memory Access)
umożliwiają bezpośrednie przesyłanie danych z
urządzenia do pamięci głównej (operacyjnej) lub z
pamięci do urządzenia.
 Przesyłanie odbywa się bez udziału procesora – trzeba
jedynie odpowiednio zainicjalizować układ DMA
(strzałka ① na rysunku).
 Po przesłaniu wszystkich bajtów sterownik DMA
generuje przerwanie sprzętowe sygnalizujące koniec
przesyłania (strzałka ② na rysunku).

PIO -> Programmed Input-Output -> zwykla transmisja za pośrednictwem procesora

Inicjalizacja układu DMA wymaga:
• wpisania adresu początkowego przesyłanego
obszaru pamięci do rejestru adresu sterownika
DMA,
• wpisania długości bloku danych do licznika
przesyłanych danych.

procedura obsługi przerwania z DMA wywoływana jest po przesłaniu bloku danych, a nie przy kazdej transmisji

układy DMA -> do/z szybkich urządzeń

wyjątki:
1)niepowodzenia (faults) ->aktualnie wykonywany rozkaz spowodował błąd; można ponowić wykonanie tego rozkzu na podstawie śladu na stosie
2)pułapki(traps) -> używane m.in w debuggerach; gdy procesor powraca do wykonywania przerwanego kodu wykonuje rozkaz nastepny po tym ktory spowodowal wyjatek
3)błedy nienaprawialne (aborts) -> nie można zlokalizowac blednego rozkazu ani kontynuwac programu

obsługa wyjatku -> zapamietanie sladu na stosie -> wyzerowanie IF -> uruchomienie podprogramu obslugi wyjatku

wyjątek nr 0 błąd dzielenia generowany, jeśli w
trakcie wykonywania rozkazu DIV lub IDIV wystąpił
nadmiar lub dzielnik był równy zero;
• wyjątek nr 6 niedozwolony kod – generowany przy

próbie wykonania rozkazu o kodzie nierozpoznanym przez procesor;
• wyjątek nr 7 urządzenie niedostępne – generowany przy próbie wykonania rozkazu odnoszącego się do niedostępnego urządzenia; przykładowo, w pewnych sytuacjach koprocesor arytmetyczny może być tymczasowo niedostępny (bit TS=1 w rejestrze CR0) – próba wykonania rozkazu koprocesora powoduje wygenerowanie tego wyjątku;

wyjątek nr 11 – brak segmentu – generowany w przypadku załadowania selektora do jednego z rejestrów segmentowych, wskazującego deskryptor, w którym bit P = 0;
• wyjątek nr 13 – błąd ochrony (ang. general protection) – generowany w przypadku próby naruszenia niedostępnych zasobów; wyjątek ten jest używany w sytuacji jeśli próba naruszenia nie może być zaklasyfikowana bardziej precyzyjnie;
• wyjątek nr 14 – błąd stronicowania – generowany, jeśli odwołanie dotyczy strony aktualnie nieobecnej w pamięci operacyjnej.