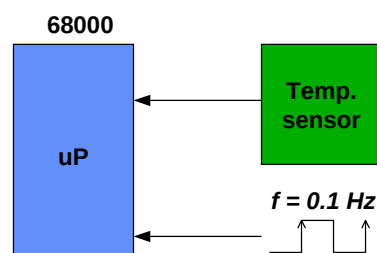


F8: Undantagshantering

- Undantagshantering i 68000
 - Vad är ett undantag?
 - Typer av undantag
 - Att skriva undantagsrutiner

Undantagshantering, vad och varför ?

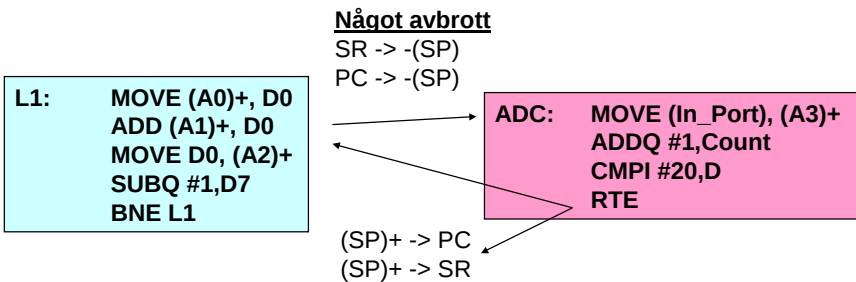
- Exempel:
 - Ett system ska mäta temperatur var 10:e sekund
 - Hur ska processorn veta att det gått 10 sekunder????
- Lösning:
 - Skicka en extern signal till processorn var 10:e sekund
 - Signalen avbryter pågående exekvering, och startar ett program som läser och lagrar temperatur
- Detta kallas interrupt
 - externa signaler som avbryter ordinarie exekvering så att processorn startat ett annat program (interruptrutin)



Interrupt är en typ av undantag

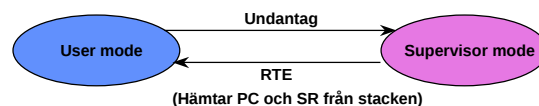
Undantag (Exception)

- En händelse (yttre eller inre händelse) som avbryter den normala exekveringen och börjar exekvera en speciell undantagsrutin
 - Kan liknas med ett subrutinanrop fast utan anrop



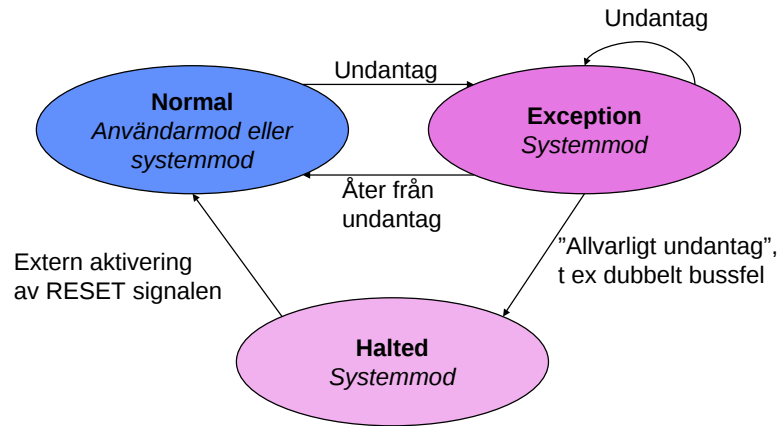
När ett undantag inträffar så:

- Lagras programräknaren (PC) på stacken
 - Utom vid Reset
- Lagras Status registret på stacken (SR)
 - Utom vid Reset



- User mode, applikationsprogram
- Supervisor mode, operativsystemet exekveras i systemmod

System states



Två orsaker till undantag

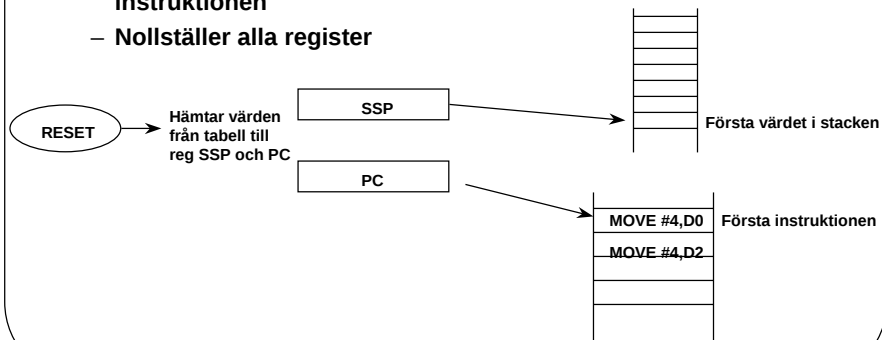
- **Yttre händelse**
 - Reset signalen aktiveras
 - Busserror
 - Interruptsignalerna aktiveras
- **Inre händelse**
 - Division med noll
 - TRAP
 - TRAPV
 - Felaktig instruktion
 - Privilegierad instruktion
 - Adresserror
 - Trace

Gruppering av undantag

- 0
 - Reset Pågående instruktion avbryts och undantagsrutinen påbörjas inom 2 klockpulser
 - Bus error
 - Addresserror
- 1
 - Trace Pågående instruktion avslutas.
 - Interrupt Därefter startas undantagshantering
 - Privilege violation
 - Illegal instruction
- 2
 - TRAP, TRAPV Vanlig instruktionsexekvering som medför
 - CHK, Div. med 0 att undantagshantering startas.

Yttre

- Reset (HALT* och RESET* aktiveras samtidigt)
 - Initierar Stackpekaren med värden från tabell
 - Initierar program räknaren till ett värde som pekar på första instruktionen
 - Nollställer alla register



Yttre forts.

- **Buserror**

- BERR* signalen aktiveras
 - Dvs. något fel uppstod i buscykeln
- Startar en undantagsrutin

- **Interrupt**

- Någon av de sju kombinationerna på interruptsignalerna (IPL0*-IPL2*) aktiveras
 - 111 Inget interrupt
 - 110 Interrupt 1 - Lägst prioritet
 - 101 Interrupt 2
 - ...
 - 000 Interrupt 7 - Högst prioritet
- En prioritetsavkodare används för att driva signalerna IPL0*-IPL2*
- Startar en undantagsrutin

Inre, t ex

- **Division med noll**

- Om ett försök till division med noll inträffar i programmet aktiveras detta undantag
 - MOVE #0,D0
 - DIVU D0,D3
- En undantagsrutin för division med noll startar

- **TRAP**

- I ett program utförs instruktionen TRAP #x, där x är 0-15
 - Undantagsrutin med vektornummer 32+x startas
 - T ex TRAP #4, så startar undantags rutinen med vektornummer 36
 - Används vanligtvis för operativsystemanrop

Adressering av undantagsrutiner

- Utifrån vilket undantag som inträffat väljer CPU vilken undantagsrutin som ska exekveras genom att titta i en tabell
 - Tabellen är initialt ALLTID placerad på adress \$000-\$3FF
 - Ex. 68340 kan relokerat tabellen till valfri position med VBR
 - Inget utom denna tabell får finnas på dessa adresser

00 0000	SSP vid RESET
00 0004	PC vid RESET
00 0008	Adress till Busserror rutin
00 000C	Adress till Adresserror rutin
	↓ osv
00 03FF	

Adressering...

- Programmeraren bestämmer vilken adress som ska lagras i tabellen
- T.ex.. om följande programkod var den som skulle exekveras när RESET aktiveras så ska:
 - Adress 00 0004 innehålla \$0001 1000
 - ORG \$11000
 - START MOVE #3,D0
 - BTST.L D0,D3
 - osv...
 - När RESET aktiveras kommer processorn att ladda PC med adress \$11000 och börja exekvera MOVE instruktionen

Vektortabell

Nr.	Adress	Användning
0	000	Reset: initiering för SSP
1	004	Reset: initiering för PC
2	008	Bussfel
3	00C	Adressfel
4	010	Illegal instruktion
5	014	Division med noll
6	018	CHK instruktion
7	01C	TRAPV-instruktion
8	020	Privilege violation
9	024	Trace
10	028	Emulering
11	02C	Emulering
12-23	030-05F	Reserverade
24	060	Falskt avbrott
25-31	064-07F	Nivå 1-7, autovektor
32-47	080-0BF	TRAP 0-F
48-63	0C0-0FF	Reserverade
64-255	100-255	Användarens avbrottsvektor

Vektoriserat avbrott 1:4

Avbrottsnivå → IPL0-IPL2

111 → FC0-FC2

Prioritetsnivå → A1-A3

0 → AS*, DS*

0 → DTACK*

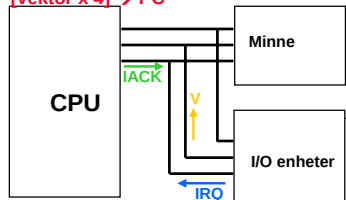
Vektor → D7-D0

1 → DS*, 1 → AS*

I/O-enhet → 1 → DTACK*

PC, SR → -(SP)

[vektor x 4] → PC

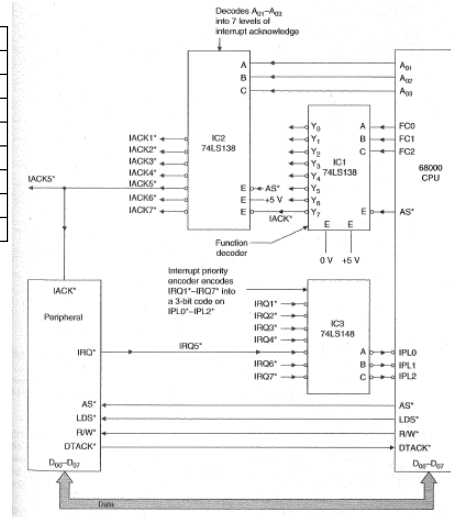


(SP)+ → PC, SR

- En I/O-enhet signalerar till processorn att den behöver hjälp genom signalen **Interrupt Request**.
- Om avbrottsnivå > i0-i2 i SR eller = 7, så avslutas aktuell instruktion och processorn signalerar samtidigt till I/O-enheten att den accepterar avbrottet – **Interrupt Acknowledge**.
- I/O-enheten identifierar sig nu genom att skicka en **Vektor** på databussen.
- Processorn **kvitterar** att den fått vektorn och startar sedan utpekad servicerutin
- Servicerutinen avslutas med **RTE**.

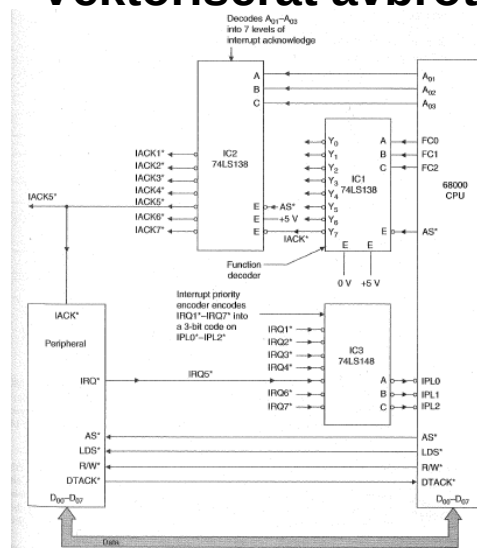
Vektoriserat avbrott 2:4

FC2	FC1	FC0	Status
0	0	0	Används ej
0	0	1	Addr. av data i användarmod
0	1	0	Addr. av program i användarmod
0	1	1	Används ej
1	0	0	Används ej
1	0	1	Addr. av data i systemmod
1	1	0	Addr. av program i systemmod
1	1	1	Avbrottskvittens



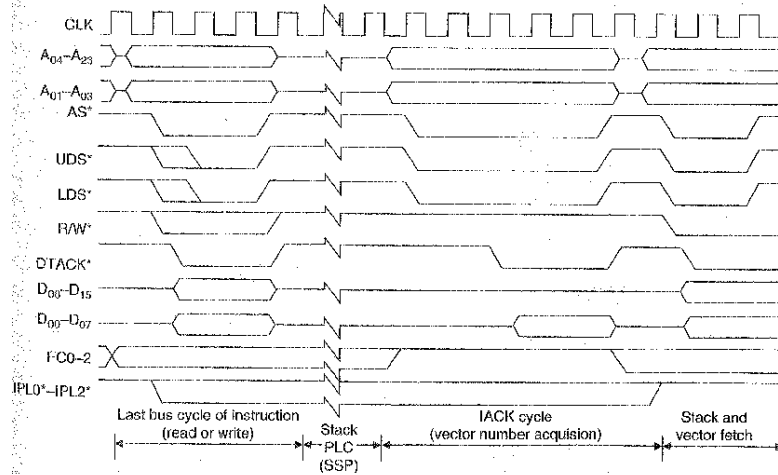
15

Vektoriserat avbrott 3:4



16

Vektoriserat avbrott 4:4



Auto-vektor-avbrott 1:2

Avbrottsnivå → IPL0-IPL2

111 → FC0-FC2

Prioritetsnivå → A1-A3

0 → AS*, DS*

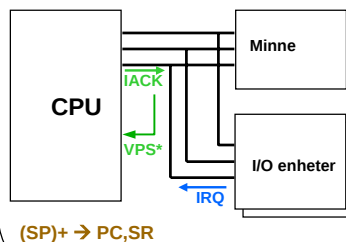
Extern logik → 0 → VPS*

1 → DS*, 1 → AS*

1 → VPS*

PC, SR → -(SP)

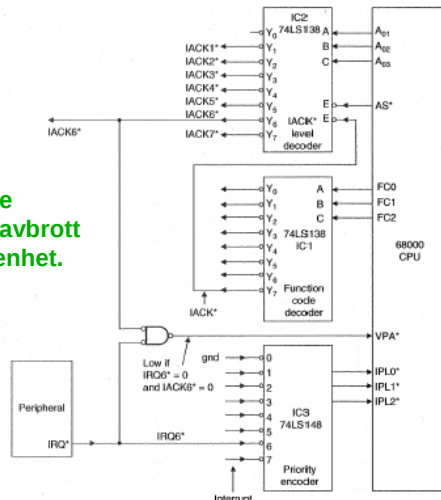
[autovektor x 4] → PC



- En I/O-enhet signalerar till processorn att den behöver hjälp genom signalen **Interrupt Request**.
- Om avbrottsnivå > i0-i2 i SR eller = 7, så avslutas aktuell instruktion och processorn signalerar samtidigt till I/O-enheten att den accepterar avbrottet – **Interrupt Acknowledge** vilket medför att extern logik omedelbart kvitterar med att aktivera VPS*.
- Processorn **kvitterar** att den registrerat autovektoravbrott och startar sedan utpekad servicerutin.
- Servicrutinen avslutas med **RTE**.

Auto-vektor-avbrott 2:2

Fördel: Enklare anslutning av avbrott för en periferienhet.



Undantagsrutiner

- En undantagsrutin skrivs på samma sätt som en subrutin
- I en undantagsrutin är det **VIKTIGT** att inte påverka registrens innehåll
- Ex. en undantagsrutin för division med noll
 - Adress \$00 0014 ska innehålla adressen \$14000

```

ORG      $14000
Div_noll MOVEM.L A0/D0-D1,-(A7)
          LEA     TEXT,A0    * A0 ska peka på texten vi vill skriva
          MOVE.W  #30,D0     * Skriv max 30 tkn
          CLR.W   D1         * Skriv till LCD skärm
          TRAP    #7
          MOVEM.L (A7)+,A0/D0-D1
          RTE          * Hoppa till baka till huvud programmet
TEXT     DC.B    'Du får inte dividera med NOLL',0
    
```