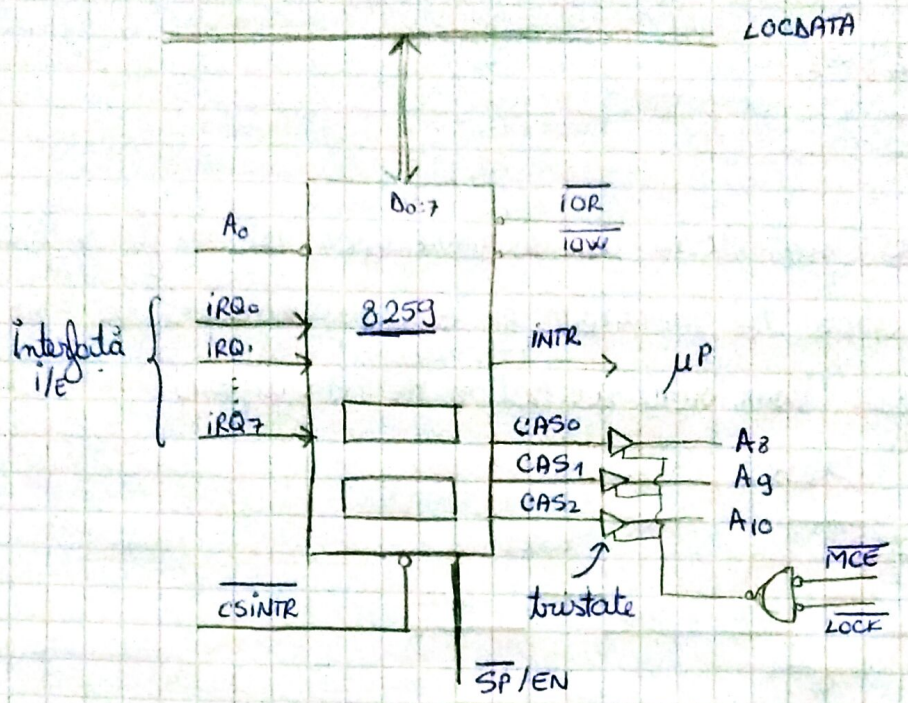
09.05.2018

CURS 11

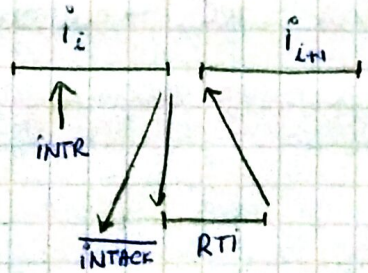


Acest chip funcționează pe 8 niveluri de întreruperi

$IRQ_0, \dots IRQ_7$ → către interfețele de intrare/ieșire

INTR → către microprocesor

Dacă se primește o întrerupere în mijlocul unei instrucțiuni, μP o să răspundă de abia la finalul instrucțiunii curente cu interupt Acknowledge (dialog asincron).

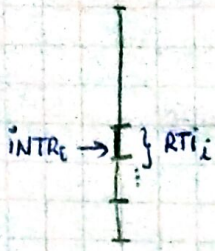


← Doar dacă avem Enable interrupt, altfel nu lucăm în considerare întreruperea.

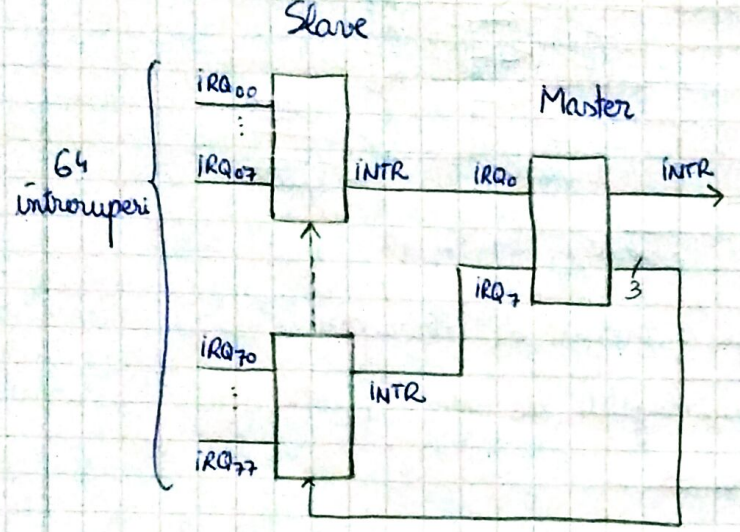
La primirea unei întreruperi, μP trebuie să știe de la ce nivel a primit întreruperea ⇒ Si îi furnizează un vector de întreruperi.

În memoria RAM există anumite celule numite celule capcană. Acestea identifică adresa unei rutine de tratare a unei întreruperi

prin furnizarea întreruperii ca intrare.



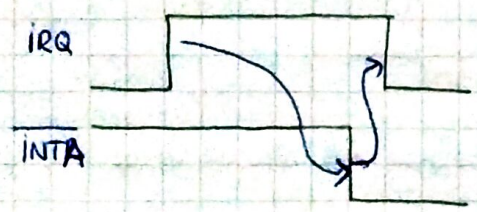
Există posibilitatea ca 8 întreruperi să nu fie suficiente, așă că vom încerca să furnizăm o organizare ierarhică: în spatele fiecărei întreruperi vom pune un sistem de întreruperi.



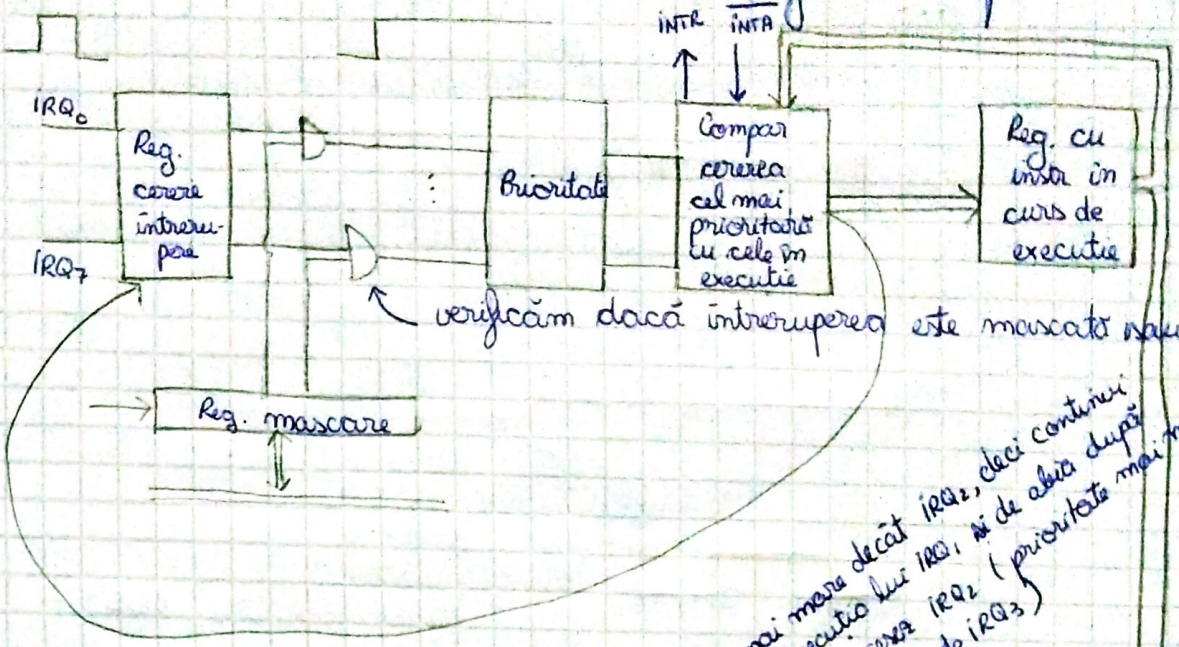
$\overline{SP}$/EN este folosit pentru a ști dacă e slave sau master.

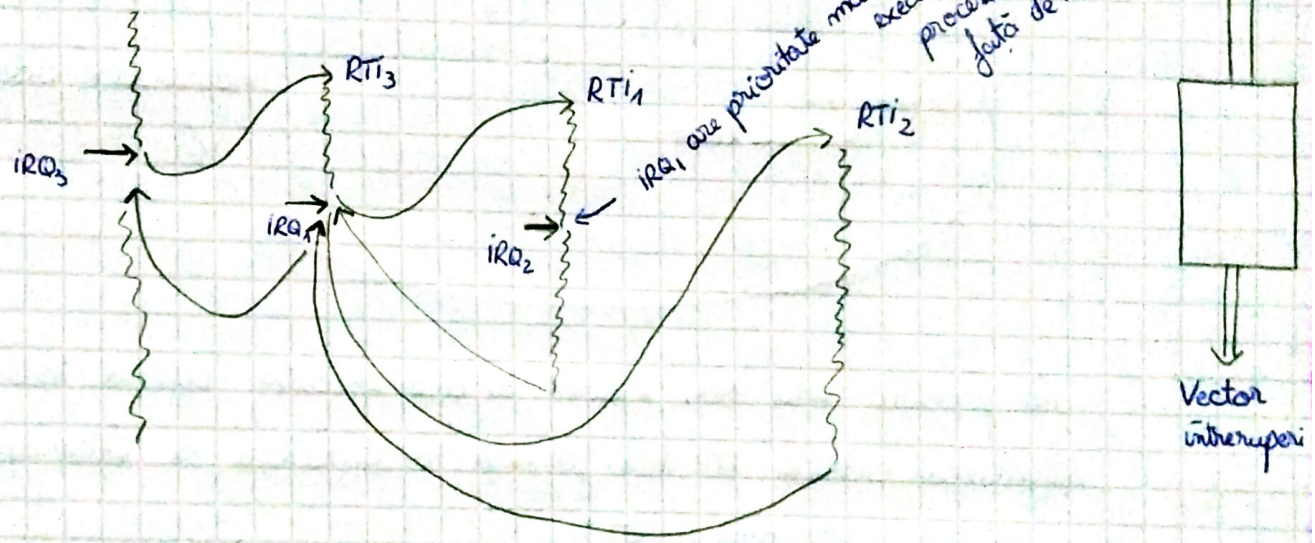
Se stabilesc priorități pentru fiecare nivel de întrerupere astfel încât să putem alege ce întrerupere tratăm dacă apar mai multe în același timp (IRQ_0 → cea mai mare prioritate, IRQ_7 → cea mai mică prioritate)

Putem bloca anumite întreruperi (permitem doar o parte din ele), vom masca acele întreruperi.



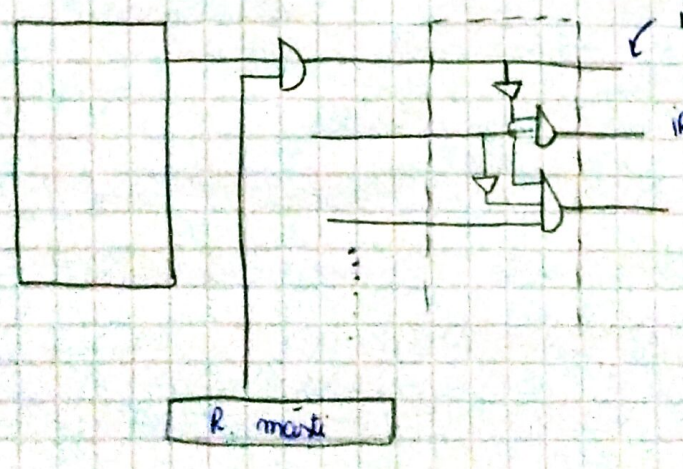
Când se primește o întrerupere, aceasta se memorează și se tratează la terminarea instrucțiunii curente în funcție de prioritate

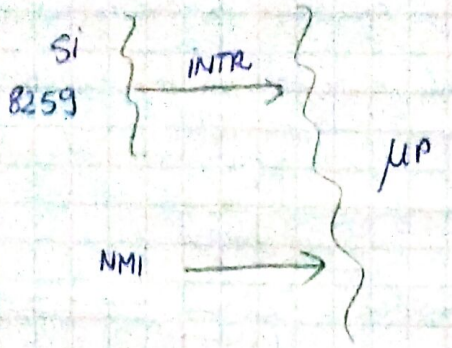


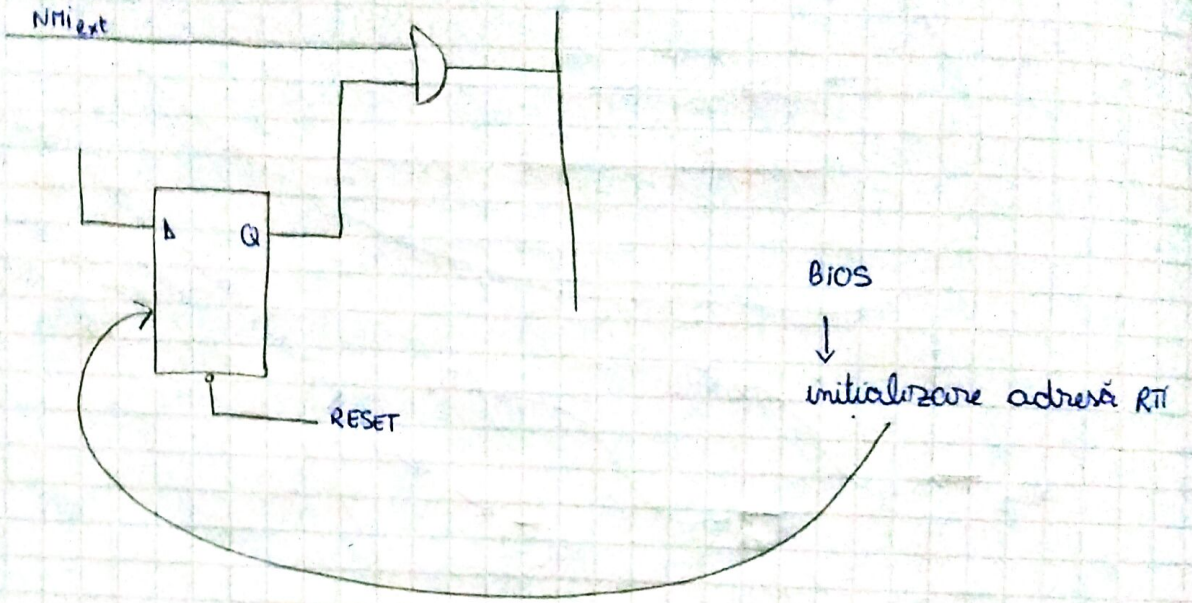


Cel mai defavorabil caz: IRQ_7 e întrerupt de IRQ_6, IRQ_6 e întrerupt de IRQ_5 etc.

Implementarea circuitului de priorități:







La primirea unei NMI, înainte de a o executa BIOS-ul trebuie să inițializeze adresa RTI. (se folosește o amânare a întreruperii până se inițializează adresa RTI)