Specificarea paralelismului operatiilor elementare

Codificarea minimă a unui set de microinstrucțiuni

Punerea problemei

- În proiectarea unităților de comandă microprogramate, după ce s-a determinat setul de microinstrucțiuni complete ce realizează execuția unui microsubbloc în minimum de paşi prin controlul tuturor operațiilor paralele ce se pot executa în sistem, se pune **problema codificării minime a acestui set de microinstrucțiuni**.
- Problema optimizării numărului de biți **constă în a stabili lungimea cuvântului memoriei de control ce specifică microinstrucțiunile, astfel încât aceasta să fie minimă**, având în vedere asigurarea controlului tuturor microoperațiilor paralele specificate de secvența de microinstrucțiuni ce descrie unitatea de comandă.
- Problema generală a **optimizării numărului de biți este o problemă din clasa "NP complete".** Apartenență la clasa "NP complete" a fost demonstrată de Robertson

µo **compatibile**
µo **incompatibile**

- Fie
  - $\mu B(\mu o)=\{\mu o_1, \mu o_2, ..., \mu o_{[\mu B]}\}$ setul de microoperații distincte din cadrul microsubblocului.
  - $\mu PT=\{\mu IC_1, \mu IC_2, ..., \mu IC_{[\mu PT]}\}$ partiția unui microsubbloc în microinstrucțiuni complete şi
- **Def. 1**
- Două microoperații μo_i şi μo_j sunt **compatibile** dacă pentru orice k, $1 <= k <= [\mu PT]$, dacă $\mu o_i \in \mu IC_k$ atunci $\mu o_j \notin \mu IC_k$.
- **Def. 2**
- Două microoperații $\mu o_i \in \mu B(\mu o)$, $\mu o_j \in \mu B(\mu o)$ sunt **incompatibile** dacă există cel puțin o microinstrucțiune completă µIC astfel încât $\mu o_i \in \mu IC_k$ şi $\mu o_j \in \mu IC_k$.

clasă de compatibilitate

- **Def. 3**
- O **clasă de compatibilitate CC(μo)** este un set (subset) al mulțimii MB(μo) în care oricare două microoperații sunt compatibile între ele.
- CC(μo)={μo | pt orice μo_i, $\mu o_j \in$ CC(μo) avem μo_i compatibilă cu μo_j}

- **Def. 4**
- O **clasă de compatibilitate maximă MCC(μo)** este acea clasa de compatibilitate la care nu mai poate fi adăugată nici o microinstrucțiune fără a se pierde compatibilitatea.
- MCC(μo)={μo_j | pt orice $\mu o_i \notin$ MCC(μo), există $\mu o_j \in$ MCC(μo) astfel încât $\mu o_i \in \mu IC_k$ şi $\mu o_j \in \mu IC_k$}.

În mod analog se defineşte clasa de incompatibilitate maximă MIC(μo).

Costul de implementare a unei clase de compatibilitate

- **Def. 5**
- **Costul de implementare a unei clase de compatibilitate** (măsurat în numărul de biți necesari pentru codificare) este dat de implementarea codificării verticale a microoperațiilor ce compun clasa.
- $\text{Cost } CC_i = [\log_2(|CC_i|+1)]$

iar costul total de implementare al cuvântului de control

- $$\text{Cost } CC = \sum_{i=1}^{k} [\log_2(|CC_i|+1)]$$

unde k este numărul de clase de compatibilitate.

- **Obs**
- **O clasă de compatibilitate corespunde unui câmp din cadrul microinstrucțiunii.**

Estimarea costului minim

- Problema codificării minime presupune două faze distincte:
  - enumerarea tuturor claselor de compatibilitate maximă MCC;
  - determinarea unui subset de MCC care să implementeze costul minim pentru codificare.

Costul implementării depinde de

- numărul de MCC necesar pentru acoperirea întregului set de microoperații şi
- numărul de microoperații din fiecare clasă de compatibilitate maximă.

- De notat faptul că un cuvânt din memoria de control, adică o microinstrucțiune completă, este un set de clase incompatibile între microoperațiile componente.
- O clasă de compatibilitate specifică cel mult o microoperație din cadrul unui cuvânt al memoriei de control.

Clase de compatibilitate maximă asociate (AMCC)

- **Def. 6**
- Pentru orice clasă de incompatibilitate IC, clasele de compatibilitate maximă ce conțin un element din IC se numesc **clase de compatibilitate maximă asociate (AMCC) asociate** clasei de incompatibilitate.
- **Prop. 1**
- ***Pentru orice clasă de incompatibilitate maximă MIC reuniunea claselor de compatibilitate maximă asociate acoperă întreg setul de microoperații.***
- **Justificare**
- Fie $MIC(\mu o)=\{\mu o_1, ..., \mu o_{|MIC|}\}$ formată din $|MIC|$ microoperații, unde $|MIC| <= |\mu B|$. Presupunem contrariul şi anume că reuniunea claselor de compatibilitate maximă asociate nu acoperă întreg setul de microoperații. Fie μo_j una din aceste microoperații presupuse neacoperite cu proprietatea că:

$$\mu o_j \in \mu B(\mu o) \text{ dar } \mu o_j \notin MIC(\mu o) \text{ şi } \mu o_j \notin \bigcup_{MIC} AMCC.$$

O clasa de incompatibilitate maximă MIC nu poate acoperi MB(μo).

- **Prop. 2**
- Pentru o clasă de incompatibilitate maximă MIC, **reuniunea oricăror k clase de compatibilitate, k < |MIC|**, nu poate acoperi setul de microoperații μB(μo).

- **Justificare**
- Considerăm MIC(μo)={μo_1, μo_2, ..., $\mu o_{|MIC|}$}. |MIC| este limita inferioară pentru numărul de clase de compatibilitate ce satisface acoperirea, deoarece o clasă de compatibilitate poate acoperi un singur element din MIC.
- Rezultă că orice reuniune de k clase de compatibilitate k < |MIC| nu poate acoperi setul de microoperații.

Clasele MCI si MCC

- **Clasele de incompatibilitate maximale MCI** sunt determinate pe baza grafului de dependență şi a conflictului de resurse şi reprezintă microoperațiile specificate de microinstrucțiunile complete ce descriu microblocul.
- **Clasele MCC sunt determinate din matricea de microoperații** ce indică incompatibilitatea. Aplicarea operatorului SAU între liniile matricii va specifica microoperațiile ce fac parte din clasa de compatibilitate maximă.
- **Calcularea MCC este o binecunoscută problemă în teoria automatelor finite**, existând numeroase metode pentru aceasta soluție. AHO arată că problema calculării MCC este o problema "NP complete".

Prop. 3 partiționare în q câmpuri

- ***Dacă un set de microoperații μB(μo) este partiționat în q câmpuri ale unei microinstrucțiuni, costul minim se va realiza atunci când (q-1) câmpuri specifică câte o microoperație (au un singur bit), iar cel de al q-lea câmp codifică restul de |μB(μo)|-q+1 microoperații.***

Justificare

- Considerăm o partiție arbitrară a LMI biți în q câmpuri. Fie câmpul cu b_{max} biți, câmpul cu lungimea cea mai mare şi fie un oricare alt câmp care are b_i biți.
- Numărul de microoperații ce poate fi codificat de cele două câmpuri este:
- **$NMO = (2^{bmax} - 1) + (2^{bi} - 1)$**
- Facem o modificare în organizarea logică a microinstrucțiunii şi mutăm un bit din câmpul cu b_i biți în câmpul cu b_{max} biți. în acest caz numărul de microoperații ce se pot codifica este:
- **$NMO' = (2^{(bmax+1)} - 1) + (2^{(bi-1)} - 1)$**
- **$NMO' - NMO = 2^{bmax} - 2^{bi} >= 0$ deoarece $b_{max} > b_i$.**
- Deci se pot codifica mai multe operații (microoperații) după modificarea structurii microinstrucțiunii.
- Repetând procesul de mutare a unui bit dintr-un câmp oarecare în câmpul de lungime maximă, numărul de microoperații ce poate fi codificat creşte. Numărul maxim de microoperații ce poate fi codificat rezultă a fi egal cu $(q+2^{(LMI-q+1)}-2)$. Transformând totul în cost se obtine costul minim:
- ***Cost = $q-1+\lfloor \log_2(|\mu B(\mu o)|-q+2) \rfloor$ când (q-1) câmpuri specifică fiecare câte o microoperație, iar ultimul câmp (|μB(μo)|-q+1) microoperații.***

Partiționare a setului MB(μo) microoperații în (q+h) câmpuri

Prop. 4

- ***Nu este posibilă o soluție de partiționare a setului MB(μo) microoperații în (q+h) câmpuri astfel încât costul să fie mai mic decât partiția în q câmpuri.***

Justificare

Fie

- $C = q-1+[\log_2(|\mu B(\mu o)|-q+2)]$ costul minim obținut prin partiția în q câmpuri şi
- $C = q+h-1+[\log_2(|\mu B(\mu o)|-q-h+2)]$ costul minim obținut prin partiția în q+h câmpuri.
- Deosebim 2 cazuri şi anume:
  - 1) $h=1$ şi $|\mu B(\mu o)|-q-1=2^k$
  - 2) $h \neq 1$ sau $|\mu B(\mu o)|-q+1 \neq 2^k$
- în primul caz:
- $C_q=q-1+k+1=q+k$
- $C_{(q+h)}=q+k$
- deci $C_q=C_{(q+h)}$
- în al doilea caz, deoarece :
- $[\log_2(|\mu B(\mu o)|-q+2)]-[\log_2(|\mu B(\mu o)|-q-h+2)] < h$
- rezultă $C_q < C_{(q+h)}$

Obsservatie

Costul minim pentru codificarea setului de microoperații MB(mO) poate să fie mai mare decât cel dat de propoziția 3 adică:

$C \geq q\text{-}1+[\log_2(|\mu B(mO)|\text{-}q+2)]$ când $|MCC|_{max} \leq |\mu B(mO)|\text{-}q$.

într-adevăr costul minim corespunde când:

(q-1) câmpuri specifică fiecare câte o microoperație;

cel de-al q-lea câmp codifică $|\mu B(mO)|\text{-}q+1$ microoperații.

Dacă $|MCC|_{max} \leq |\mu B(mO)|\text{-}q$, rezultă că $|MCC|_{max} < |\mu B(mO)|\text{-}q+1$,

ceea ce ar face ca în ultimul câmp să nu fie toate microoperațiile compatibile.

În acest caz trebuie renunțat la organizarea microinstrucțiunii în (q-1) câmpuri de 1 bit, dar prin aceasta se măreşte şi costul de implementare.

Metodă de minimizare a numărului de biți necesari pentru codificarea microinstrucțiunilor complete

- **1. Se alege clasa de incompatibilitate maximă care are cardinalitatea maximă MIC.**

Fie **$MIC_m \in MIC$** astfel încât $|MIC_m| >= |MIC_j| <$ pentru $j \# m$, $1 <= j <= |MIC|$

Se generează clasele de compatibilitate maximă asociate AMCC clasei MIC.

$AMCC_m$={MCC>= pt orice $\mu o \in MIC_m \exists$ MCC astfel încât $\mu o \in$ MCC}

- **2. Se formează tabela de acoperire modificată prin considerarea numai a claselor de compatibilitate maximă ce aparțin AMCC.**

TAM : $AMCC_m$ x $\mu B(\mu o)$ --> B

Se caută multimea de clase de compatibilitate maximă esențiale $\{MCC_e\}$ inclus în $AMCC_m$ astfel încât există MCC_e unic pentru care : $\mu o_i \in MCC_e$ avem TAM $(MCC_e, \mu o_i)=1$.

Se elimină coloanele corespunzătoare microoperațiilor ce sunt acoperite de clasele esențiale şi cele corespunzătoare microoperațiilor componente ale clasei MIC_m, obținându-se o **tabelă de acoperire redusă** :

TAR : $AMCC_m$ x ($\mu B(\mu o) \setminus MIC_m$) \ $\{MCC_e\}$ --> B

Cont

- **3. Se generează setul soluțiilor de acoperire a microoperațiilor $\{MCC_{ap}\}$: $\{\mu B(\mu o) \setminus MIC_m) \setminus \{MCC_e\}$**

  Fie $SOL=\{SOL_1, SOL_2, ..SOL_p\}$ soluțiile de acoperire în care

  $SOL_j = \wedge / (\{MCC_e\} \cup \{MCC_{ap}\})$

  Se consideră soluția parțială cu cardinalitatea minimă SOL_j.

  Se generează soluția de acoperire a microoperațiilor ce aparțin clasei MIC_m, încă neacoperite: $\{MCC_{am}\}$.

- **4. Se generează setul soluțiilor de acoperire completă:**

  $SOLC_j=SOL_j \wedge \{MCC_{am}\}$.

  Se calculează costul soluției de acoperire completă cu cardinalitatea minimă.

Exemplu

- Considerăm setul de microinstrucțiuni din cadrul microsubblocului :

  $\mu B=\{\mu o_1, \mu o_2, \mu o_3, \mu o_4, \mu o_5, \mu o_6, \mu o_7, \mu o_8, \mu o_9, \mu o_{10}\}$

- Presupunem că pe baza dependenței de date şi a conflictului de resurse între microoperații a rezultat următoarea partiție a microsubblocului :
- $\mu IC_1=\{\mu o_1, \mu o_2, \mu o_4\}$
- $\mu IC_2=\{\mu o_1, \mu o_3, \mu o_5\}$
- $\mu IC_3=\{\mu o_2, \mu o_6, \mu o_8, \mu o_9\}$
- $\mu IC_4=\{\mu o_4, \mu o_5, \mu o_7, \mu o_8\}$
- $\mu IC_5=\{\mu o_3, \mu o_4, \mu o_5, \mu o_6, \mu o_7\}$
- $\mu IC_6=\{\mu o_6, \mu o_9, \mu o_{10}\}$
- $\mu IC_7=\{\mu o_7, \mu o_{10}\}$

$\mu IC_1 = \{\mu o_1, \mu o_2, \mu o_4\}$
$\mu IC_2 = \{\mu o_1, \mu o_3, \mu o_5\}$
$\mu IC_3 = \{\mu o_2, \mu o_6, \mu o_8, \mu o_9\}$
$\mu IC_4 = \{\mu o_4, \mu o_5, \mu o_7, \mu o_8\}$
$\mu IC_5 = \{\mu o_3, \mu o_4, \mu o_5, \mu o_6, \mu o_7\}$
$\mu IC_6 = \{\mu o_6, \mu o_9, \mu o_{10}\}$
$\mu IC_7 = \{\mu o_7, \mu o_{10}\}$

$\mu o \backslash \mu o$	μo_1	μo_2	μo_3	μo_4	μo_5	μo_6	μo_7	μo_8	μo_9	μo_{10}
μo_1	X	X	X	X	X					
μo_2	X	X		X		X		X	X	
μo_3										
μo_4										
μo_5										
μo_6										
μo_7										
μo_8										
μo_9										
μo_{10}										

Tabela de incompatibilitate

- $\mu IC_1=\{\mu o_1, \mu o_2, \mu o_4\}$
- $\mu IC_2=\{\mu o_1, \mu o_3, \mu o_5\}$
- $\mu IC_3=\{\mu o_2, \mu o_6, \mu o_8, \mu o_9\}$
- $\mu IC_4=\{\mu o_4, \mu o_5, \mu o_7, \mu o_8\}$
- $\mu IC_5=\{\mu o_3, \mu o_4, \mu o_5, \mu o_6, \mu o_7\}$
- $\mu IC_6=\{\mu o_6, \mu o_9, \mu o_{10}\}$
- $\mu IC_7=\{\mu o_7, \mu o_{10}\}$

$\mu o \backslash \mu o$	μo_1	μo_2	μo_3	μo_4	μo_5	μo_6	μo_7	μo_8	μo_9	μo_{10}
μo_1	x	x	x	x	x					
μo_2	x	x		x		x		x	x	
μo_3	x		x	x	x	x	x			
μo_4	x	x	x	x	x	x	x	x		
μo_5	x		x	x	x	x	x	x		
μo_6		x	x	x	x	x	x	x	x	x
μo_7			x	x	x	x	x	x		x
μo_8		x		x	x	x	x	x	x	
μo_9		x				x		x	x	x
μo_{10}						x	x		x	x

Clasele maximale de incompatibilitate

- $MIC_1=\{\mu o_1, \mu o_2, \mu o_4\}$
- $MIC_2=\{\mu o_1, \mu o_3, \mu o_4, \mu o_5\}$
- $MIC_3=\{\mu o_2, \mu o_4, \mu o_6, \mu o_8\}$
- $MIC_4=\{\mu o_2, \mu o_6, \mu o_8, \mu o_9\}$
- **$MIC_5=\{\mu o_3, \mu o_4, \mu o_5, \mu o_6, \mu o_7\}$**
- **$MIC_6=\{\mu o_4, \mu o_5, \mu o_6, \mu o_7, \mu o_8\}$**
- $MIC_7=\{\mu o_6, \mu o_7, \mu o_{10}\}$
- $MIC_8=\{\mu o_6, \mu o_9, \mu o_{10}\}$

- **MIC_{max} = MIC_5 sau MIC_6**

$\mu o \backslash \mu o$	μo_1	μo_2	μo_3	μo_4	μo_5	μo_6	μo_7	μo_8	μo_9	μo_{10}
μo_1	x	x	x	x	x					
μo_2	x	x		x		x		x	x	
μo_3	x		x	x	x	x	x			
μo_4	x	x	x	x	x	x	x	x		
μo_5	x		x	x	x	x	x	x		
μo_6		x	x	x	x	x	x	x	x	x
μo_7			x	x	x	x	x	x		x
μo_8		x		x	x	x	x	x	x	
μo_9		x				x		x	x	x
μo_{10}						x	x		x	x

Clasele maximale de compatibilitate

- $MCC_1=\{\mu o_1, \mu o_6\}$
- $MCC_2=\{\mu o_1, \mu o_7, \mu o_9\}$
- $MCC_3=\{\mu o_1, \mu o_8, \mu o_{10}\}$
- $MCC_4=\{\mu o_2, \mu o_3, \mu o_{10}\}$
- $MCC_5=\{\mu o_2, \mu o_5, \mu o_{10}\}$
- $MCC_6=\{\mu o_2, \mu o_7\}$
- $MCC_7=\{\mu o_3, \mu o_8, \mu o_{10}\}$
- $MCC_8=\{\mu o_3, \mu o_9\}$
- $MCC_9=\{\mu o_4, \mu o_9\}$
- $MCC_{10}=\{\mu o_4, \mu o_10\}$
- $MCC_{11}=\{\mu o_5, \mu o_9\}$

$\mu o \setminus \mu o$	μo_1	μo_2	μo_3	μo_4	μo_5	μo_6	μo_7	μo_8	μo_9	μo_{10}
μo_1	x	x	x	x	x					
μo_2	x	x		x		x		x	x	
μo_3	x		x	x	x	x	x			
μo_4	x	x	x	x	x	x	x	x		
μo_5	x		x	x	x	x	x	x		
μo_6		x	x	x	x	x	x	x	x	x
μo_7			x	x	x	x	x	x		x
μo_8		x		x	x	x	x	x	x	
μo_9		x				x		x	x	x
μo_{10}						x	x		x	x

Tabela de acoperire a microoperațiilor de către clasele maximale de compatibilitate

- $MCC_1=\{\mu o_1, \mu o_6\}$
- $MCC_2=\{\mu o_1, \mu o_7, \mu o_9\}$
- $MCC_3=\{\mu o_1, \mu o_8, \mu o_{10}\}$
- $MCC_4=\{\mu o_2, \mu o_3, \mu o_{10}\}$
- $MCC_5=\{\mu o_2, \mu o_5, \mu o_{10}\}$
- $MCC_6=\{\mu o_2, \mu o_7\}$
- $MCC_7=\{\mu o_3, \mu o_8, \mu o_{10}\}$
- $MCC_8=\{\mu o_3, \mu o_9\}$
- $MCC_9=\{\mu o_4, \mu o_9\}$
- $MCC_{10}=\{\mu o_4, \mu o_10\}$
- $MCC_{11}=\{\mu o_5, \mu o_9\}$

MCC\μo	μo_1	μo_2	μo_3	μo_4	μo_5	μo_6	μo_7	μo_8	μo_9	μo_{10}
MCC_1	x					x				
MCC_2	x						x		x	
MCC_3	x							x		x
MCC_4		x	x							x
MCC_5		x			x					x
MCC_6		x					x			
MCC_7			x					x		x
MCC_8			x						x	
MCC_9				x					x	
MCC_{10}				x						x
MCC_{11}					x				x	

Obs

- Se observă că există două clase maximale de incompatibilitate cu cardinalitate 5 (MIC_5 şi MIC_6).
- Costul minim absolut, conform proprietății 4 este **C = 7**.
- ***Să considerăm $AMCC_6$ - clasele maximale de compatibilitate asociate clasei maximale de incompatibilitate MIC_6 :***

  $AMCC_6$={MCC_1, MCC_2, MCC_3, MCC_5, MCC_6, MCC_7, MCC_9, MCC_{10}, MCC_{11})
- Conform propoziției 1 clasele maximale de compatibilitate din **$AMCC_6$** acoperă toate microoperațiile microsubblocului.

Tabela de acoperire modificată

	μo_1	μo_2	μo_3	μo_4	μo_5	μo_6	μo_7	μo_8	μo_9	μo_{10}
MCC_1	x					**x**				
MCC_2	x						x		x	
MCC_3	x							x		x
MCC_5		x			x					x
MCC_6		x					x			
MCC_7			**x**					x		x
MCC_9				x					x	
MCC_{10}				x						x
MCC_{11}					x				x	

Se observă că MCC_1 şi MCC_7 sunt esențiale şi vor face parte din soluția finală.

Tabela de acoperire redusă

- Considerăm, conform algoritmului, acoperite microoperațiile:
  - incluse în clasele de compatibilitate maximă **MCC_1 şi MCC_7** esențiale şi
  - cele componente ale clasei de incompatibilitate maximă **MIC_6**.
- Astfel, din tabela de acoperire se elimină microoperațiile
  - **μo_1, μo_6, μo_3, μo_8, μo_{10}, respectiv μo_4, μo_5, μo_7.**

MCC \µo	μo_2	μo_9
MCC_2		x
MCC_5	x	
MCC_6	x	
MCC_9		x
MCC_{11}		x

Acoperirea microoperațiilor μo_2 şi μo_9

- Se observă că tabela de acoperire s-a redus substanțial. Acoperirea microoperațiilor μo_2 şi μo_9 se poate face cu ajutorul următoarelor clase maximale de compatibilitate :
- MCC_2, MCC_5
- MCC_2, MCC_6
- MCC_5, MCC_9
- MCC_5, MCC_{11}
- MCC_6, MCC_9
- MCC_6, MCC_{11}

MCC \µo	μo_2	μo_9
MCC_2		x
MCC_5	x	
MCC_6	x	
MCC_9		x
MCC_{11}		x

Setul soluțiilor

- SOL = {SOL_1, SOL_2, SOL_3, SOL_4, SOL_5, SOL_6} astfel :
- SOL = {
  - **MCC_1 MCC_7** MCC_2 MCC_5 ;
  - **MCC_1 MCC_7** MCC_2 MCC_6 ;
  - **MCC_1 MCC_7 MCC_5 MCC_9 ;**
  - **MCC_1 MCC_7** MCC_5 MCC_{11} ;
  - **MCC_1 MCC_7** MCC_6 MCC_9 ;
  - **MCC_1 MCC_7** MCC_6 MCC_{11}
  - }
- Considerând soluția : **MCC_1 MCC_7 MCC_5 MCC_9 $\in$ SOL**, se acoperă toate microoperațiile cu excepția $\mu o_7 \in MIC_6$.
- Pentru acoperirea lui μo_7 se poate lua una din clasele maximale de compatibilitate MCC_2 sau MCC_6.
- $SOLC_{31}$={MCC_1, MCC_7, MCC_5, MCC_9, MCC_2}
- $SOLC_{32}$={MCC_1, MCC_7, MCC_5, MCC_9, MCC_6}

Setul soluțiilor complete

- $SOLC_3 = \{SOLC_{31}, SOC_{32}\}$

unde

- **$SOLC_{31}=\{MCC_1, MCC_7, MCC_5, MCC_9, MCC_2\}$**
- $SOLC_{32}=\{MCC_1, MCC_7, MCC_5, MCC_9, MCC_6\}$

Ambele soluții complete au aceeaşi cardinalitate ce specifică numărul minim de câmpuri necesar pentru codificarea microinstrucțiunilor ce descriu microsubblocul.

Alegând $SOLC_{31}$, rezultă următoarea grupare a microoperațiilor:

MC2 MC7 MC5 MC9 MC1

$\{ \mu o_1 \mu o_7 \mu o_9 ; \mu o_3 \mu o_8 \mu o_{10} ; \mu o_2 \mu o_5 ; \mu o_4 ; \mu o_6 \}$

Microperatiile cuprinse intr-un camp nu mai apar in alte campuri

Sunt mai multe posibilittati

MC2 MC5 MC7 MC9 MC1

$\{ \mu o_1 \mu o_7 \mu o_9 ; \mu o_2 \mu o_5 \mu o_{10} ; \mu o_3 \mu o_8; \mu o_4 ; \mu o_6 \}$

- ce necesită 8 biți pentru codificare.

- **$MCC_1=\{\mu o_1, \mu o_6\}$**
- **$MCC_2=\{\mu o_1, \mu o_7, \mu o_9\}$**
- $MCC_3=\{\mu o_1, \mu o_8, \mu o_{10}\}$
- $MCC_4=\{\mu o_2, \mu o_3, \mu o_{10}\}$
- **$MCC_5=\{\mu o_2, \mu o_5, \mu o_{10}\}$**
- $MCC_6=\{\mu o_2, \mu o_7\}$
- **$MCC_7=\{\mu o_3, \mu o_8, \mu o_{10}\}$**
- $MCC_8=\{\mu o_3, \mu o_9\}$
- **$MCC_9=\{\mu o_4, \mu o_9\}$**
- $MCC_{10}=\{\mu o_4, \mu o_10\}$
- $MCC_{11}=\{\mu o_5, \mu o_9\}$