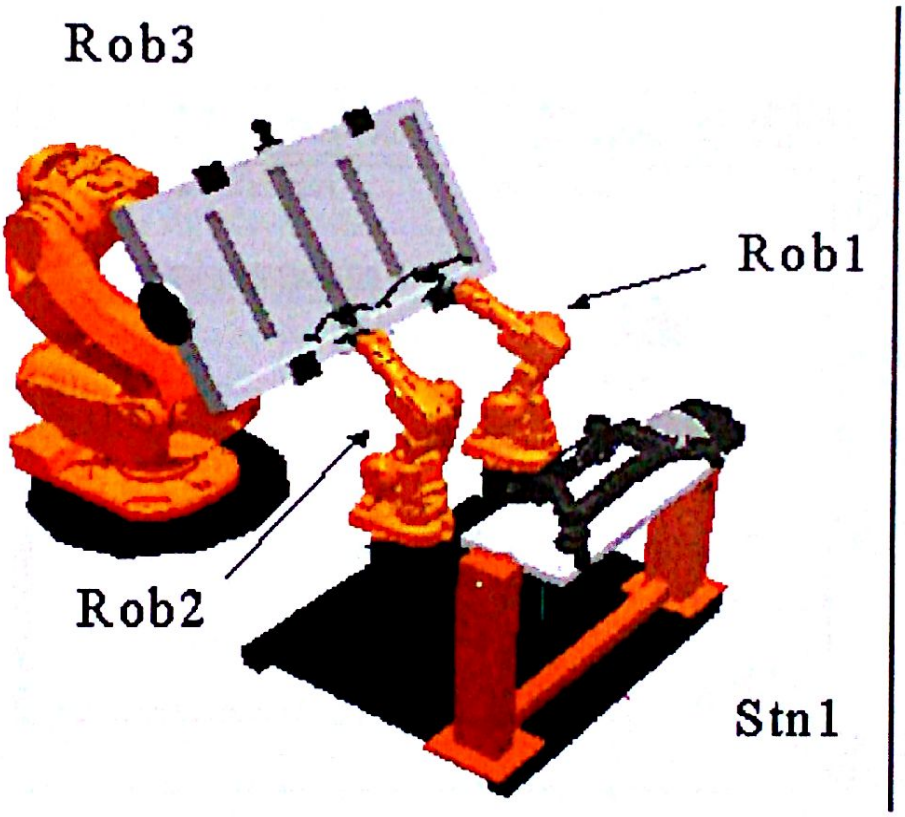
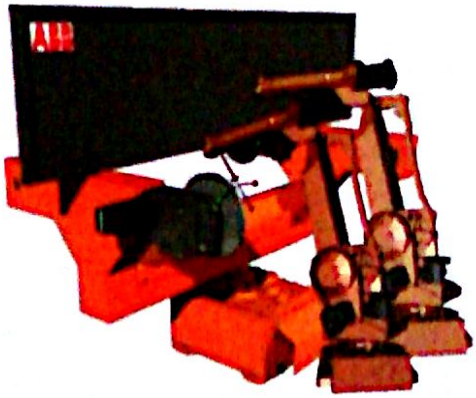
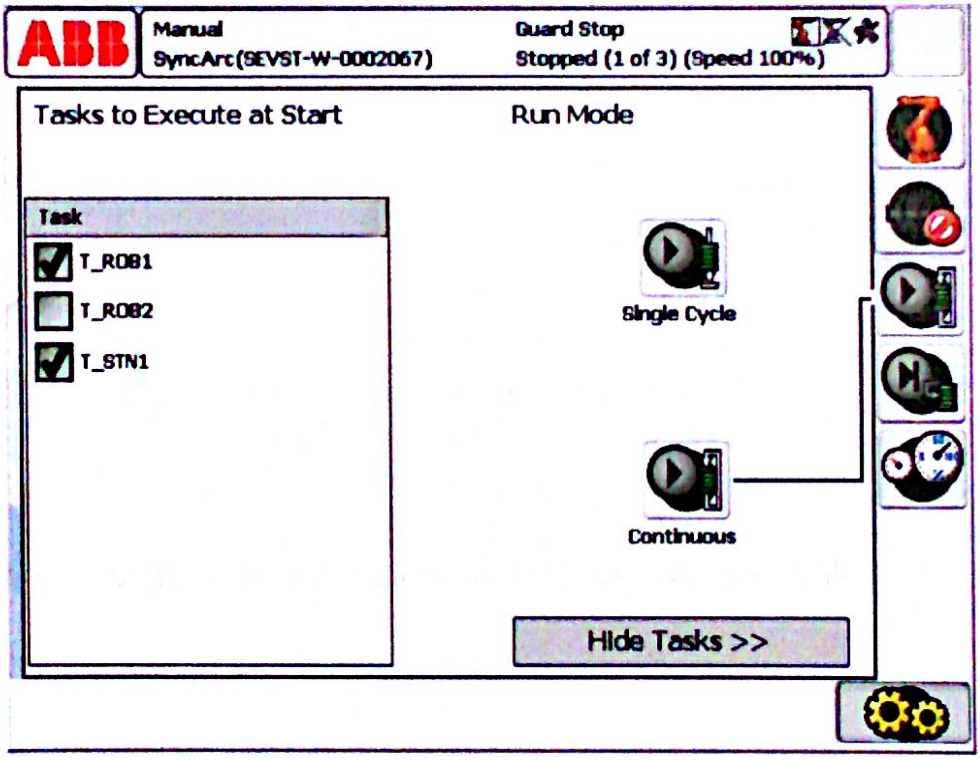
A MultiMove alkalmazás előnyei:

- több robot is tud ugyanabban az objektumhoz rögzített koordináta-rendszerben dolgozni
- az egy robot által mozgatott a koordináta-rendszert a többi robot képes követni (robothoz rögzített koordináta-rendszer)

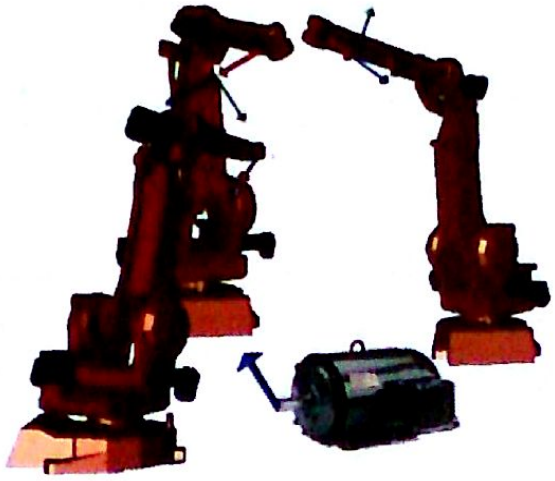


- több robot együttműködésben képes nehéz objektumok emelésére
- lehetőség van bármely ABB robottípusok kombinálására
- Lehetőség van a különböző robot taskok különálló futtatására, tesztelés, vagy akár üzem közben is!





- Bármely Task futtathatja a saját manipulátorát félig – vagy teljesen függetlenült!



3, Alapfogalmak:

Koordináció:

Az objektumhoz rögzített koordináta-rendszert az azt használó robot képes követni

Szinkronizáció:

A mozgások egymással párhuzamosan történnek; a párhuzamos végrehajtást nem térben, hanem időben érjük

Pozícionáló:

Tcp-vel nem rendelkező mechanikai egység, amely csak csukóirányú mozgások végrehajtására képes. Egy vagy több tengellyel rendelkezik, objektum koordináta-rendszert képes kezelni illetve mozgatni.

4, MultiMove-val kapcsolatos mozgásvezérlési formák

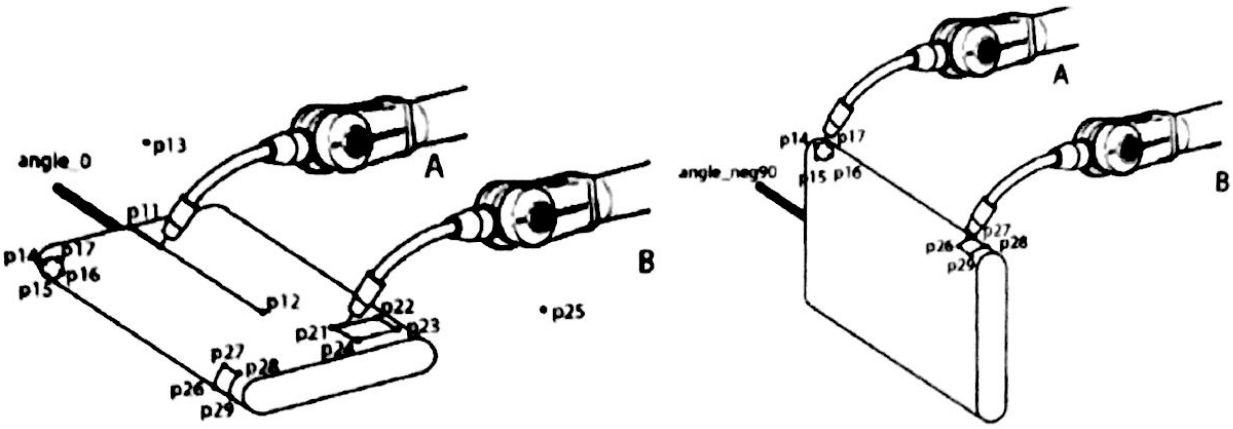
Az IRC5 vezérlők felhasználása és a programstruktúra tervezés első, célszerű lépése annak felmérése, hogy a termelés során a robotok (és a pozícionálók) milyen szinten és szinkronitással kell hogy együttműködjenek.

Példaképpen, ha egy cellában két robot található, és :

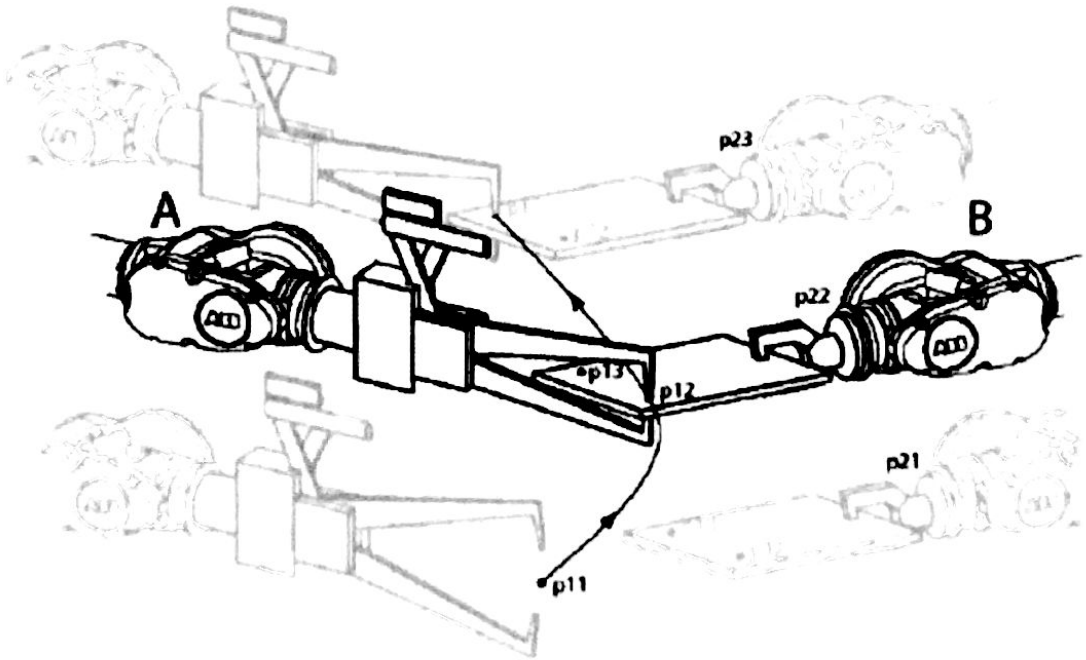
- mindegyik a saját BASE koordináta-rendszerében dolgozik egymástól függetlenül, különböző munkadarabokkal, akkor a mozgási szervezettségük együttműködés szempontjából *TELJESEN FÜGGETLEN MOZGÁS*

- pozicionálón elhelyezett munkadarabokkal, mindegyik a saját BASE koordináta-rendszerében dolgoznak, és az asztalnak fordulnia kell; hogy ne ütközzenek, ekkor a két robot és az asztal programja egy adott pontban bevárják egymást, s csak ha a három programmutató (PP) mindegyike (s így a manipulátorok is) elérkezett a kívánt pozícióba, akkor mehet a mozgás tovább. Ezt a formát *FÉLIG FÜGGELEN MOZGÁS*-nAk nevezzük

- mindegyik a saját BASE koordináta-rendszerében dolgozik, egy munkadarabon, és a feladat megkívánja az összes együtt-mozgás közötti szinkronizációt minden lépésnél, akkor *SZINKRON MOZGÁS*-ról beszélünk



- csak az egyik robot dolgozik a saját BASE koordináta-rendszerében, a második robot mozgása az előzőéhez kötött, az első robothoz rögzített koordináta-rendszer szerint mozog, ezt a mozgást *SZINKRON KOORDINÁLT MOZGÁS* -nak nevezzük.



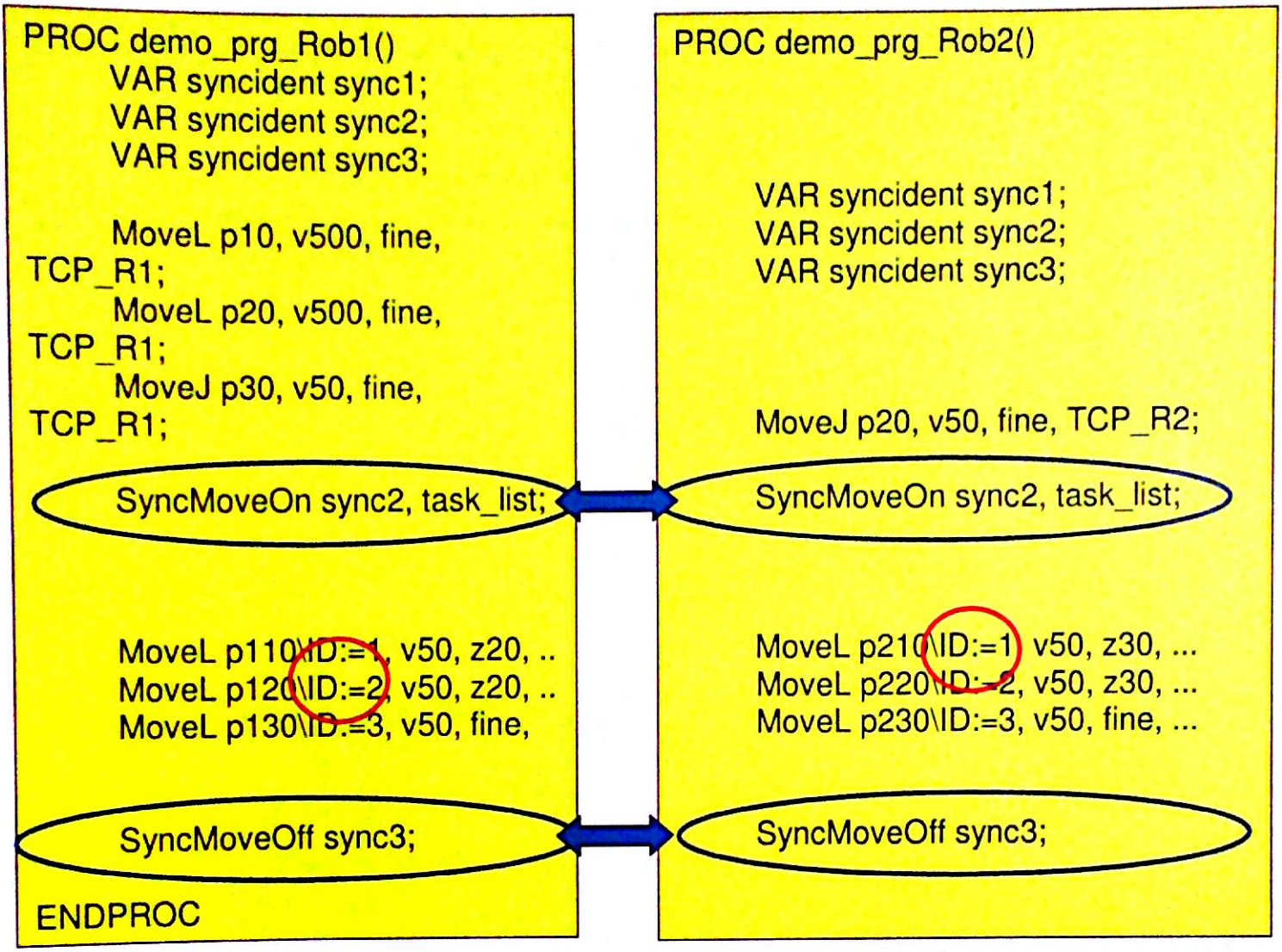
Példa az előbb felsorolt mozgásezérlési formákra:

4.1. „Teljes" független mozgás...

```
PROC demo_prg_Rob1()

        MoveL p10, v500, fine, TCP_R1;
        MoveJ p20, v50, fine, TCP_R1;
        ....

ENDPROC
```

```
PROC demo_prg_Rob2()

        MoveL p10, v500, fine, TCP_R2;
        MoveL p20, v500, fine, TCP_R2;
        MoveL p30, v500, fine, TCP_R2;
        MoveJ p40, v50, fine, TCP_R2;
        ....

ENDPROC
```

4.3 „Szinkronizált mozgás”



Látható, hogy a SyncMoveOn szinkronmozgás-utasításon kívül a sorokat azonosítókkal, azaz angolul „ID”-kal is elláttuk.

Fontos tudnivaló, hogy szinkron mozgatásnál a sebességek nem az általunk definiált értékkel fognak végrehajtóni, hanem az úgynevezett Path Planner (pályagenerátor) által meghatározott értékkel.

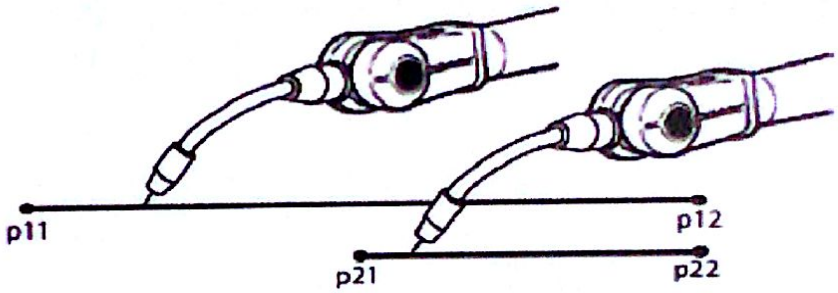
Ezt azt jelenti, hogy ha két Taskot szinkron mozgás módban futtatunk, akkor az első Task szinkron mozgás idejéhez állítja be a második Task a sebességet, nem pedig az általunk definiált sebességet használja (csakúgy, mint az ArcL utasítás, hegesztés közben). A szinkron mozgások így azonos idő alatt hajtódnak végre.

Part of T_ROB1 task program:

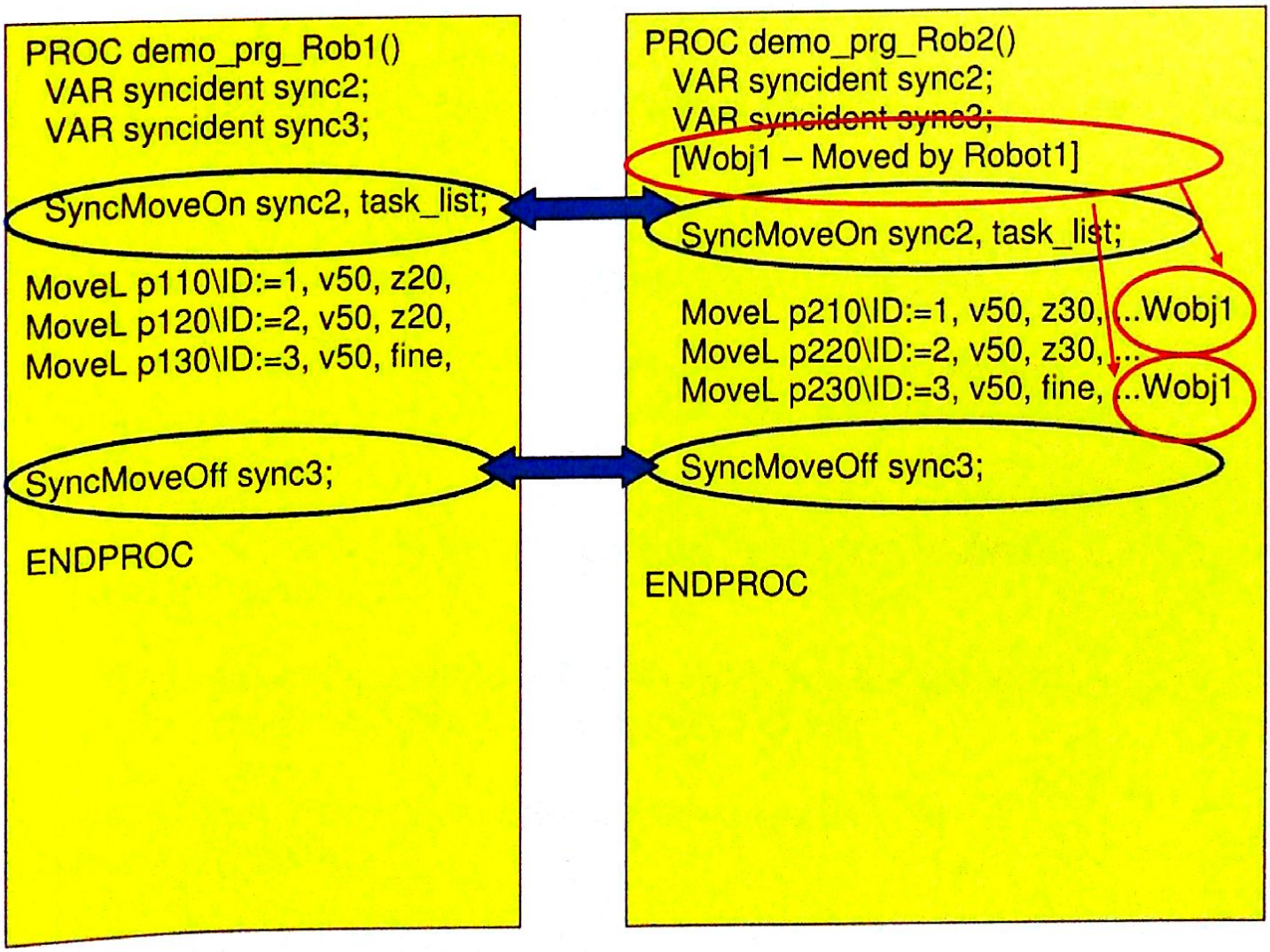
```
MoveJ p11, v1000, fine, tool1;
SyncMoveOn sync1, all_tasks;
MoveL p12\ID:=10, v100, fine, tool1;
```

Part of T_ROB2 task program:

```
MoveJ p21, v1000, fine, tool2;
SyncMoveOn sync1, all_tasks;
MoveL p22\ID:=10, v500, fine, tool2;
```



4.4. Szinkron-koordinált mozgás

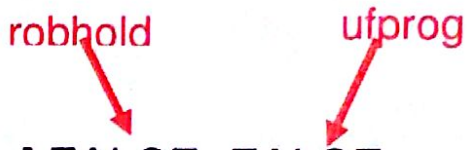


WorkObject (Wobj) definíció szinkron koordinált mozgás esetén:

-Robhold: Az adott RAPID programban szereplő robot hordozza a WorkObjectet?

-ufprog. A WorkObject állandó vagy mozgatott ?

A lenti példában láthatjuk, a WorkObject a ROB_1 által mozgatott, és bármely másik robot által használható.




```
Pers wobjdata wobj_rob1 := [ FALSE, FALSE,
                    "ROB_1",
[ [ 0, 0, 0], [1, 0, 0, 0] ], [ [0, 0, 250], [1, 0, 0, 0] ] ];
```

- A WorkObject-nek abban a Rapid-task-ban kell lennnie definiálva, ahol használjuk.

- MultiMove nélkül a koordinált WorkObject minden mechanikai egység által használható és mozgatható egy TASK-on belül.

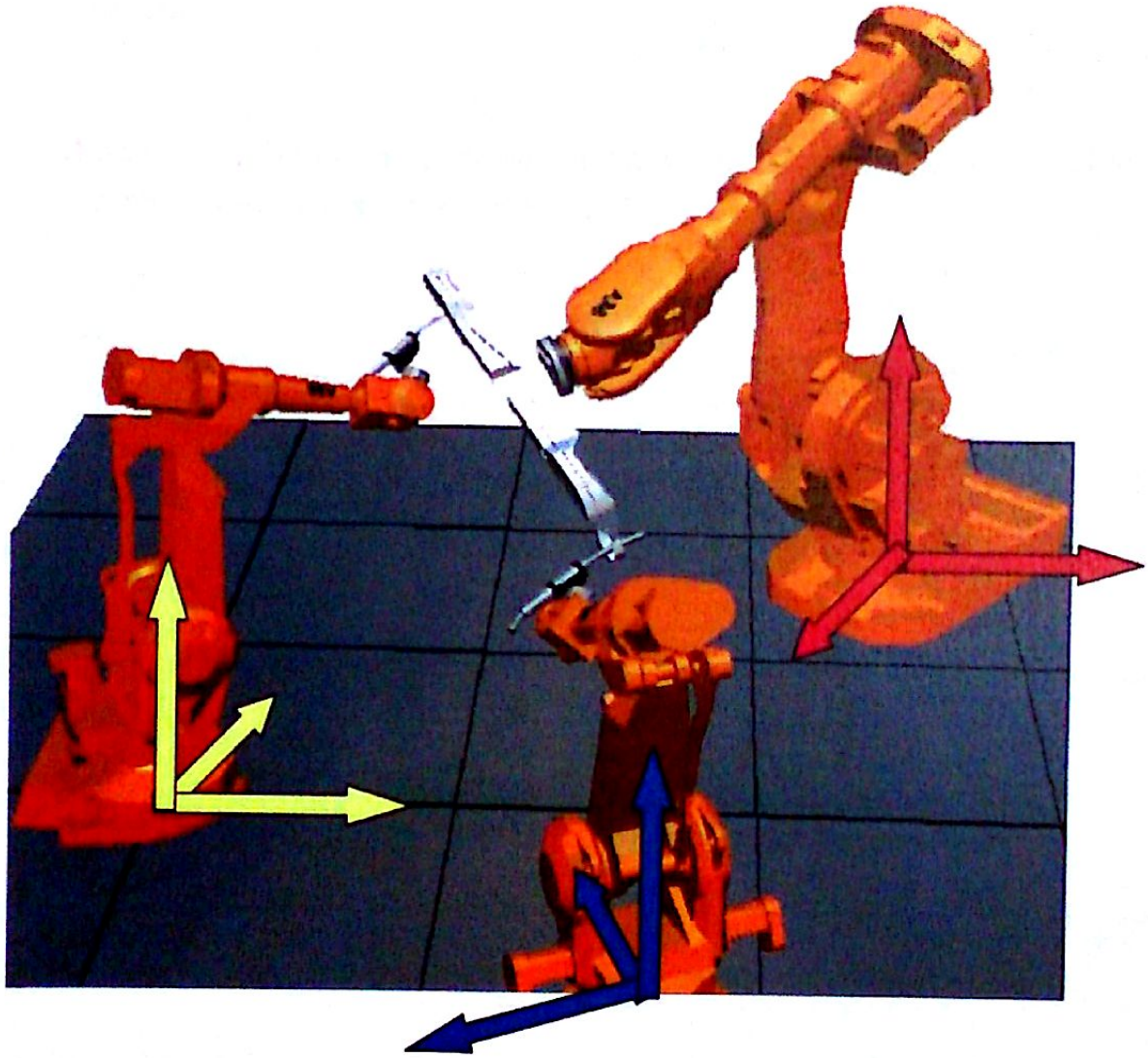
- With MultiMove a coordinated WorkObject can be moved by a Mechanical Unit in one Rapid-task and used by a Mechanical Unit in another Rapid-task

- Minden szinkronizált Rapid-programnak ugyanannyi szinkron-utasítást tartalmaznia!

- Manipulátorhoz rögzíthetünk WorkObjectet. Ennek függvényében a többi manipulátor képes ennek a mozgását követni!

- Csak akkor lehetséges, ha a manipulátorok egy Coordinated Groups-ba tartoznak

5, MultiMove kalibráció...



A MultiMove rendszerek kalibrálási eljárása alapjaiban megegyezik az alap S4C vezérlő rendszerek módszereivel, azzal a különbséggek, hogy egy robotot ki kell neveznünk referenciának, amihez képest megáhatározhatjuk a többi manipulátor helyzetét.

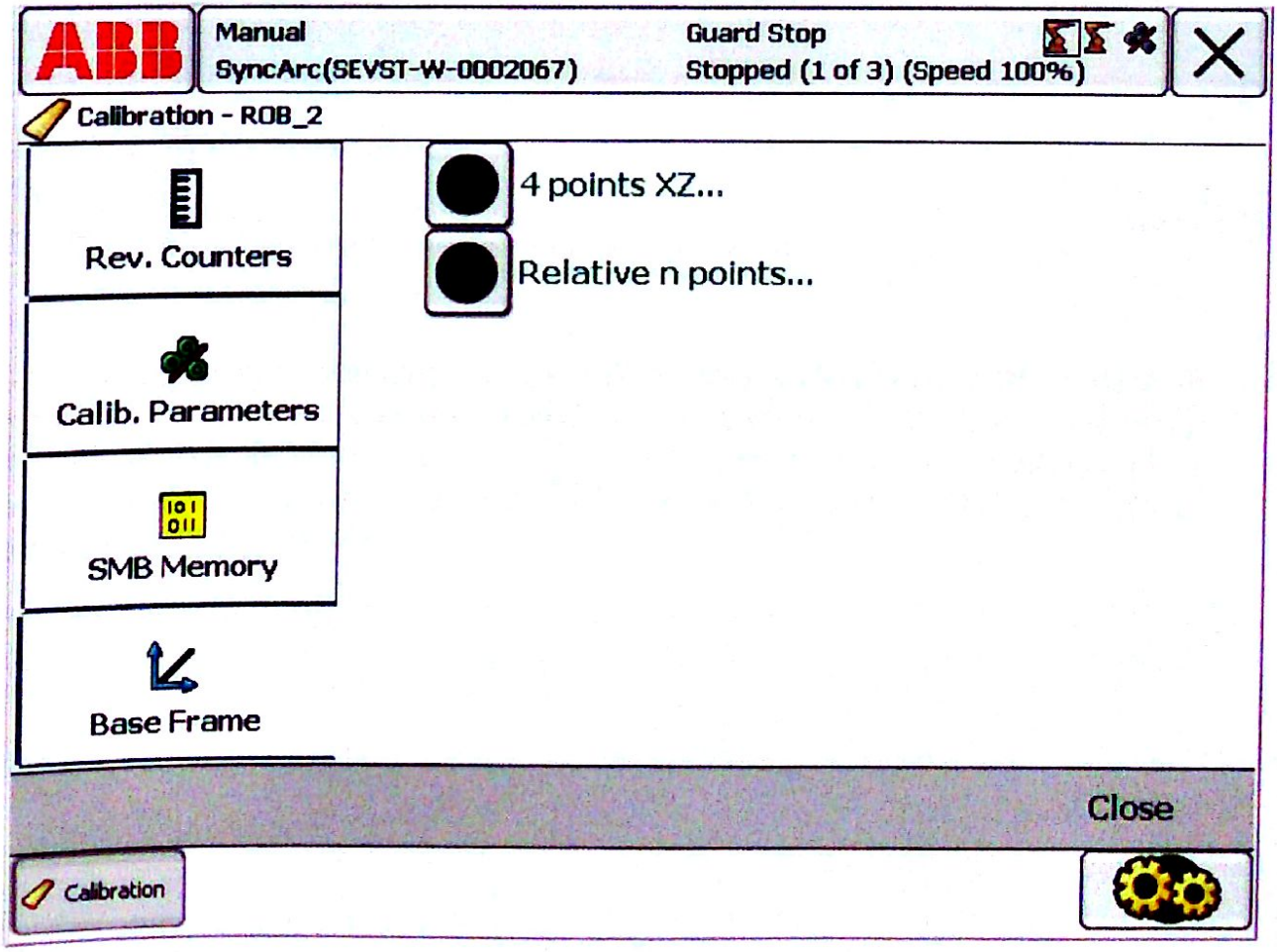
Két robot kalibrációja

- Az ötlet azon alapul, hogyan lehet már előzőleg definiált TCP-ket együtt mozgatni;

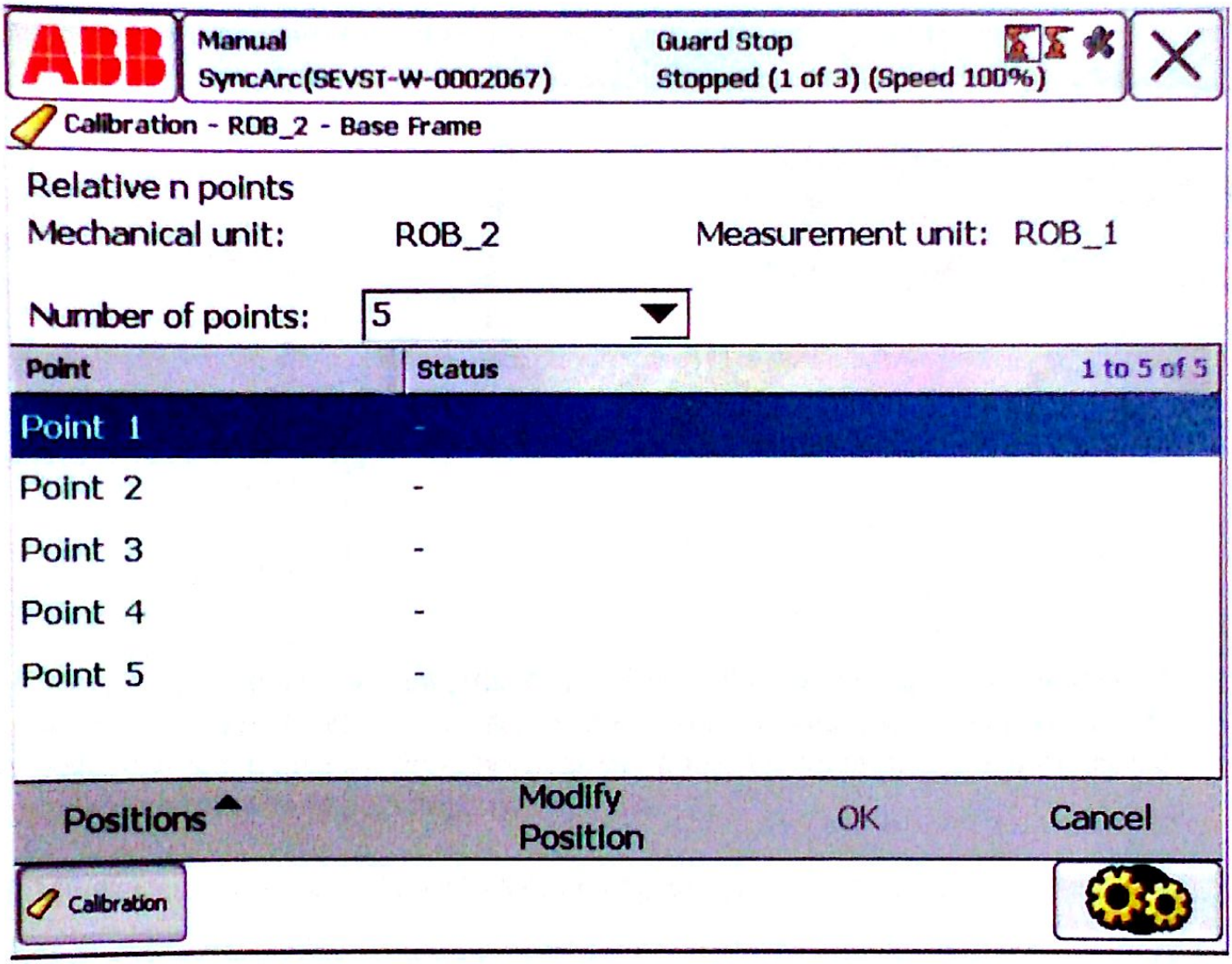
- Mielőtt a kalibráció elkezdődne, a szerszámnak a Mozgatás ablakban az érintett robothoz kell lennie kiválasztva

1, Főablakban kiválasztjuk a Calibration menüpontot, majd abban a kalibrálni kívánt robotot

2, Kiválasztjuk a BaseFrame menüpontot, s az alábbi ablak jelenik meg



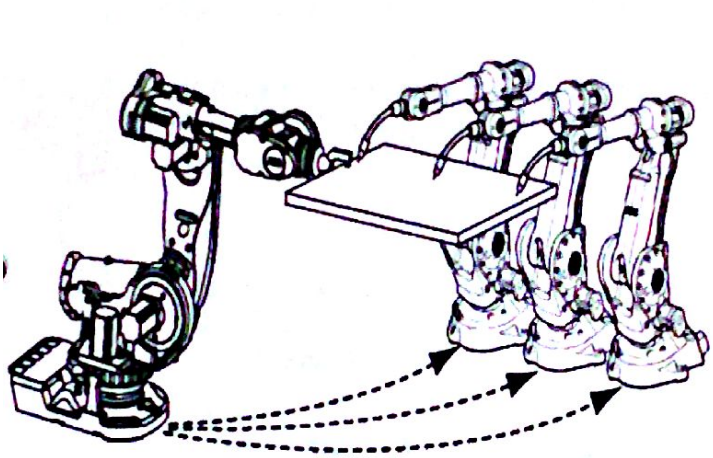
3. Itt a Relative n points pontot választjuk. A minél nagyon pontosság és a mérési szórás elkerülése végett célszerű minimum 5 pontban elvégezni a pozíciók felvételét.

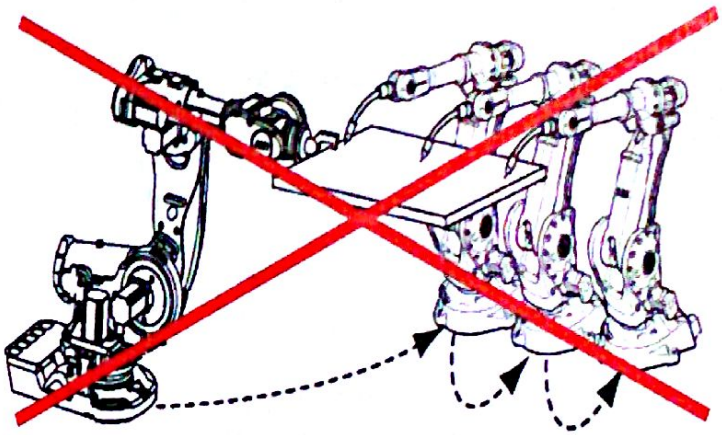


A kurzorral az első pontra lépünk, a két TCP-t összejáratjuk (egy pontba, célszerű erre a célra külön szerszámot készíttetni a rajznak megfelelően. A pozíció módosítás (Modify Position) után a kurzorral a következő sorra lépünk, és egy újabb pozíciót veszünk fel, lehetőleg minél távolabb az előzőtől. Újabb módosítás, és így tovább.

Több robot kalibrációja

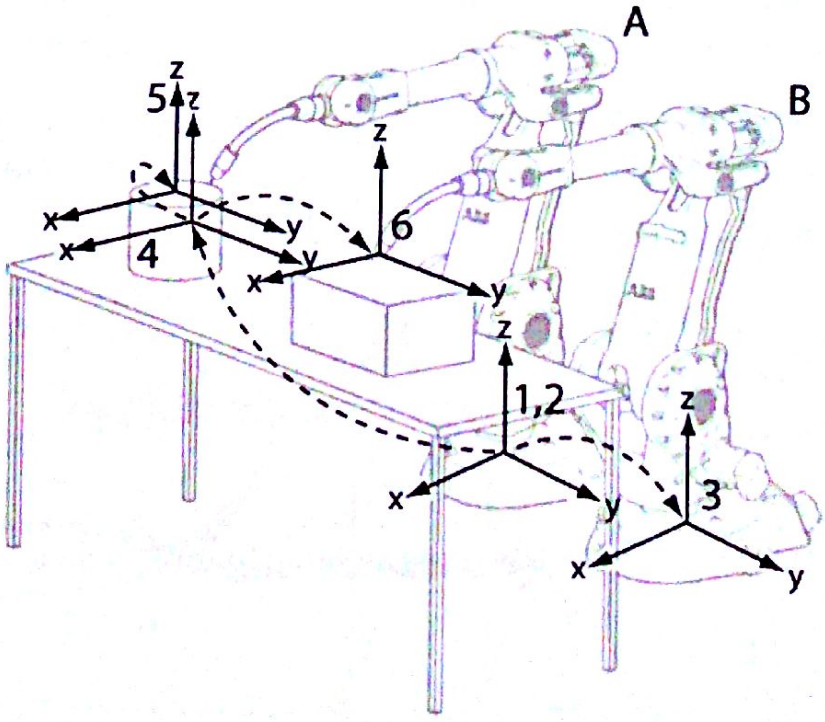
A lehető legpontosabb kalibráció érdekében kerüljük a lánc-kalibrálást, lehetőleg az összes robotot egyszerre állítsuk be a mozgatandó objektumhoz!





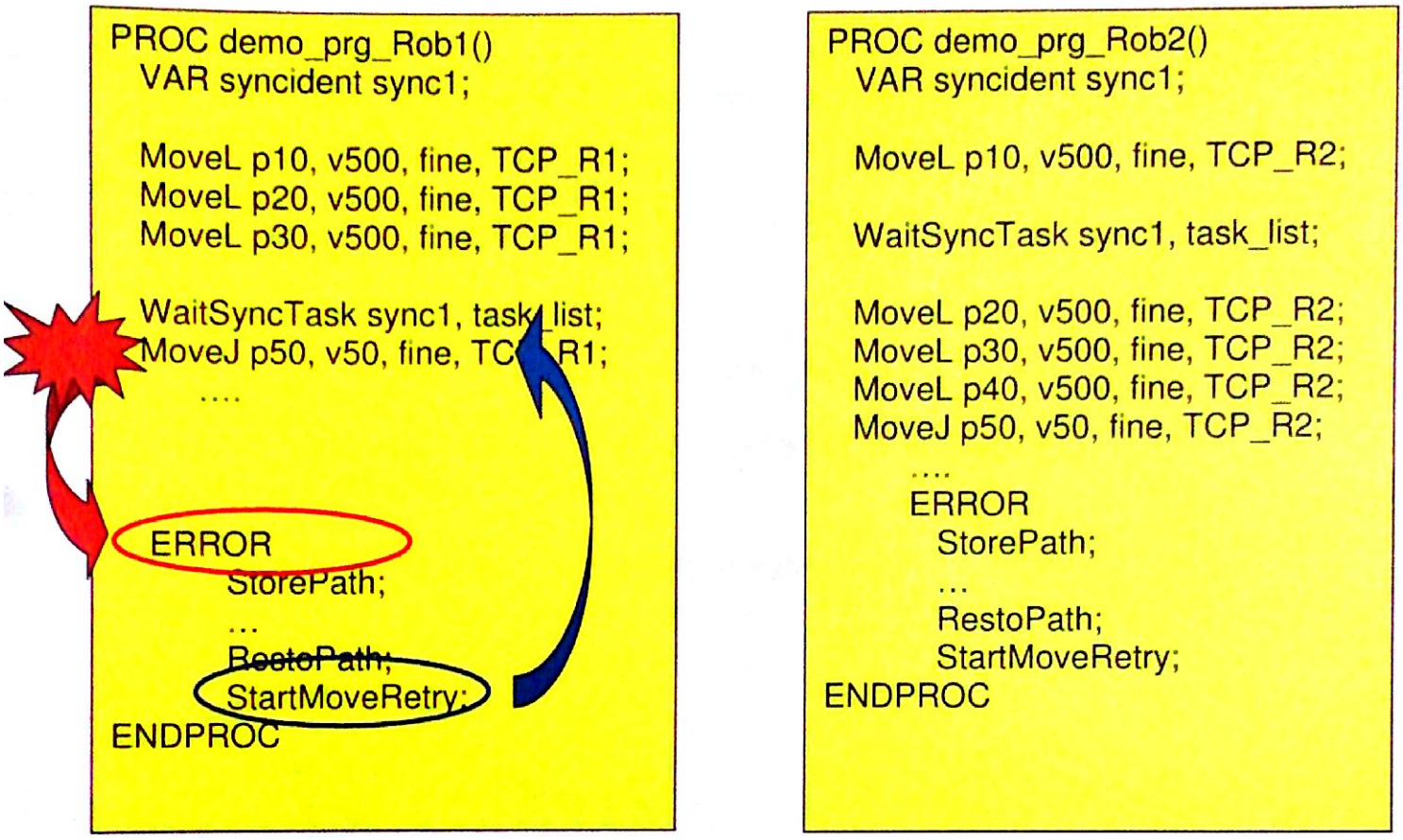
Ennek megfelelően, a képen látható mozgatott objektumon célszerű mindhárom robothoz egy-egy bejelölést hozzárendelni, majd a kalibrációs ablakot három példányban megnyitva az egy-egy robothoz, a már előzőleg megismert eljárással mindhárom robot öt-öt pozícióját módosítjuk.

Minden más kalibrációt hasonló módon végzünk, mint az S4CPlus esetében

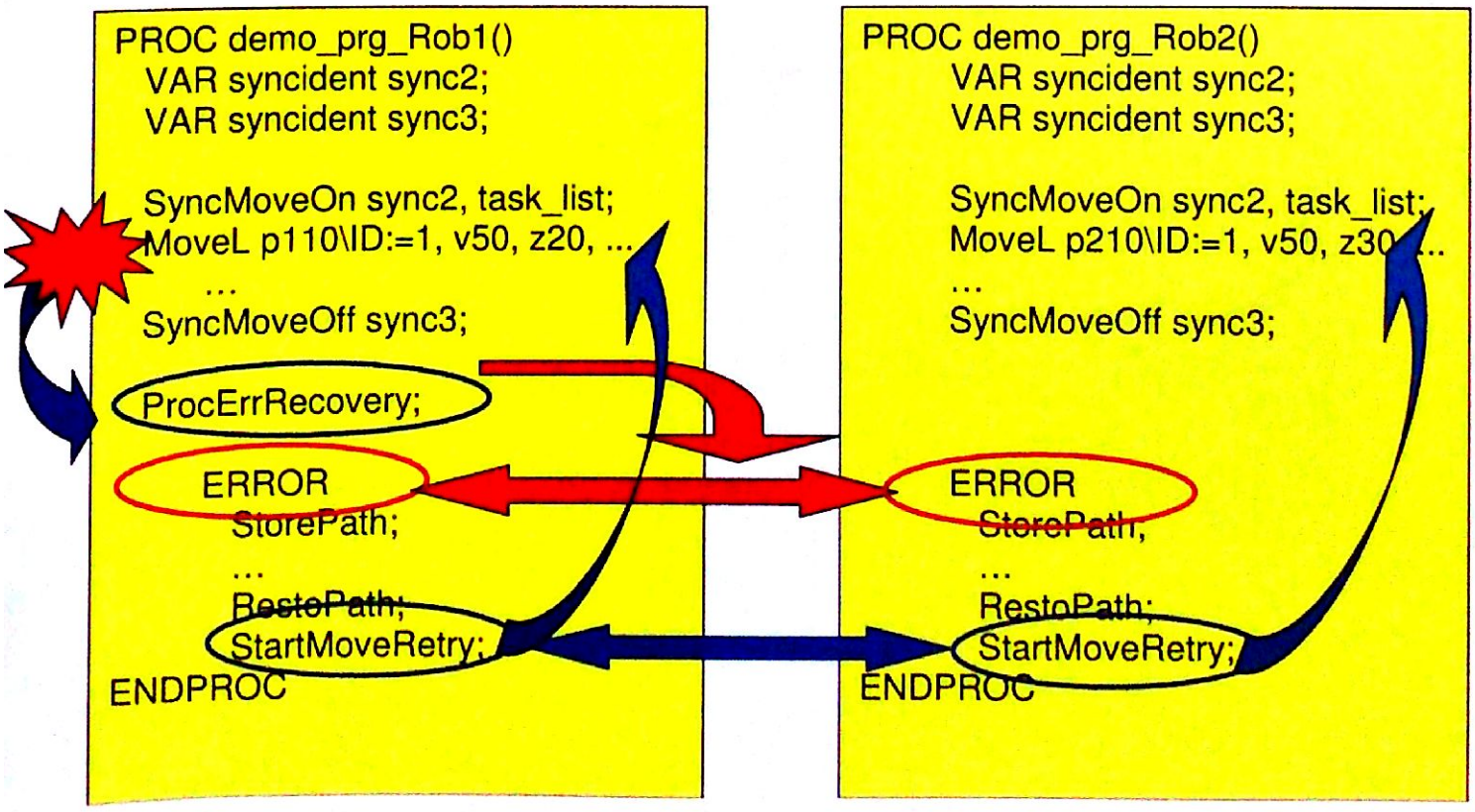


6, MultiMove hibakezelés

MultiMove Hibakezelés...Független mód



MultiMove Hibakezelés...Szinkronizált mód



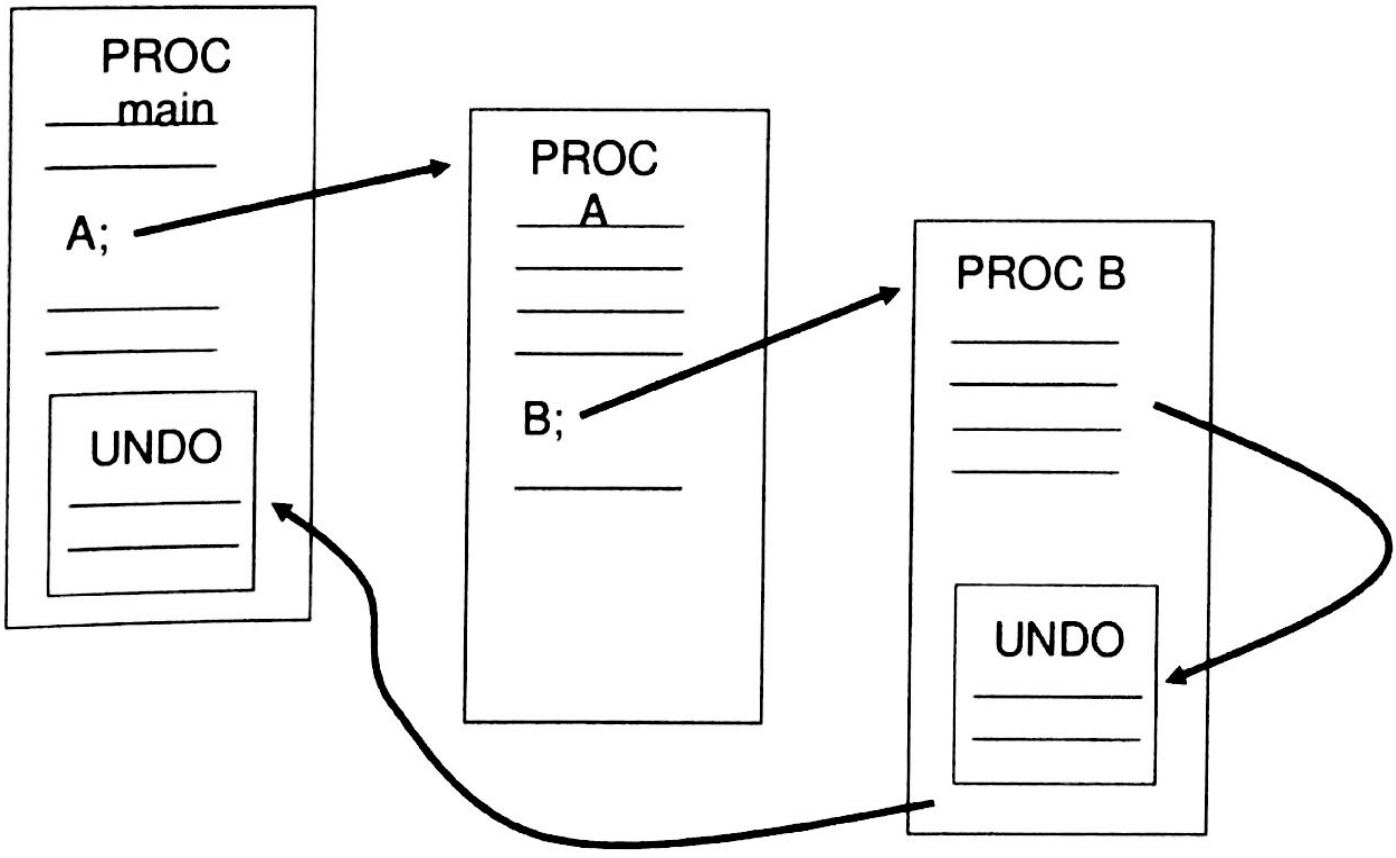
UNDO funkciók

- Az UNDO funkció a rutinok megtisztítására szolgál, mikor a programmutató kiugrik a rutinból.

- Hasonló módon készítjük el a rutinok számára, mint a Hibakezelőket (Error Handlers).

- UNDO használható a következő esetekben:
  - A rutinok által megnyitott fájlok bezárása.
  - IO jelek nullázása.
  - Meghívni a SyncMoveUndo utasítást ha a rutin tartalmaz szinkronizált mozgást

-UNDO-kezelőkben nem használható mozgás-utasítás.



6, Külső tengelyek

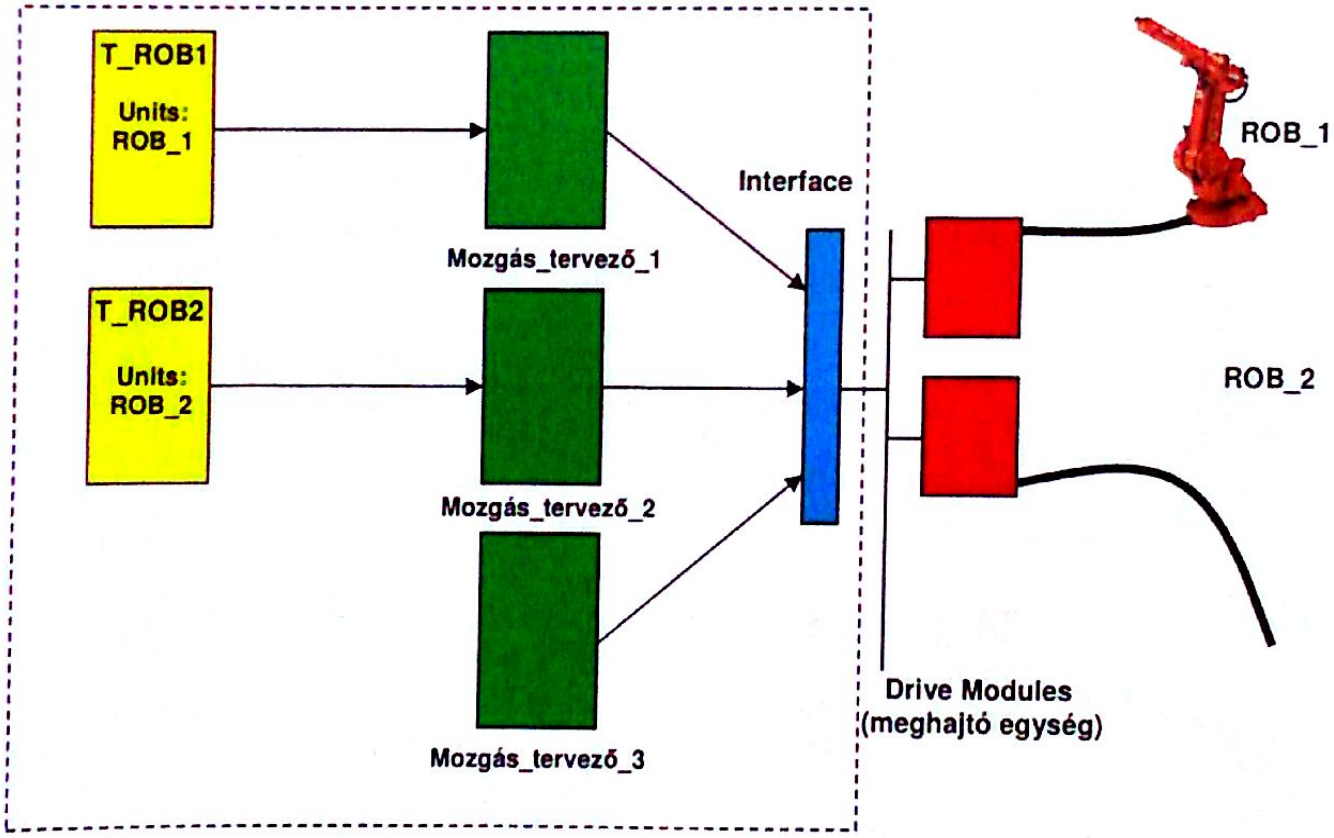
- MultiMove nélküli rendszerekben az opcionális külső tengelyeket ugyanúgy konfiguráljuk, mint korábban.

- MultiMove rendszerekben el kell dönteni, melyik task-ban definiáljuk.Lehetőség van egy már létező task-ban használni, vagy egy új task-ot létrehozni a számára.

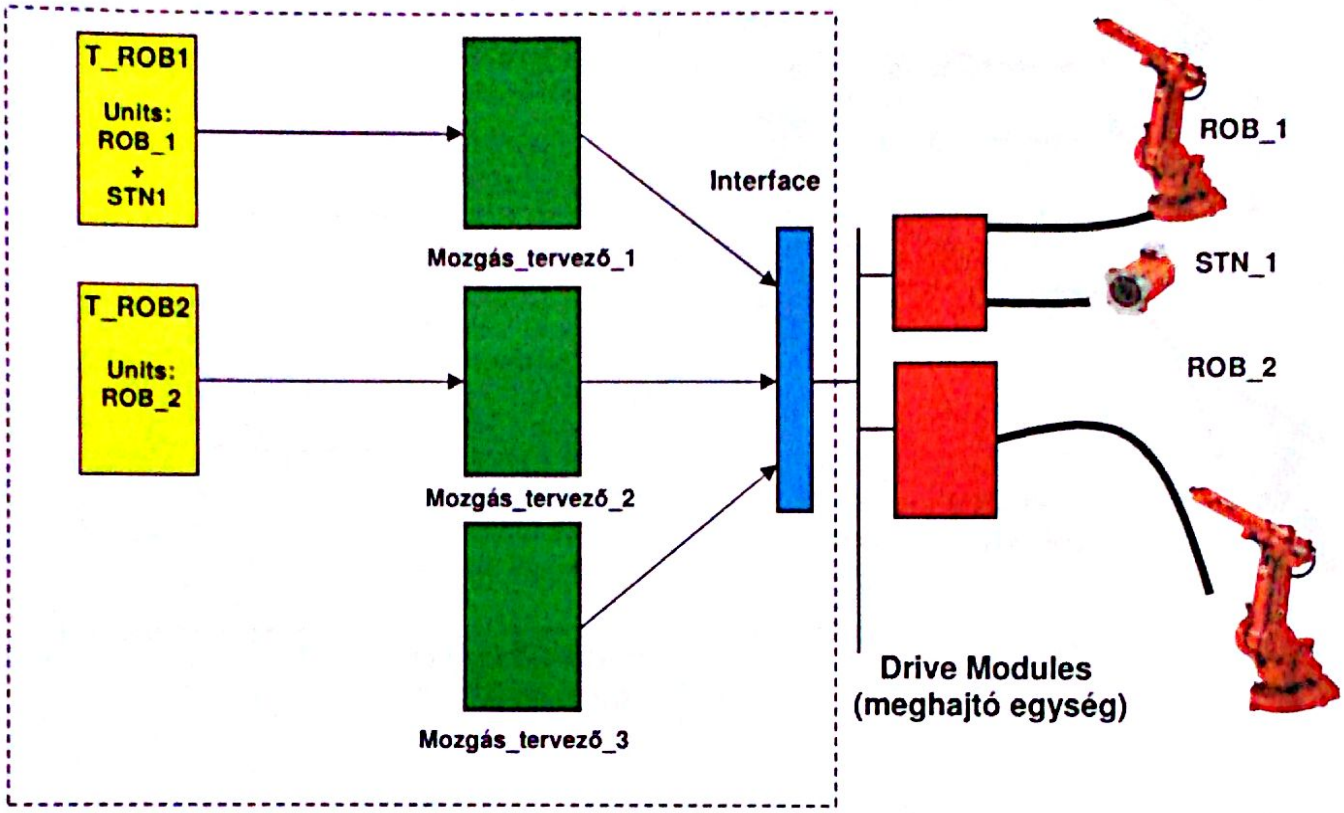
- Külső tengelyek Rapid-taskban robot nélküli (-TCP nélküli) mozgatására a MoveExtJ utasítást kell használni.

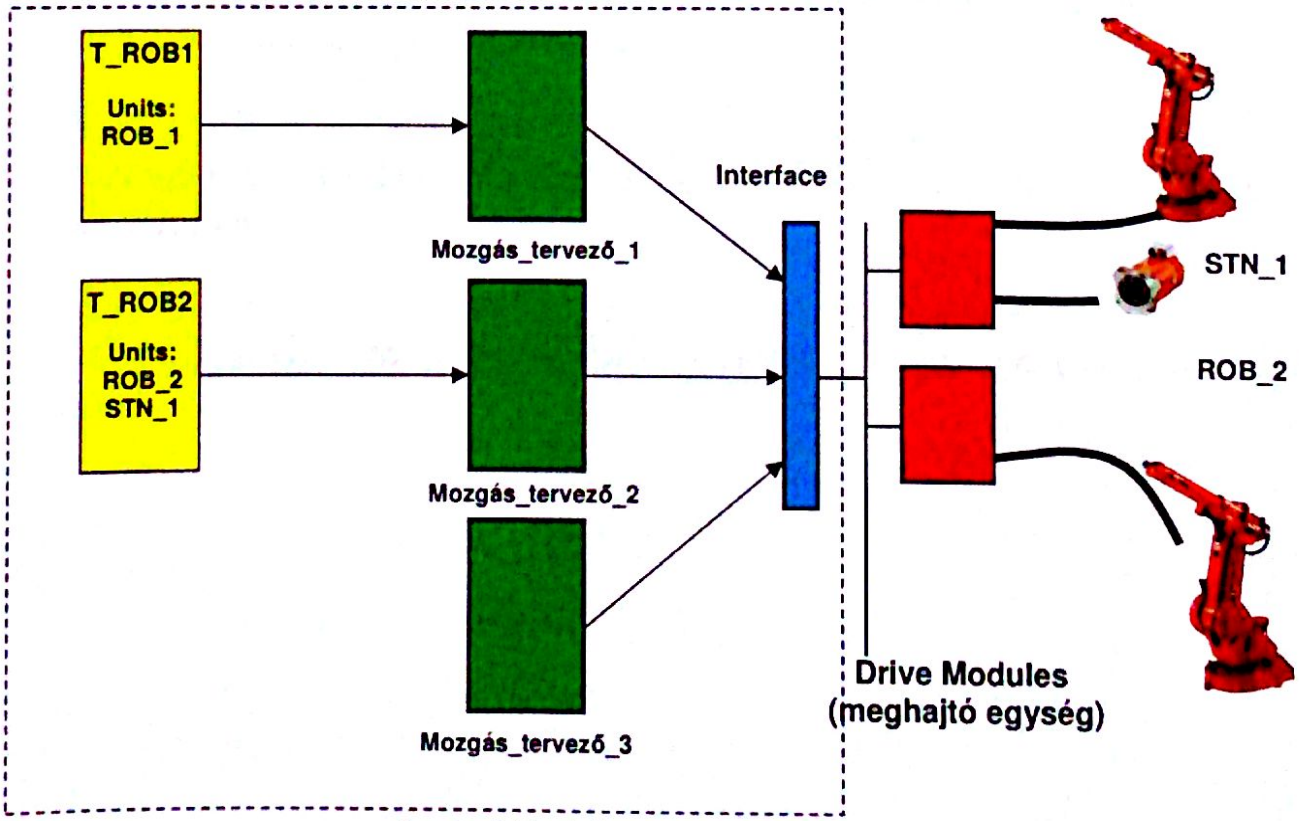
Vezérlési módok:

6.1, Általános vezérlés

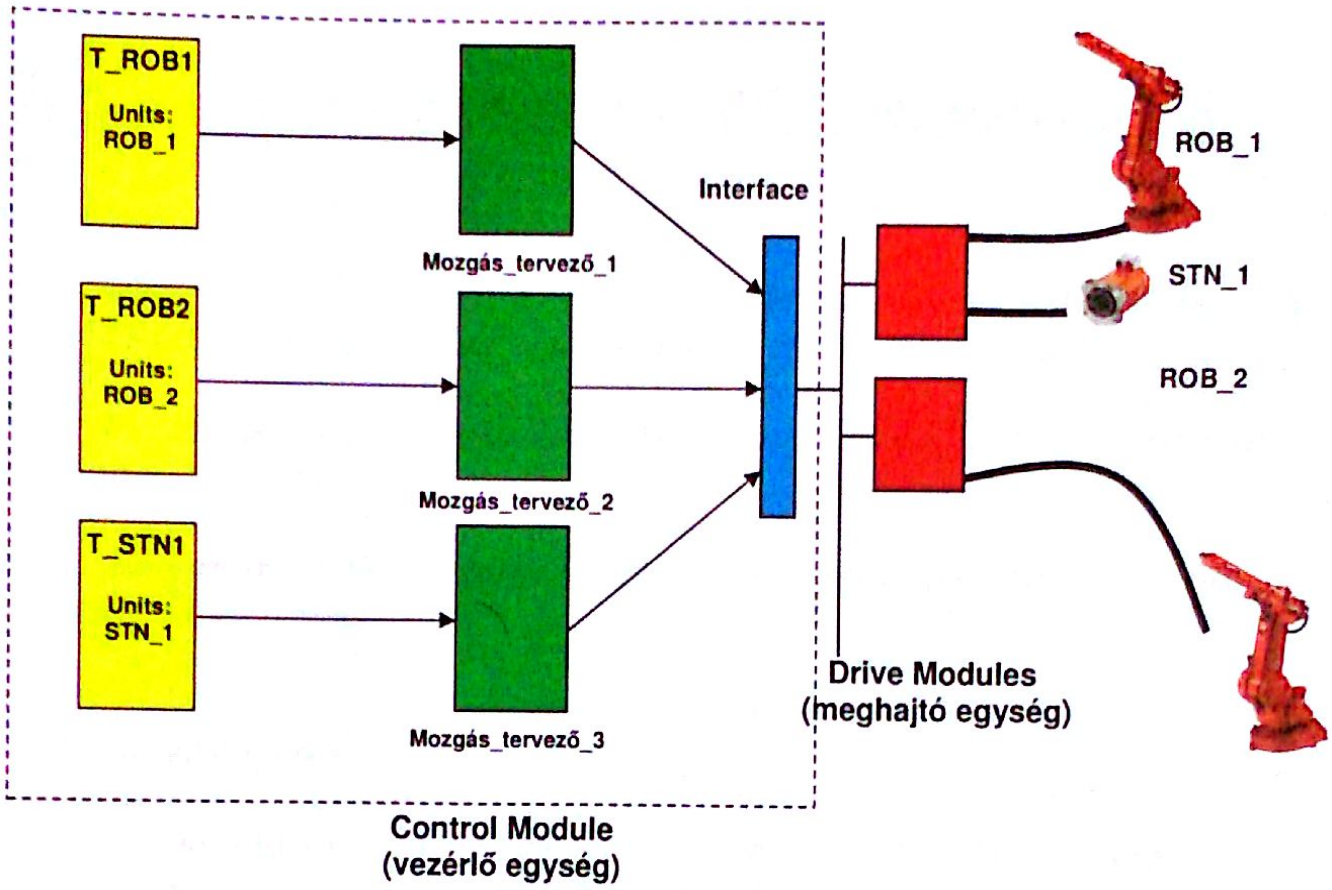


6.2, Robot-task által hordozott külső tengely vezérlés





6.3 Külön Task által hordozott külső tengely vezérlés



- A külső tengelyek fizikai kapcsolási módjai teljesen külön kezeltek és függetlenek a Rapid-konfigurációtól.

- A Rapid-task konfigurálás végső lépése és a rendszer felépítése a SYS.cfg-ben tárolódik.

- Az mozgás tervezők száma maximum 5 (RW5.05). RW5.06. verzió esetén 6 lehet.

7, A RAPID- programnyelv MoltiMove vonatkozásai

- A Rapid működése és a Rapid-programozás módja, illetve végrehajtása nem változott.Mindössze néhány új utasítást és adattípust adtunk hozzá.

- Mindegyik Rapid-program képes kezelni egy robotot TCP-vel, illetve több mint 6 külső tengelyt.

- Koordinált együttmozgás esetén a a Rapid programok egymással szinkronban futnak.

7.1 Új adattípusok:

- **syncident** : Szinkronizációs utasítások azonosítására szolgál (p.l.: start, stop szinrkronizáció) különböző Rapid programokban.

- **tasks** : azon Rapid-taskok listája, amelyek egymással szinkronban futnak.

- **identno** : Szinkronizált mozgásutasítások azonosítására szolgál különböző Rapid programokban

7.2 Új RAPID- utasítások

- **WaitSyncTask**. Rapid-programok szinkronizálására szolgál a program egy adott pontjában.

- **SyncMoveOn**. Szinkronizált mozgás indítására szolgál.

- **SyncMoveOff**. Szinkronizált mozgás megállítására szolgál.

- **MoveExtJ**. Ez az új mozgásutasítás arra szolgál, amikor csak a külső tengely (TCP nélkül) aktív a rapid-programban.

- **IsSyncMoveOn**. Beazonosítja, hogy a Rapid-program szinkron módban van-e vagy sem

- **SyncMoveUndo**. Szinkronizáció kikapcsolására szolgál.