

דוגמא 1:

```
Module M
{
  VAR
    u : 1..10;
    :
  ASSIGN
    :
    :
    next(u) := a.x & a.y;
    :
}
```

ב-SMV ייתכן מצב בו עבור מודול מסוים M מופיעים בביטויים (של define או assignment או lhs או ברשימת actual) משתנים לא מקומיים למשל נניח a.x ונניח גם a.y. (דוגמא 1)

ב-CDL יש צורך להעביר את הפרמטרים בצורה מפורשת. לשם כך עלינו לפרק את a לרכיביו.

ייתכנו שני מצבים:

1. a הוא מופע של מודול (נאמר A) בתוך M (דוגמא 2):

יש להעביר את ההצהרות על x ו-y מתוך A החוצה למודול האב (או אף גבוה יותר) במודול האב (במקרה שלנו M) הם יקראו a#x ו-a#y, כלומר שם המופע משורשר עם שם המשתנה. בנוסף, יש צורך שברשימת ה-formals של A יופיעו גם x ו-y. (דוגמא 2) כך, לכל מופע α של A נעביר כפרמטרים את $\alpha\#x$ ואת $\alpha\#y$ שיוצארו ברמה גבוהה יותר. יש לשים לב שיייתכן מצב בו עבור מופע מסוים a1 אין צורך ממשי ב-a1.x וב-a1.y אך מכיוון שעבור מופע אחר של A – a2 הייתה גישה אל a2.x ו-a2.y, הם נוספו לרשימת ה-formals של A ולכן גם עבור a1 וגם עבור a2 ההצהרה על x ו-y תועבר לרמה גבוהה יותר. (דוגמאות 3א, 3ב)

דוגמא 2:

```
Module M
{
  VAR
    u : 1..10;
    a#x : x_type ;
    a#y : y_type ;
    A a(p1,p2,...,pn ,a#x
,a#y);
    :
  ASSIGN
    :
    next(u) := a#x & a#y;
    :
}

Module A(f1,f2,...,fn ,x,y)
{
  :
  :
}
```

דוגמא 2:

```
Module M
{
  VAR
    u : 1..10;
    A a(p1,p2,...,pn);
    :
  ASSIGN
    :
    next(u) := a.x & a.y;
    :
}

Module A(f1,f2,...,fn)
{
  VAR
    x : x_type ;
    y : y_type ;
    :
    :
}
```

דוגמא 3:

```
Module main
// no use of a1.x, a1.y
{
  VAR
    a1#x : x_type;
    a1#y : y_type;
    A a1(p1,p2,...,pn,a1#x,a1#y);
    M m();
    :
  ASSIGN
    :
}

Module M
// uses a2.x and a2.y
{
  VAR
    u : 1..10;
    a2#x : x_type;
    a2#y : y_type;
    A a2(p1,p2,...,pn,a2#x,a2#y);
    :
  ASSIGN
    :
    next(u):=a2#x & a2#y;
    :
}

Module A(f1,f2,...,fn,x,y)
{
  :
  :
}
```

דוגמא 3:

```
Module main
// no use of a1.x, a1.y
{
  VAR
    A a1(p1,p2,...,pn);
    M m();
    :
  ASSIGN
    :
}

Module M
// uses a2.x and a2.y
{
  VAR
    u : 1..10;
    A a2(p1,p2,...,pn);
    :
  ASSIGN
    :
    next(u):=a2.x & a2.y;
    :
}

Module A(f1,f2,...,fn)
{
  VAR
    x : x_type;
    y : y_type;
    :
    :
}
```

עבור מופע a1 (במודול main) אין דרישה חיצונית ל-a1.x או ל-a1.y, למרות זאת העברנו את 'a1.x' ואת 'a1.y' בעקבות דרישה חיצונית עבור מופע a2 (במודול M) בהצבה
.next (u) = a2.x & a2.y

2. a הועבר ל-M כפרמטר (דוגמא 4):

דוגמא 4:

```
Module M(f1, f2, ..., a, ..., fn)
{
  VAR
  :
  ASSIGN
  :
  next (u) := a.x & a.y;
  :
}
```

במקרה זה, a הוא למעשה formal parameter. יש צורך שברשימת ה-formals של M יופיעו באופן מפורש משתנים שנמצאים ב-a ומשמשים ב-M לחישוב.

במקרה זה – a#x ו-a#y.

כמובן שכעת מכל מופע של M עבור הפרמטר האקטואלי המתאים ל-a, נאמר k, תידרש העברה של k.x ו-k.y.

בשלב ראשון, אוספים בכל מודול M את הדרישות לתת-פרמטרים. את הדרישות רושמים בהתאם למקרה, עבור תת פרמטר p:

אם מדובר ב-formal אזי מוסיפים לו ברשימת תת-פרמטר את p.

אם מדובר במשתנה i שהוא מופע של מודול T, מוסיפים לרשימת הבקשות מ-T את תת הפרמטר p.

אם מדובר במשתנה i שהוא אינו מופע של מודול, זו שגיאה סמנטית.

בשלב השני, יש לייצר הצהרות מתאימות במקומות המתאימים.