* כדי להריץ את הסימולציה יש לכתוב את הפקודה הבאה:

(הקובץ הוא הקובץ שנוצר ע"י ה-TLV)

* כאשר הפלט יהיה קובץ בשם (שהוגדר ע"י המשתמש)

הרצת ה- TLV

TLV-BASIC

* בשפת TLV-B כל הקבצים הם בעלי סיומת

על מנת להריץ את הקובץ count.tlv יש להקליד בשורת הפקודה:

% tlv count.tlv

(הקובץ count.tlv)

(הקובץ count.fds)

(הקובץ count.sms)

(הקובץ count.spl)

(הקובץ count.log)

* כאשר מקלידים את הפקודה tlv ולאחר מכן מופיע הסימן "<<"

<< simulate 100;

מורה לסימולטור להריץ את הסימולציה למשך 100 מחזורי שעון.

- כאשר הסימולטור יגיע למחזור שעון 101, הוא יעצור את הסימולציה

show_all;

* אין תמיכה ב CTL

* אין תמיכה ב ROBDD: כל הביטויים הם

- שפת הקלט: SMV. יש ב NuSMV תמיכה ב ROBDD

קובץ הקלט: junk.spl (או junk.smv)

הרצה: NuSMV -i junk.smv

Let a := x /\ y;

>>

a :: bool : הדפסת סוג המשתנה

Let dynamic_var := expression;

* ① - expression ... אפשרי - מחרוזת בפני עצמה:

(לפי סדר עדיפות)

<-> , -> , ! , V , & , ^

כל אחד מאופרטורים יכול לפעול על integer ועל boolean

אופרטור / , * , - , + לוגיים מחזירים =< , => , <> , =!

- אופרטורים על expression יכולים להיות מחרוזת

מחרוזת מוקפת ב" ומסתיימת ב- ;

Print expr;

expr יכול להיות: ① string, ואז הוא יודפס

>> Print "to print";
to print
>>

② expr יכול להיות משתנה, ואז יודפס את ה-RoBDD

שים לב! אם המשתנה לא מוגדר - true. אבל אם הוא מוגדר

פקודה

```
from expr;
```

כאשר expr הוא ביטוי המייצג פונקציה בוליאנית. הפקודה מחזירה את ה-BDD שמייצג את הפונקציה (כמו שראינו בהרצאה).

---

simulate - הפקודה

הפקודה הבאה מקבלת ביטוי בוליאני exp ומספר שלם step_no, ומחזירה את הערך של exp לאחר step_no צעדים.

```
eval exp, step_no;
```

דוגמה

```
>> eval x&x, 1;
```

מחזיר את הערך של x&x לאחר צעד אחד. כלומר, הערך של x בצעד 1.

```
chk_inv cut_array, expression_array, n;
```

- trunc n;

- step e;

- cont n;

- start e;

(23)

show n ;

[redacted]

show_all ;

last ;

(Any-key) کسی بھی کلید کو دبائیں

* פקודה זו מחזירה את ערך המשתנה

* שליטה על מהירות הסימולציה

לפעמים נרצה להאט את הסימולציה כדי שנוכל לראות את התנועה של הרובוטים בצורה טובה יותר. לשם כך נשתמש בפקודה הבאה:

```
setstep n,e;
```

כאשר n הוא מספר שניות ו-e הוא מספר הצעדים שיתבצעו בכל n שניות.

```
appstep -e;
```

מאפשר לנו לבצע צעדים לאחור

```
setvar n,V,e;
```

מאפשר לנו לשנות את ערך המשתנה n ל-V בכל e צעדים.

```
setloop n;
```

על מנת לשנות את מספר הלולאות שיתבצעו ל n

שנה את מספר הלולאות ל-5

```
setloop 5;
```

קובץ ה־ pile.spl וקובץ ה־ pile.smv

קובץ ה־ emptyx.spl וקובץ ה־ emptyx.smv

```
fsimtl ltl (temporal_formula);
```

```
sf;(e)ltl.ltwis
```

valid ltl (temporal_formula);

(שקף מספר 84-85)

(2)

* אין צורך לבדוק ב- yp שאינו
(למשל כאשר הוא מוגדר כ- register)

אחרת לדוגמה אם ביל"ב לא קיים
על ה- exp_dvj ונקבל הודעה INVALID-
(שגיאה)
אין ל"ם הגדרה מוסכמת
(כמעט) כל הפעולות שנעשות על
(כמעט) כל הפעולות שנעשות על exp_dvj תהיינה
(שגיאה)

(לכל הפחות אם)