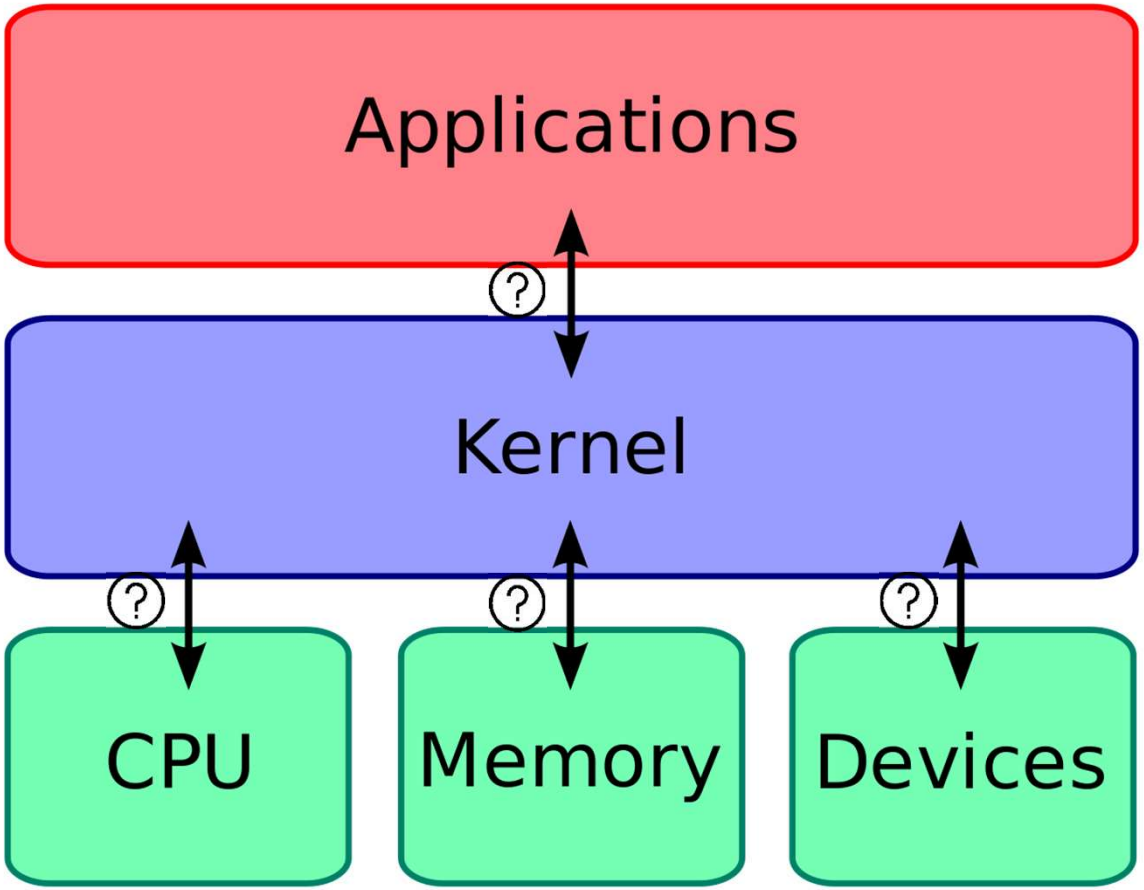
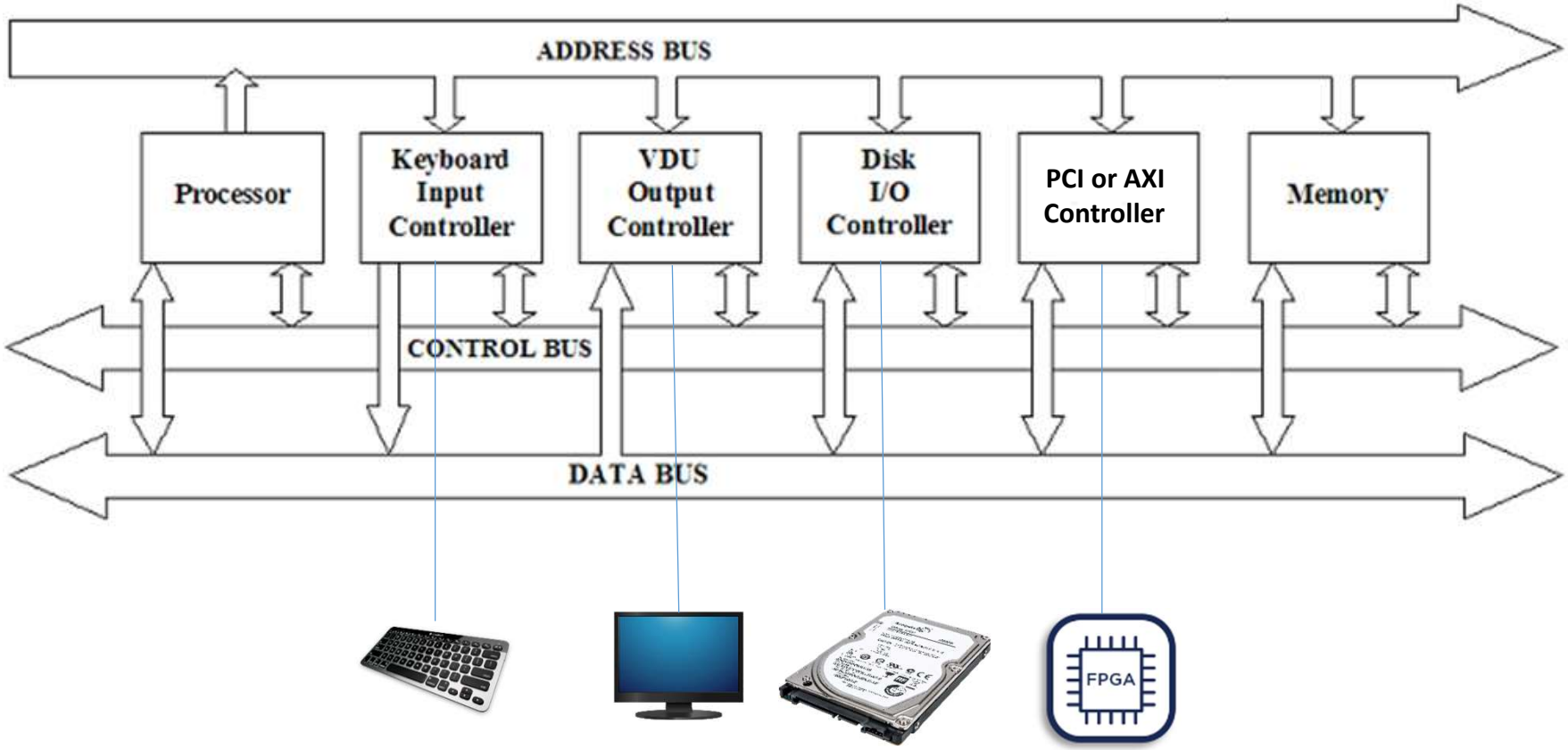
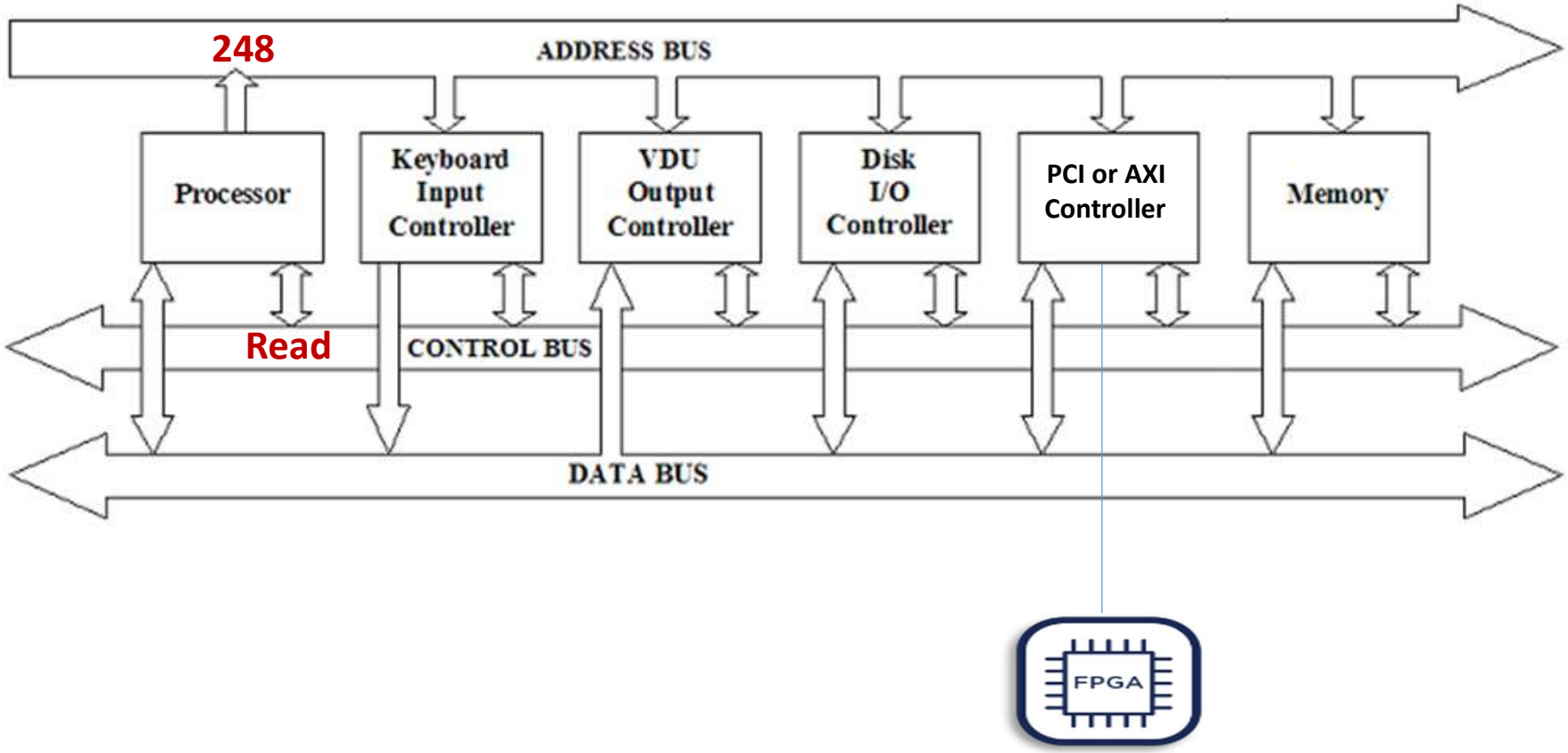


בלמ"ס

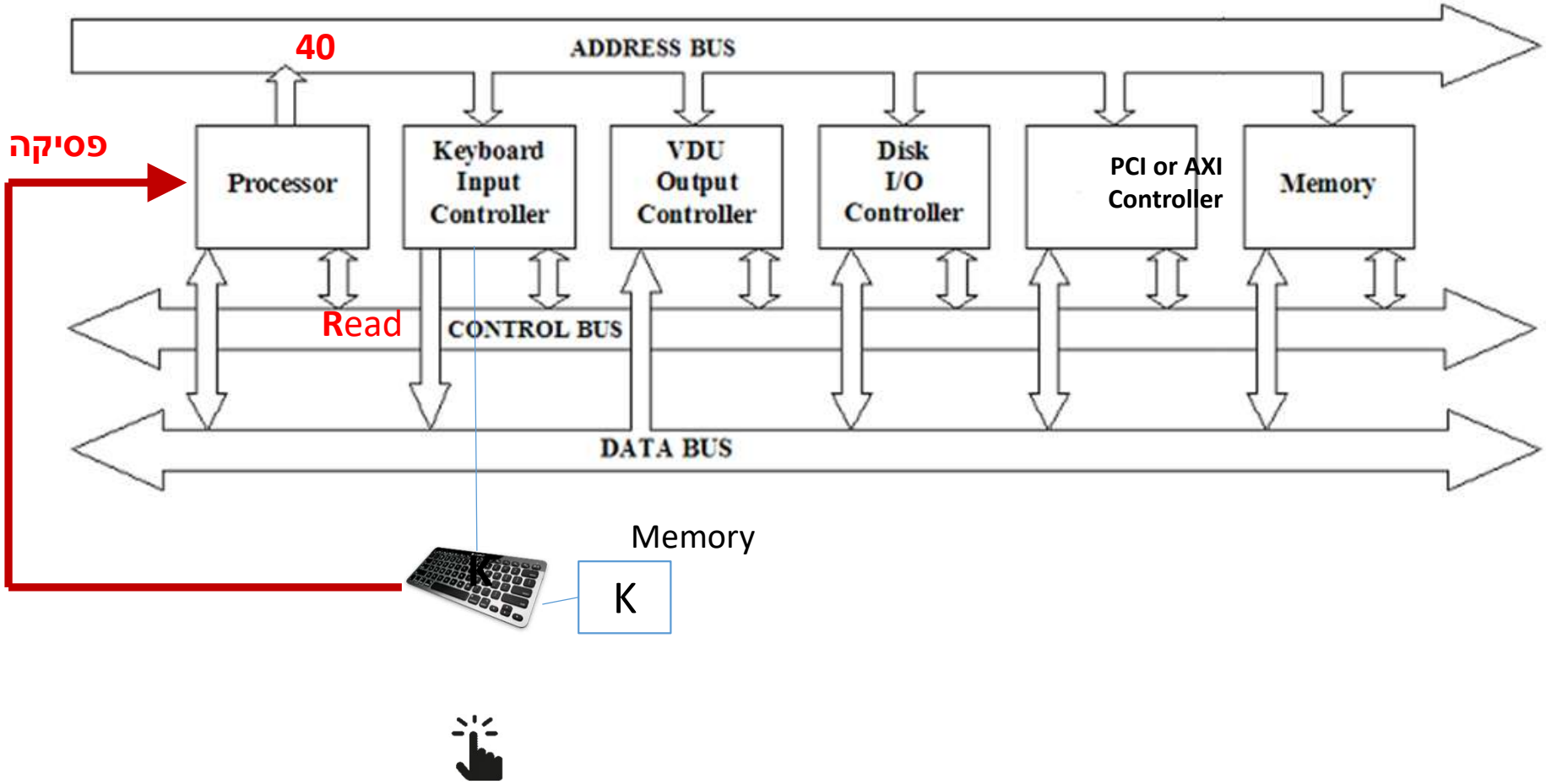
מנגנון הפעולה ממבט של תוכנה



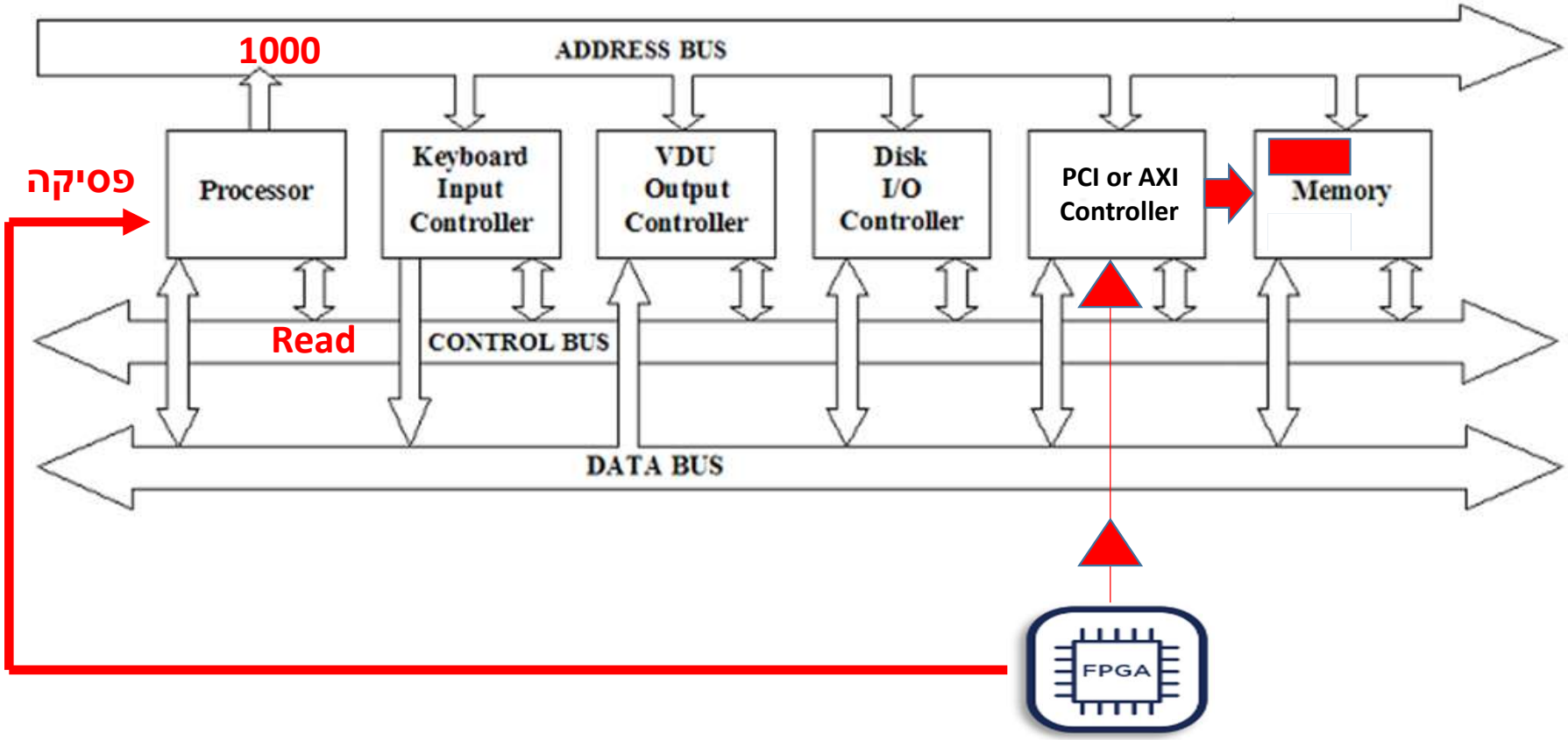
קריאה/כתיבה סינכרונית



קריאה/כתיבה אסינכרונית



קריאה/כתיבה אסינכרונית בשילוב DMA



איך כותבים קוד שפונה לחומרה?

- משימה: כתבו קוד אשר קורא תו אחד (8 ביט) מערוץ ה- uart.

UART Module Address Map

Registers in the UART module

Module Instance	Base Address
uart0	0xFFC02000
uart1	0xFFC03000

```
#define ALTERA_UART_SIZE        32

#define ALTERA_UART_RXDATA_REG      0
#define ALTERA_UART_TXDATA_REG      4
#define ALTERA_UART_STATUS_REG      8
#define ALTERA_UART_CONTROL_REG     12
#define ALTERA_UART_DIVISOR_REG     16
#define ALTERA_UART_EOP_REG     20
```

גישת המעבד(CPU) להתקני קלט/פלט

- ישנן שתי דרכים מקובלות לגישת מעבד להתקני פלט/קלט:
  1. I/O Ports – קיים במעבדי אינטל לטובת תמיכה לאחור
  2. **Memory-Mapped I/O (MMIO) – שיטה הקיימת בארכיטקטורות ARM, אינטל שהן הארכיטקטורות הנפוצות.**

Memory-mapped I/O (MMIO):

- בשיטה זו מרחב הכתובות מחולק בין הזיכרון (RAM) ובין כל התקני ה-I/O (FPGA, Hard Disk, Graphic Card, Keyboard…).
- כל התקן מקבל תחום או מספר תחומי כתובות הממופים אליו.
- פנייה של המעבד לכתובת X המשוייכת/ממופה להתקן Y תגרום לגישת כתיבה/קריאה להתקן Y
- מוגבלת למרחב הכתובות שמגדירה הארכיטקטורה:
  32bit = 4G addresses
- המעבד פונה לכתובות אלה בדיוק כפי שפונה לזיכרון.
- פקודות מכונה של פנייה לזיכרון – mov dest, src או load/store
- ה- bus שדרכו המעבד פונה לזיכרון משמש גם לפניות להתקני ה-I/O

נסו לפתור את המשימה

UART Module Address Map

Registers in the UART module

Module Instance	Base Address
uart0	0xFFC02000
uart1	0xFFC03000

הצעה לפתרון:

```
unsigned int *baseAddress = 0xFFC02000;

unsigned char rx_val = *(unsigned char *)baseAddress;
```

מציג את חלוקת הכתובות (פיסיות) של ההתקנים – # cat /proc/iomem

המשך הניסיון

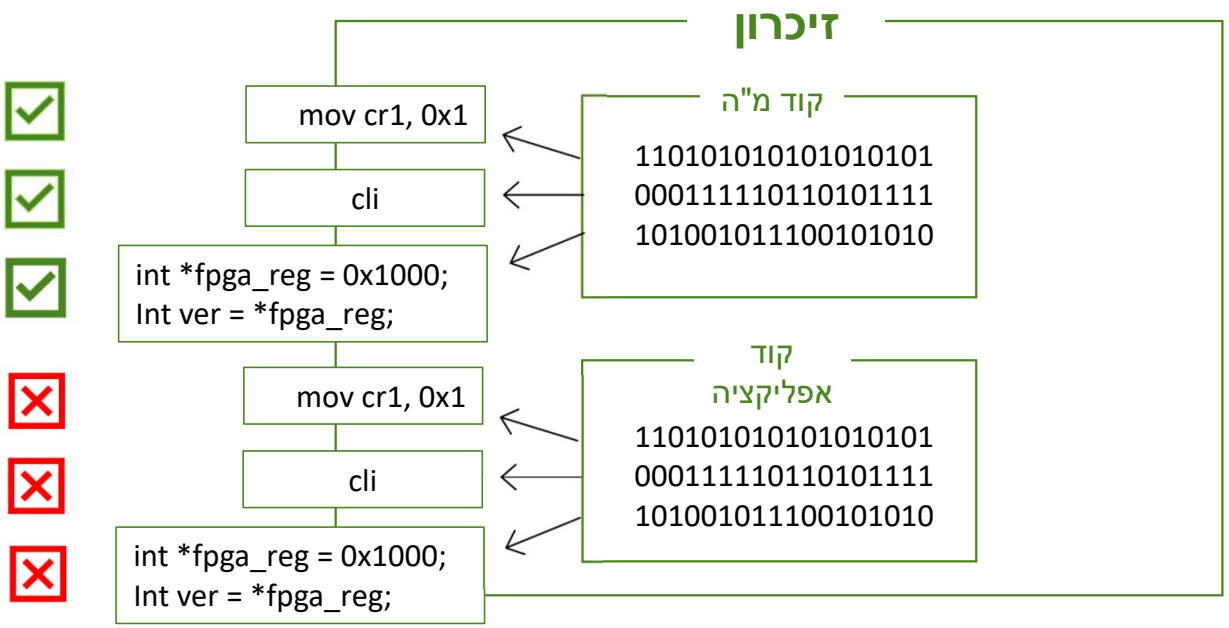
- האם הפתרון יעבוד???
- לא בהכרח! תלוי בהרשאות שניתנות לקוד שרץ.
  נרחיב על כך בהמשך.

Tux - linux mascot

פריבילגיות של מערכת ההפעלה

למערכת ההפעלה יש שלוש פריבילגיות שלאפליקציה אין:

1. גישה לרגיסטרים של קינפוג החומרה.
2. הרצת פקודות מכונה מסויימות כגון: cli, sti
3. שליטה בטבלאות המרת הכתובות ← שליטה בכתובות שהאפליקציה יכולה לגשת אליהן.



איך דווקא הקוד של מערכת ההפעלה מקבל פריבילגיות גבוהות???

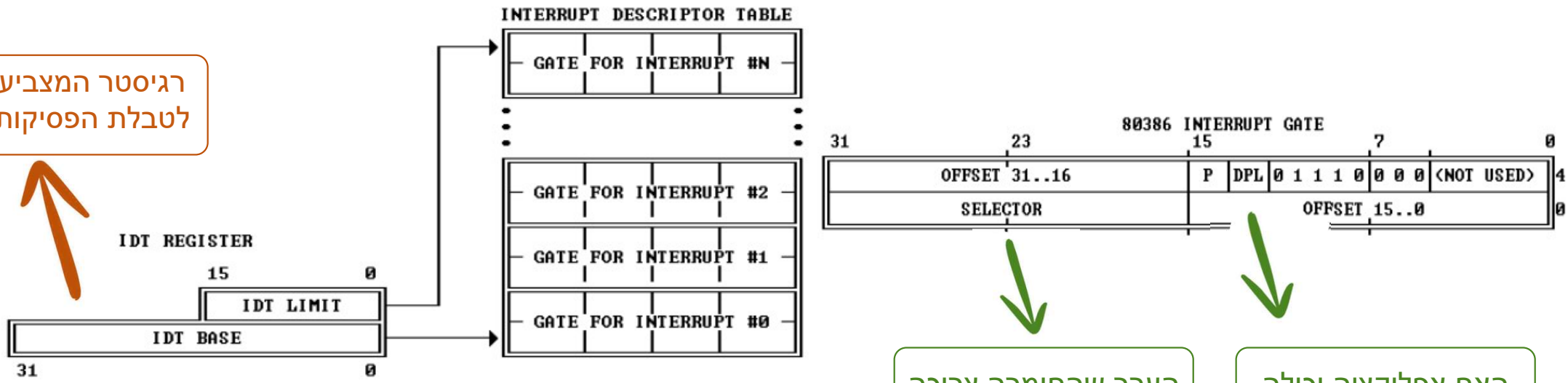
תשובה: רגיסטר CS

טבלת הפסיקות

הדרך **היחידה** דרכה ניתן להריץ קוד של מערכת ההפעלה לינוקס כשאפליקציות רצות

מ"ה מקצה זיכרון וממלאת
את טבלת הפסיקות

Figure 9-1. IDT Register and Table



רגיסטר המצביע
לטבלת הפסיקות

הערך שהחומרה צריכה
לכתוב לרגיסטר CS

האם אפליקציה יכולה
ליזום את הפסיקה