203970736,

**דוקומנטציה**

**אופן פעולת התוכנית**

ראשית, התוכנית קוראת את הקובץ parameters.txt ומכניסה את כל הנתונים למבנה הנתונים הבא (נעשה בפונקציה SetCfgParameters):

typedef struct \_CfgParameters {

int add\_nr\_units, mul\_nr\_units, div\_nr\_units;

int add\_nr\_reservation, mul\_nr\_reservation, div\_nr\_reservation;

int add\_delay, mul\_delay, div\_delay;

int mem\_delay;

int mem\_nr\_load\_buffers, mem\_nr\_store\_buffers;

} CfgParameters;

לאחר מכן, אנו בונים את כל מבני הנתונים שיהיו התחנות באמצעות הפונקיצה PrepareReservationStations:

typedef struct \_Station {

bool is\_busy;

char \*command;

float v\_j;

float v\_k;

int q\_j;

int q\_j\_station\_offset;

int q\_k;

int q\_k\_station\_offset;

int addr;

bool is\_load;

bool is\_store;

bool is\_add;

bool is\_sub;

bool is\_mult;

bool is\_div;

bool is\_halt;

int cycle\_entered;

bool is\_ready\_for\_exec;

bool is\_in\_exec;

int cycle\_to\_finish\_exec;

int original\_inst;

int issue\_num;

int PC;

int imm;

}Station;

לאחר מכן אנו קוראים את הקובץ memin ומכנסים את כל הזכרון למשתנה int MEM[MEM\_SIZE]

כעת אנו רצים על הלולאה הראשית של התוכנית עד אשר כל הפקודות סיימו (השדה is\_busy שווה ל-false בכל התחנות הקיימות).

**בתוך הלולאה אנו מבצעים** **Fetch**  לשתי פקודות מתוך הזכרון (MEM). כל הפקודות שעושים להם Fetch נכנסות לתוך מערך שהוא מבנה הנתונים הבא:

typedef struct \_InstQueue {

bool is\_busy;

int op;

int dst;

int src0;

int src1;

int imm;

int original\_inst;

} InstQueue;

**כעת, אנו מבצעים פעמיים issue באופן הבא:**  
מפעילים את הפונ' issue שלוקחים את הפקודה הראשונה מתוך ה- inst\_queue.   
הפונ' מזהה את סוג ההודעה (ADD/SUB, DIV, MULT, LD, ST) ולפי סוגה בוחנת האם יש מקום פנות בתחנה המתאימה. אם אין, היא לא מבצעת issue. חיפוש מקום פנוי נעשה ע"י הפונ' SearchFirstNonBusy, וההכנסה עצמה לתחנה נעשה ע"י הפונקציה EnterToReservationStation. לאחר מכן מעדכנים את קובץ האוגרים באמצעות הפונקציה UpdateRegisters. להלן מבנה הנתונים של קובץ האוגרים:

typedef struct \_Registers {

char \*reg\_name;

float V;

int Q; // ADD\_SUB\_RESORVATION\_STATION 1, MUL\_RESORVATION\_STATION 2, DIV\_RESORVATION\_STATION 3

int station\_offset;

} Registers;

כעת, אנו מבצעים POP לפקודה הזו מה-inst\_queue באמצעות קריאה לפונקציה PopInstQueue.

**בשלב זה אנו מבצעים Execution באופן הבא:**בתוך הפונ' Exec שמבצעת את תהליך ה-Execution, אנו בודקים, עבור כל סוג הוראה את כמות היחידות הפונקציונאליות הזמינות ב-cycle הנוכחי. זה מתבצע בפונ' GetNumberOfWorkingExecUnits.לאחר מכן, אנו עוברים על כל ההוראות שבתחנות ומגדירים את השדה is\_ready\_for\_exec שבמערך התחנות. פקודה שמוכנה להיכנס ליחידה הפונקציונאלית מזוהה כך:1. התחנה מוגדרת כתפוסה (השדה is\_busy שווה ל-true), כלומר, התחנה אינה מכילה "זבל".  
2. אם ה-cycle הנוכחי גדול מה-cycle שבו ההוראה נכנסה לתחנה (שדה cycle\_entered מכיל את ה-cycle שבו ההוראה נכנסה לתחנה המתאימה.), וגם:  
3. הפקודה לא נמצאת כבר ביחידה הפונקציונאלית (כלומר, השדה is\_in\_exec שווה ל- false), וגם:  
4. השדות q\_jו- q\_k שווים ל-0 (זה סימן לכך שאין TAG ושהפקודה לא מחכה שפקודה אחרת תסתיים).

כל ההוראות שהתנאים ההלו מתקיימים בהן, מגדירים עבורן את השדה is\_ready\_for\_execל- true.לאחר מכן מפעילים את הפונ' EnterToExec שעוברת על כל התחנות ומחפשת את **ההוראה הראשונה** המוכנה להיכנס ליחידה הפונקציונאלית (במידה והיא פנויה). ואז נגדיר שהשדה is\_in\_exec שווה ל-true ונגדיר גם את השדה cycle\_execute\_end שיכיל את ה-cycle שהפקודה אמורה לסיים את תהליך ה-Execution לפי הנתונים בקובץ הפרמטרים.

**שלב הבא אנו מבצעים כתיבה ל-CDB:**  
עוברים על כל התחנות, ועבור כל הוראה בודקים האם ה-cycle הנוכחי גדול מ- cycle\_execute\_end שלה, ושהשדה is\_in\_exec שלה שווה ל-true. במידה וכן, זה סימן לכך שהפקודה סיימה את ה-Execution וזה הזמן להעבירה אל ה-CDB המתאים.  
בשלב זה אנו מבצעים את הפעולות החשבוניות או את הגישה למערך MEM כדי לקרוא את הנתונים ממנו או לכתוב אליו.  
אם ההוראה שמוכנה להיכתב ל-CDB היא LD/ST אז מפעילים את הפונ' DoMemCalc שקוראת\כותבת מ\לזיכרון. הדבר נעשה תוך המרה מ\לפרומט single precision.  
לאחר מכן, מפעילים את הפונ' RemoveLabel שעוברת על כל התחנות ועל קובץ האוגרים ומסירה את ה-TAG ושמה את הערך שנכתב על ה-CDB בכל המקומות שתלויים בהוראה שזה הרגע נכתבה ל-CDB. התחנה מוחקת את ההוראה שנכתבה כדי לפנות מקום להוראה חדשה.

**קבצי בדיקות**

בדיקה ראשונה:

**בדיקה שנייה:**

**בדיקה שלישית:**

בזיכרון קיימים הנתונים הבאים:

MEM[F] = 2.5

MEM[10] = -3.4

תרגום הפקודות של בדיקה השלישית להוראות הוא:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| OP | DST | SRC0 | SRC1 | IMM | מה קורה |
| ST |  |  | F8 | 00F | **MEM[F]=8** |
| LD | F7 |  |  | 00F | **F11 = 8** |
| ST |  |  | F6 | 00F | **MEM[F]=6** |
| ADD | F8 | F8 | F1 | 000 | F8=8+1 |
| ADD | F8 | F8 | F1 | 000 | F8=9+1 |
| LD | F7 |  |  | 010 | **F7=MEM[10]=-3.4** |
| MULT | F8 | F8 | F7 | 000 | F8=(8+2)\*(-3.4) |
| DIV | F8 | F8 | F2 | 000 | **F8=(8+2)\*(-3.4)/2=-17** |

(הערכים המודגשים הם הערכים העדכניים של האוגרים\הזיכרון)