Übungsblatt 2

Baran Güner, bg160 Tobias Hangel, th151

November 2022

Aufgabe 1

a) **ACCLd** und **IRd** werden freigegeben, damit sie im ALU verrechnet werden können.
ALUAd damit das Ergebnis an den Adressbus kann und im Speicher die richtige Zelle adressiert wird.
SPDd damit der Wert des SP-Registers an den Datenbus gelangt und im Speicher an der ACC+i Adresse gespeichert wird.

-0.25 IN2Dd

b) **ACCDd** damit der Akkumulator in den Datenbus geladen wird.
DDId damit das IN1-Register erreicht werden kann.

c) **ACCDd** und **DRd** damit der Inhalt des Akkumulators erst an den Datenbus und dann in den rechten Pfad der ALU gelangt.
IN1Ld damit der Inhalt des IN1-Registers in den linken Pfad der ALU gelangt.
Das Ergebnis gelangt über **ALUDId** in das ACC-Register.

d) **STORE S i**, da das i in der ALU verrechnet wird, das Ergebnis aber ebenfalls i sein muss. Folglich liegt am linken Operanden-Bus eine 0 an.

Aufgabe 2

-0.25 auf dem Stack muss Platz geschafft werden
-0.25 es gibt keinen Befehl Store mit zwei Registern, es gibt nur Store Reg i oder Storein SP Acc i

a) **push():** STORE ACC SP
pop(): ADDI SP 1
LOAD ACC SP

-0.25 es gibt keinen Befehl Load mit zwei Registern, es gibt nur

wenn nach dem ADDI SP 1 ein Hardwareinterrupt kommt, kann es zu Problemem kommen, mehr dazu im Tutorat

b) Der Benutzer hat keinen vollen Zugriff auf den Befehlssatz, so kann PC zum Bespiel nicht manuell verändert werden.
Dies dient dazu, dass der Benutzer keine potentiell problematischen Anweisungen erteilen kann.

wir haben für die RETI sowas wie einen Usermode bisher nicht eingeführt, aber das ist die beste Antwort bisher

Da die Aufgabe seltsam gestellt ist, gibt es hierfür trotz allem volle Punkte.

Aufgabe 3

fopen() öffnet eine Datei am gegebenen Dateipfad und assoziiert einen stream mit ihr. Dafür wird auf Systemebene openat() verwendet, was einen Eintrag in der systemweit zugreifbaren Tabelle offener Dateien mit Informationen zu Datei und Status erstellt.
mit fprintf(fptr,"%d ",i); werden alle Integer von 0 bis 2499 in den durch fptr gegebenen zuvor erstellten Stream geschrieben. Auf Systemebene wird dabei erst fstat() ausgeführt, was Informationen zum gegebenen stream sammelt und dann drei mal write(3)(die 3 ist dabei die ID der file description in der Tabelle) um die Zahlen(0-2499) alle in den Stream zu schreiben. Dabei ist mehr als eine Ausführung nötig, da ein einzelnes write()(abhängig von der Implementierung, in diesem Fall mit Linux) nur eine beschränkte Anzahl bytes schreiben kann.
Anschließend wird die Datei mit fclose() geschlossen, was auf Systemebene über close() den stream schließt.
Dann folgt ein erneutes fopen(), was auf Systemebene erneut als openat() interpretiert wird, dieses mal jedoch mit anderen flags, da die Datei nur gelesen und nicht beschrieben werden soll.
Mit fscanf() wird nun in fptr jedes Element das dem gegebenen Format "%d ëntspricht gezählt und die Anzahl in value vermerkt.
Da der stream dank dem Schließen und Öffnen wieder ßurückgesetzt"wurde erfolgt dies wieder vom Anfang der Datei, die Anzahl der gezählten Elemente beträgt 2500.
Schließlich wird mit fclose() erneut der stream geschlossen und mit return(exit_group() auf Systemebene) das Programm terminiert.