Diesmal sieht die Korrektur etwas anders aus als sonst. Ich hab den RETI-Code aller Studenten mithilfe des im PicoC-Compilers https://github.com/matthejue/PicoC-Compiler/releases eingebauten RETI-Interpreters ausgeführt, genauer mittels des Befehls `picoc_compiler -b -p c.reti -S -P 2 -D 15`. Ich habe versucht den Code von euch Studenten lauffähig zu machen, sodass dieser die Aufgabenstellung erfüllt. Alle Korrekturanmerkungen sind in der `c.reti`-Datei als Kommentare zu finden. Die Dateien `c.uart_r` und `c.uart_s` sind zur Simualation einer UART da und stehen für das Empfangs- und Statusregister und die darin enthalten Zahlen werden sobald auf die entsprechendedn Register zugegriffen wird gepopt.
Eure Korrektur ist unter https://github.com/matthejue/Abgaben_Blatt_3/tree/main/Blatt3/aepfel zu finden.

13/14

Betriebssysteme

Übungsblatt 3

Micha Erkel Felix Ruh

16/20

Aufgabe 1

a) Programm zum abholen der Daten:

LOADI IN1 0	IN1 auf 0 setzen (hier kann spaeter Inhalt aus R1 addiert werden).
LOADI DS 0	Zugriff auf Daten im EPROM
LOAD DS r	Konstante 010...0 in DS laden –> Zugriff auf UART
LOAD ACC 2	Statusregister R2 in Akkumulator laden
ANDI ACC 2	Bitweiser Vergleich von R2 mit 2 (00...10) -> falls b1 = 1 dann wird ACC = 2
JUMP== -2	Wenn ACC = 0, wird die Schleife von vorne gestartet
ADD IN1 1	Addiere den Inhalt aus R1 zum Inhalt im Register IN1
LOAD ACC 2	Lade erneut R2 in ACC
SUBI ACC 2	Setze b1 von R2 auf 0
STORE ACC 2	Update R2 im UART

b) Das Programm zum abspeichern der Daten im IN1:

LOADI IN2 4	Benutze IN2 als Schleifenzaehler
recall code	Füge Code aus Teil a) ein.
MULI IN1 256	Multipliziere den Inhalt aus IN1 mit 2^8 um 8 Bits nach links zu shiften
SUBI IN2 1	Verringer den Schleifenzähler um 1
LOAD ACC IN2	Lade den Schleifenzähler in den ACC
JUMP> -10	Vergleiche den Schleifenzähler, falls dieser nicht = 0, wiederhole die Schleife - springe auch durch a)

c) Das Programm zum abspeichern der Daten im SRAM:

LOADI SP a	Lass den Stack-Pointer auf a zeigen
recall code	Füge Code aus Teil b) ein - incl. Code aus a)
LOADI DS 0	Zugriff auf Daten im EPROM
LOAD ACC t	Lade die Konstante t (Codierung für "LOADI PC 0") in den ACC
LOAD DS s	Konstante 10...0 in DS laden –> Zugriff auf SRAM
STORE IN1 SP	Speicher die Eingabe aus IN1 im Stack des SRAM, an der Stelle SP, ab
ADDI SP 1	Lass den Stack-Pointer auf die nächste Speicherzelle zeigen
SUB ACC IN1	Subtrahiere von t im ACC den Befehl im IN1 - Ergebnis = 0, falls t == dem Befehl
JUMP≠ -21	Vergleiche, ob der letzte Befehl 'LOADI PC 0' ist, falls nicht springe zum Anfang des Programms - springe auch durch a)/b)
LOADI SP a	Lass den Stack-Pointer auf a zeigen - springe nach Übertragung an Stelle a

Aufgabe 2

Eine weitere Möglichkeit, die in DS gespeicherte Adresse zu verändern, wäre mit den Befehlen SUBI/ ADDI/ MULI.
Obwohl im DS eigentlich Kürzel für die verschiedenen Adressräume stehen, können diese auch als ganz normale Zahl verstanden werden. Es lassen sich dann natürlich die oben genannten Operationen durchführen, um den Zahlenwert - und damit den Adressraum - entsprechend zu verändern.

Welche konkretten Befehle?
Geh in die richtige Richtung 3/6