Musterlösung Blatt 14

Bonus Aufgabe 1

zu zeigen ist hier: $sim(d^i, h^{t(i)})$

Beweis durch Induktion über i

$i = 0$: $h^0.E = Ein(h^{-1}, eev^{-1})$

$$= \neg jisr(h^{-1}, eev^{-1}) \wedge \ldots$$
$$= 0 \wedge \ldots$$
$$= 0$$

$h^0.phase = 0$ analog zu

wichtig ist hier:

$$pcce(h^{-1}, eev^{-1}) = ex(h^{-1}) \vee jisr(h, eev)$$
$$= ex(h^{-1}) \vee 1$$
$$= 1$$

$\rightarrow h^0.pc = 0_{32}$

Der Induktionsanfang gilt (wie auch im Folgenden folgt die Korrektheit auch aus vorherigen Korrektheitsbeweisen)

IS: Fallunterscheidung

- $\neg jisr(h, eev)$

  Sei $h.mode[1] = 1$. Es muss also gezeigt werden: $sim(d^i, h^{4 \cdot i})$

  zunächst wird gezeigt: $I(d^i) = h^{4i+2}.I$

$$ptea(h^{4i})00 = h^{4i}.pto[31:2]00 +_{32} 0^{10} \circ va(h^{4i})[31:12]00$$
$$= d^i.pto +_{32} 0^{10} \circ h^{4i}.pc[31:12]00$$
$$= d^i.pto +_{32} 0^{10} \circ d^i.pc[31:12]00$$
$$= ptea(d^i, d^i.pc)$$

$$h^{4i+1}.pte = h^{4i}.m(ptea(h^{4i})) = d^i.m(ptea(h^{4i})00) = d^i.m(ptea(d^i, d^i.pc))$$

$$
\begin{aligned}
ma(h^{4i+1})00 &= h^{4i+1}.pte[31:12] \circ h.va[11:2]\ 00 \\
&= h^{4i+1}.pte[31:12] \circ h.pc[11:2]\ 00 \\
&= h^{4i+1}.pte[31:12] \circ d.pc[11:0] \\
&= d^i.m_4(ptea(d^i, d^i.pc)) \\
&= pma(d^i, d^i.pc)
\end{aligned}
$$

$$
\rightarrow h^{4i+2}.I = h^{4i+1}.m(ma(h^{4i+1})) = d^i.m(pma(d^i, d^i.pc)) = I(d)
$$

Vollkommen analog wird bewiesen, dass im Zyklus $4i+3$ bei einer gefetchten Load o. store Instruktion von der $pma(d^i, d.ea)[31:2]$ geladen wird

Obige Schlussfolgerung geht aus Fig. 202 im Skript hervor und aus

$h^{4i}.E = 0$ $\quad$ $h^{4i}.phase = 0$

$h^{4i+1}.phase = phasein(h^{4i}, eev^{4i}) = 1$

$h^{4i+1}.E = Ein(h^{4i}, eev^{4i}) = 0$

$h^{4i+2}.E = h^{4i+1}.E \oplus h^{4i+1}.phase = 1$

... Diese Gleichungen erhält man durch einfaches Einsetzen.

Das Gleiche gilt für die Korrektheit der Prädikate pcce, mw, gw

Sei $h.mode[1] = 0$: wurde gezeigt in Kapitel 8

- $jisr(h, eev)$: folgt durch Einsetzen und der Korrektheit der Maschine mit Interrupts

Bonus Aufgabe 2

Nur sich ändernde Komponenten werden angegeben

Es gilt stets: $cvm'.c.pr = tail(cvm.c.pr)$

$cvm'.c.prn = tail(cvm.c.prn)$

a) $hd(cvm.c.pr) = e = readgpr(u, i)$

$$cvm'.c.m(x) = \begin{cases} \langle cvm.vm(\tilde{u}).gpr(\tilde{i}_5)\rangle & x = lv(e, cvm.c) \\ cvm.c.m(x) & \text{sonst} \end{cases}$$

✓

$hd(cvm.c.pr) = writegpr(u, i, e)$

$$cvm'.vm(\tilde{u}).gpr(x) = \begin{cases} \tilde{e}_{32} & x = \tilde{i}_{32} \wedge x \neq 0_5 \\ cvm.vm(\tilde{u}).gpr(x) & \text{sonst} \end{cases}$$

b) $hd(cvm.c.pr) = readpage(P, u, i)$

$$cvm'.c.m(x) = \begin{cases} \langle cvm.vm(\tilde{u}).m_4(\tilde{i}_{20} \circ j_{10} \circ 00)\rangle & x = P*[j] \quad j \in [0:2^{10}-1] \\ cvm.c.m(x) & \text{sonst} \end{cases}$$

$hd(cvm.c.pr) = writepage(u, i, P)$

$$cvm'.vm(\tilde{u}).m(x) = \begin{cases} byte(i, cvm.c.m(P*[j])_{32}) & x = \tilde{i}_{20} \circ j_{10} \circ i_2 \wedge j \in [0:1023] \wedge i \in [0:3] \\ cvm'.vm(\tilde{u}).m(x) & \text{sonst} \end{cases}$$

c) Sei $y = (d, c)$

$hd\ (c.pr) = readdisk\ (ppx, spx)$

$$d'.ms(x) = \begin{cases} d.sm(\widetilde{spx}_{28})[j] & x = \widetilde{ppx}_{20} \circ j_{12} \land j \in [0 : 2^{12}-1] \\ & \land x \in A(ram) \\ d.ms(x) & \text{sonst} \end{cases}$$

$y' = d'$

$hd\ (c.pr) = writedisk\ (ppx, spx)$

$$d'.sm(x) = \begin{cases} d.ms_{4K}(\widetilde{ppx}_{20} \circ 0^{12}) & x = \widetilde{spx}_{28} \\ d.sm(x) & \text{sonst} \end{cases}$$

d)
```
uint readgpr (uint u , uint i) {
    uint res;
    res = PCB[u].pcb.gpr(i);
    return res
}

void writegpr (uint u, uint i, uint e) {
    PCB[u].pcb.gpr(i) = e
}
```

```
void readpage (page* P, uint u, uint i) {
    uint a;
    uint b;
    gpr(1) = PTA[PTOI[u] + i];
    gpr(2) = P {1};
    asm (ori 1 1 0b 1^20 0^12);
    a = gpr (1) {2};
    b = gpr (2);
    copyms (a, b, 1024)
}
```

Die Implementation von copyms findet man auf S. 365 des Skripts

```
void writepage (uint u, uint i, page* P) {
    uint a;
    uint b;
    gpr(1) = PTA[PTOI[u] + i];
    asm (ori 1 1 0b 1^20 0^12);
    a = gpr(1);
    gpr(1) = P;
    b = gpr(1);
    copyms (b, a, 1024)
}
```

Die Implementation von readdisk ist dem Skript auf S. 366 zu entnehmen

```
void writedisk (uint ppx, uint spx) {
    int y;
    write ms (spx, hdbase + 4 * K);
    writems (2, hdbase + 4 * (K+1));
    copyms (ppx*4*K, hdbase, K);
    y = 2;
    while y != 0 {
        readms (y, hdbase + 4 * K + 4)
    }
}
```

Bonus Aufgabe 3

sei $N = (N.d, N.ic)$ eine erweiterte Konfiguration.
Die Übergangsfunktion Δ gibt die Menge aller möglichen Nachfolgekonfigurationen an:

$$\Delta(N.d, N.ic) = \begin{cases} \{(\delta(d, eev_{isa}^{N.ic}), N.ic+1)\} & hdidle(N.d) \\ \{(\delta(d, eev_{isa}^{N.ic}), N.ic+1), (\eta(N.d), N.ic)\} & \text{sonst} \end{cases}$$

Wie wir sehen ist die Berechnung (N^i) nichtdeterministisch. Indem wir Disk-Schritte umordnen können wir dies vermeiden. Korrektheitsbeweise werden einfacher, die Gefahr für Spaghettiprogrammierung sinkt.

Zu beachten ist dabei, dass Disk Liveness jetzt für umgeordnete Berechnungen nicht mehr gegeben ist. Deshalb eine zusätzliche Softwarebedingung:

$$\forall_i \forall y \in \{x(j), x(j)+1\} \wedge hdbusy(M^y.d) \rightarrow \exists t > y : pdacc(M, t)$$