JOURNAL OF FORMALIZED MATHEMATICS
Volume 9, Released 1997, Published 2003
Inst. of Computer Science, Univ. of Białystok

While Macro Instructions of $\mathbf{SCM}_{\rm FSA}$

Jing-Chao Chen
Shanghai Jiaotong University
Shanghai

Summary. The article defines *while macro instructions* based on $\mathbf{SCM}_{\rm FSA}$. Some theorems about the generalized halting problems of *while macro instructions* are proved.

MML Identifier: `SCMFSA_9`.

WWW: `http://mizar.org/JFM/Vol9/scmfsa_9.html`

The articles [15], [21], [16], [6], [22], [9], [10], [11], [5], [7], [12], [17], [8], [14], [20], [18], [19], [13], [4], [3], [1], and [2] provide the notation and terminology for this paper.

One can prove the following two propositions:

(1) For every macro instruction I and for every integer location a holds card(**if** $a = 0$ **then** I; Goto(insloc(0)) **else** (Stop$_{\rm SCM_{FSA}}$)) $=$ card$I + 6$.

(2) For every macro instruction I and for every integer location a holds card(**if** $a > 0$ **then** I; Goto(insloc(0)) **else** (Stop$_{\rm SCM_{FSA}}$)) $=$ card$I + 6$.

Let a be an integer location and let I be a macro instruction. The functor **while** $a = 0$ **do** I yields a macro instruction and is defined as follows:

(Def. 1) **while** $a = 0$ **do** $I =$ (**if** $a = 0$ **then** I; Goto(insloc(0)) **else** (Stop$_{\rm SCM_{FSA}}$))$+\cdot$(insloc(card$I+$ 4)$\dot{\longmapsto}$goto insloc(0)).

The functor **while** $a > 0$ **do** I yielding a macro instruction is defined as follows:

(Def. 2) **while** $a > 0$ **do** $I =$ (**if** $a > 0$ **then** I; Goto(insloc(0)) **else** (Stop$_{\rm SCM_{FSA}}$))$+\cdot$(insloc(card$I+$ 4)$\dot{\longmapsto}$goto insloc(0)).

One can prove the following proposition

(3) For every macro instruction I and for every integer location a holds card(**if** $a = 0$ **then** Stop$_{\rm SCM_{FSA}}$ **else** (**if** $a > 0$ **then** Stop$_{\rm SCM_{FSA}}$ **else** (I; Goto(insloc(0))))) $=$ card$I + 11$.

Let a be an integer location and let I be a macro instruction. The functor **while** $a < 0$ **do** I yields a macro instruction and is defined by:

(Def. 3) **while** $a < 0$ **do** $I =$ (**if** $a = 0$ **then** Stop$_{\rm SCM_{FSA}}$ **else** (**if** $a > 0$ **then** Stop$_{\rm SCM_{FSA}}$ **else** (I; Goto(insloc(0)))))$+\cdot$(insloc(card$I$ 4)$\dot{\longmapsto}$goto insloc(0)).

One can prove the following propositions:

(4) For every macro instruction I and for every integer location a holds card(**while** $a = 0$ **do** I) $=$ card$I + 6$.

(5) For every macro instruction I and for every integer location a holds $\mathrm{card}(\mathbf{while}\ a > 0\ \mathbf{do}\ I) = \mathrm{card}\,I + 6$.

(6) For every macro instruction I and for every integer location a holds $\mathrm{card}(\mathbf{while}\ a < 0\ \mathbf{do}\ I) = \mathrm{card}\,I + 11$.

(7) For every integer location a and for every instruction-location l of $\mathbf{SCM}_{\mathrm{FSA}}$ holds $\mathbf{if}\ a = 0\ \mathbf{goto}\ l \neq \mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}}$.

(8) For every integer location a and for every instruction-location l of $\mathbf{SCM}_{\mathrm{FSA}}$ holds $\mathbf{if}\ a > 0\ \mathbf{goto}\ l \neq \mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}}$.

(9) For every instruction-location l of $\mathbf{SCM}_{\mathrm{FSA}}$ holds $\mathrm{goto}\ l \neq \mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}}$.

(10) Let a be an integer location and I be a macro instruction. Then $\mathrm{insloc}(0) \in \mathrm{dom}(\mathbf{while}\ a = 0\ \mathbf{do}\ I)$ and $\mathrm{insloc}(1) \in \mathrm{dom}(\mathbf{while}\ a = 0\ \mathbf{do}\ I)$ and $\mathrm{insloc}(0) \in \mathrm{dom}(\mathbf{while}\ a > 0\ \mathbf{do}\ I)$ and $\mathrm{insloc}(1) \in \mathrm{dom}(\mathbf{while}\ a > 0\ \mathbf{do}\ I)$.

(11) Let a be an integer location and I be a macro instruction. Then $(\mathbf{while}\ a = 0\ \mathbf{do}\ I)(\mathrm{insloc}(0)) = \mathbf{if}\ a = 0\ \mathbf{goto}\ \mathrm{insloc}(4)$ and $(\mathbf{while}\ a = 0\ \mathbf{do}\ I)(\mathrm{insloc}(1)) = \mathrm{goto}\ \mathrm{insloc}(2)$ and $(\mathbf{while}\ a > 0\ \mathbf{do}\ I)(\mathrm{insloc}(0)) = \mathbf{if}\ a > 0\ \mathbf{goto}\ \mathrm{insloc}(4)$ and $(\mathbf{while}\ a > 0\ \mathbf{do}\ I)(\mathrm{insloc}(1)) = \mathrm{goto}\ \mathrm{insloc}(2)$.

(12) Let a be an integer location, I be a macro instruction, and k be a natural number. If $k < 6$, then $\mathrm{insloc}(k) \in \mathrm{dom}(\mathbf{while}\ a = 0\ \mathbf{do}\ I)$.

(13) Let a be an integer location, I be a macro instruction, and k be a natural number. If $k < 6$, then $\mathrm{insloc}(\mathrm{card}\,I + k) \in \mathrm{dom}(\mathbf{while}\ a = 0\ \mathbf{do}\ I)$.

(14) For every integer location a and for every macro instruction I holds $(\mathbf{while}\ a = 0\ \mathbf{do}\ I)(\mathrm{insloc}(\mathrm{card}\,I + 5)) = \mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}}$.

(15) For every integer location a and for every macro instruction I holds $(\mathbf{while}\ a = 0\ \mathbf{do}\ I)(\mathrm{insloc}(3)) = \mathrm{goto}\ \mathrm{insloc}(\mathrm{card}\,I + 5)$.

(16) For every integer location a and for every macro instruction I holds $(\mathbf{while}\ a = 0\ \mathbf{do}\ I)(\mathrm{insloc}(2)) = \mathrm{goto}\ \mathrm{insloc}(3)$.

(17) Let a be an integer location, I be a macro instruction, and k be a natural number. If $k < \mathrm{card}\,I + 6$, then $\mathrm{insloc}(k) \in \mathrm{dom}(\mathbf{while}\ a = 0\ \mathbf{do}\ I)$.

(18) Let s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, I be a macro instruction, and a be a read-write integer location. If $s(a) \neq 0$, then $\mathbf{while}\ a = 0\ \mathbf{do}\ I$ is halting on s and $\mathbf{while}\ a = 0\ \mathbf{do}\ I$ is closed on s.

(19) Let a be an integer location, I be a macro instruction, s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, and k be a natural number. Suppose that

(i) I is closed on s and halting on s,

(ii) $k < \mathrm{LifeSpan}(s + \cdot (I + \cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0))))$,

(iii) $\mathbf{IC}_{(\mathrm{Computation}(s+\cdot((\mathbf{while}\ a=0\ \mathbf{do}\ I)+\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(1+k)} = \mathbf{IC}_{(\mathrm{Computation}(s+\cdot(I+\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(k)} + 4$, and

(iv) $(\mathrm{Computation}(s+\cdot((\mathbf{while}\ a = 0\ \mathbf{do}\ I)+\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(1+k) \restriction (\mathrm{Int\text{-}Locations} \cup \mathrm{FinSeq\text{-}Locations}) = (\mathrm{Computation}(s+\cdot(I+\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(k) \restriction (\mathrm{Int\text{-}Locations} \cup \mathrm{FinSeq\text{-}Locations})$.

Then $\mathbf{IC}_{(\mathrm{Computation}(s+\cdot((\mathbf{while}\ a=0\ \mathbf{do}\ I)+\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(1+k+1)} = \mathbf{IC}_{(\mathrm{Computation}(s+\cdot(I+\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(k+1)} + 4$ and $(\mathrm{Computation}(s+\cdot((\mathbf{while}\ a = 0\ \mathbf{do}\ I)+\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(1+k+1) \restriction (\mathrm{Int\text{-}Locations} \cup \mathrm{FinSeq\text{-}Locations}) = (\mathrm{Computation}(s+\cdot(I+\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(k+1) \restriction (\mathrm{Int\text{-}Locations} \cup \mathrm{FinSeq\text{-}Locations})$.

(20) Let a be an integer location, I be a macro instruction, and s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$. Suppose I is closed on s and halting on s and $\mathbf{IC}_{(\mathrm{Computation}(s+\cdot((\mathbf{while}\ a=0\ \mathbf{do}\ I)+\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(1+\mathrm{LifeSpan}(s+\cdot(I+\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))} = \mathbf{IC}_{(\mathrm{Computation}(s+\cdot(I+\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(\mathrm{LifeSpan}(s+\cdot(I+\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))} + 4$. Then $\mathrm{CurInstr}((\mathrm{Computation}(s+\cdot((\mathbf{while}\ a = 0\ \mathbf{do}\ I)+\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(1+\mathrm{LifeSpan}(s+\cdot(I+\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))) = \mathrm{goto}\ \mathrm{insloc}(\mathrm{card}\,I + 4)$.

(21) For every integer location a and for every macro instruction I holds $(\textbf{while}\ a = 0\ \textbf{do}\ I)(\mathrm{insloc}(\mathrm{card}\,I+4)) = \mathrm{goto}\ \mathrm{insloc}(0)$.

(22) Let s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, I be a macro instruction, and a be a read-write integer location. Suppose I is closed on s and halting on s and $s(a) = 0$. Then $\mathbf{IC}_{(\mathrm{Computation}(s+\cdot((\textbf{while}\ a=0\ \textbf{do}\ I)+\cdot\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(\mathrm{LifeSpan}(s+\cdot(I+\cdot\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0))))+3)} = \mathrm{insloc}(0)$ and for every natural number k such that $k \leq \mathrm{LifeSpan}(s+\cdot(I+\cdot\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0))))+3$ holds $\mathbf{IC}_{(\mathrm{Computation}(s+\cdot((\textbf{while}\ a=0\ \textbf{do}\ I)+\cdot\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(k)} \in \mathrm{dom}(\textbf{while}\ a = 0\ \textbf{do}\ I)$.

In the sequel s is a state of $\mathbf{SCM}_{\mathrm{FSA}}$, I is a macro instruction, and a is a read-write integer location.

Let us consider s, I, a. The functor $\mathit{StepWhile{=}0}(a,I,s)$ yields a function from $\mathbb{N}$ into $\prod$(the object kind of $\mathbf{SCM}_{\mathrm{FSA}}$) and is defined by:

(Def. 4) $(\mathit{StepWhile{=}0}(a,I,s))(0) = s$ and for every natural number i holds $(\mathit{StepWhile{=}0}(a,I,s))(i+1) = (\mathrm{Computation}((\mathit{StepWhile{=}0}(a,I,s))(i)+\cdot((\textbf{while}\ a = 0\ \textbf{do}\ I)+\cdot s_0)))(\mathrm{LifeSpan}((\mathit{StepWhile{=}0}(a,I,s))(i)+\cdot(I+\cdot s_0))+3)$, where $s_0 = \mathrm{Start\text{-}At}(\mathrm{insloc}(0))$.

In the sequel k, n denote natural numbers.

We now state the proposition

(25)[1] $(\mathit{StepWhile{=}0}(a,I,s))(k+1) = (\mathit{StepWhile{=}0}(a,I,(\mathit{StepWhile{=}0}(a,I,s))(k)))(1)$.

The scheme *MinIndex* deals with a unary functor $\mathcal{F}$ yielding a natural number and a natural number $\mathcal{A}$, and states that:

There exists k such that $\mathcal{F}(k) = 0$ and for every n such that $\mathcal{F}(n) = 0$ holds $k \leq n$

provided the following conditions are met:

- $\mathcal{F}(0) = \mathcal{A}$, and
- For every k holds $\mathcal{F}(k+1) < \mathcal{F}(k)$ or $\mathcal{F}(k) = 0$.

Next we state three propositions:

(26) For all functions f, g holds $f+\cdot g+\cdot g = f+\cdot g$.

(27) For all functions f, g, h and for every set D such that $(f+\cdot g)\restriction D = h\restriction D$ holds $(h+\cdot g)\restriction D = (f+\cdot g)\restriction D$.

(28) For all functions f, g, h and for every set D such that $f\restriction D = h\restriction D$ holds $(h+\cdot g)\restriction D = (f+\cdot g)\restriction D$.

We now state several propositions:

(29) For all states s_1, s_2 of $\mathbf{SCM}_{\mathrm{FSA}}$ such that $\mathbf{IC}_{(s_1)} = \mathbf{IC}_{(s_2)}$ and $s_1\restriction(\mathrm{Int\text{-}Locations} \cup \mathrm{FinSeq\text{-}Locations}) = s_2\restriction(\mathrm{Int\text{-}Locations} \cup \mathrm{FinSeq\text{-}Locations})$ and $s_1\restriction I_1 = s_2\restriction I_1$ holds $s_1 = s_2$, where $I_1 =$ the instruction locations of $\mathbf{SCM}_{\mathrm{FSA}}$.

(30) Let I be a macro instruction, a be a read-write integer location, and s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$. Then $(\mathit{StepWhile{=}0}(a,I,s))(0+1) = (\mathrm{Computation}(s+\cdot((\textbf{while}\ a = 0\ \textbf{do}\ I)+\cdot s_0)))(\mathrm{LifeSpan}(s+\cdot(I+\cdot s_0))+3)$, where $s_0 = \mathrm{Start\text{-}At}(\mathrm{insloc}(0))$.

(31) Let I be a macro instruction, a be a read-write integer location, s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, and k, n be natural numbers. Suppose $\mathbf{IC}_{(\mathit{StepWhile{=}0}(a,I,s))(k)} = \mathrm{insloc}(0)$ and $(\mathit{StepWhile{=}0}(a,I,s))(k) = (\mathrm{Computation}(s+\cdot((\textbf{while}\ a = 0\ \textbf{do}\ I)+\cdot\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(n)$. Then $(\mathit{StepWhile{=}0}(a,I,s))(k) = (\mathit{StepWhile{=}0}(a,I,s))(k)+\cdot((\textbf{while}\ a = 0\ \textbf{do}\ I)+\cdot\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))$ and $(\mathit{StepWhile{=}0}(a,I,s))(k+1) = (\mathrm{Computation}(s+\cdot((\textbf{while}\ a = 0\ \textbf{do}\ I)+\cdot\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(n+(\mathrm{LifeSpan}((\mathit{StepWhile{=}0}(a,I,s))(k)+\cdot(I+\cdot\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0))))+3))$.

[1] The propositions (23) and (24) have been removed.

(32) Let I be a macro instruction, a be a read-write integer location, and s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$. Suppose that

(i) for every natural number k holds I is closed on $(\mathit{StepWhile{=}0}(a,I,s))(k)$ and halting on $(\mathit{StepWhile{=}0}(a,I,s))(k)$, and

(ii) there exists a function f from $\prod$(the object kind of $\mathbf{SCM}_{\mathrm{FSA}}$) into $\mathbb{N}$ such that for every natural number k holds $f((\mathit{StepWhile{=}0}(a,I,s))(k+1)) < f((\mathit{StepWhile{=}0}(a,I,s))(k))$ or $f((\mathit{StepWhile{=}0}(a,I,s))(k)) = 0$ but $f((\mathit{StepWhile{=}0}(a,I,s))(k)) = 0$ iff $(\mathit{StepWhile{=}0}(a,I,s))(k)(a) \neq 0$.

Then **while** $a = 0$ **do** I is halting on s and **while** $a = 0$ **do** I is closed on s.

(33) Let I be a parahalting macro instruction, a be a read-write integer location, and s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$. Given a function f from $\prod$(the object kind of $\mathbf{SCM}_{\mathrm{FSA}}$) into $\mathbb{N}$ such that let k be a natural number. Then $f((\mathit{StepWhile{=}0}(a,I,s))(k+1)) < f((\mathit{StepWhile{=}0}(a,I,s))(k))$ or $f((\mathit{StepWhile{=}0}(a,I,s))(k)) = 0$ but $f((\mathit{StepWhile{=}0}(a,I,s))(k)) = 0$ iff $(\mathit{StepWhile{=}0}(a,I,s))(k)(a) \neq 0$. Then **while** $a = 0$ **do** I is halting on s and **while** $a = 0$ **do** I is closed on s.

(34) Let I be a parahalting macro instruction and a be a read-write integer location. Given a function f from $\prod$(the object kind of $\mathbf{SCM}_{\mathrm{FSA}}$) into $\mathbb{N}$ such that let s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$. Then $f((\mathit{StepWhile{=}0}(a,I,s))(1)) < f(s)$ or $f(s) = 0$ but $f(s) = 0$ iff $s(a) \neq 0$. Then **while** $a = 0$ **do** I is parahalting.

(35) For all instruction-locations l_1, l_2 of $\mathbf{SCM}_{\mathrm{FSA}}$ and for every integer location a holds $l_1 \mapsto$ goto l_2 does not destroy a.

(36) For every instruction i of $\mathbf{SCM}_{\mathrm{FSA}}$ such that i does not destroy intloc(0) holds Macro(i) is good.

Let I, J be good macro instructions and let a be an integer location. One can check that **if** $a = 0$ **then** I **else** J is good.

Let I be a good macro instruction and let a be an integer location. Observe that **while** $a = 0$ **do** I is good.

The following propositions are true:

(37) Let a be an integer location, I be a macro instruction, and k be a natural number. If $k < 6$, then $\mathrm{insloc}(k) \in \mathrm{dom}(\mathbf{while}\ a > 0\ \mathbf{do}\ I)$.

(38) Let a be an integer location, I be a macro instruction, and k be a natural number. If $k < 6$, then $\mathrm{insloc}(\mathrm{card}\,I + k) \in \mathrm{dom}(\mathbf{while}\ a > 0\ \mathbf{do}\ I)$.

(39) For every integer location a and for every macro instruction I holds $(\mathbf{while}\ a > 0\ \mathbf{do}\ I)(\mathrm{insloc}(\mathrm{card}\,I + 5)) = \mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}}$.

(40) For every integer location a and for every macro instruction I holds $(\mathbf{while}\ a > 0\ \mathbf{do}\ I)(\mathrm{insloc}(3)) = \mathrm{goto}\ \mathrm{insloc}(\mathrm{card}\,I + 5)$.

(41) For every integer location a and for every macro instruction I holds $(\mathbf{while}\ a > 0\ \mathbf{do}\ I)(\mathrm{insloc}(2)) = \mathrm{goto}\ \mathrm{insloc}(3)$.

(42) Let a be an integer location, I be a macro instruction, and k be a natural number. If $k < \mathrm{card}\,I + 6$, then $\mathrm{insloc}(k) \in \mathrm{dom}(\mathbf{while}\ a > 0\ \mathbf{do}\ I)$.

(43) Let s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, I be a macro instruction, and a be a read-write integer location. If $s(a) \leq 0$, then **while** $a > 0$ **do** I is halting on s and **while** $a > 0$ **do** I is closed on s.

The following four propositions are true:

(44) Let a be an integer location, I be a macro instruction, s be a state of $\mathbf{SCM}_{\rm FSA}$, and k be a natural number. Suppose that

(i) I is closed on s and halting on s,

(ii) $k < {\rm LifeSpan}(s{+}\cdot(I{+}\cdot\,{\rm Start\text{-}At}({\rm insloc}(0))))$,

(iii) $\mathbf{IC}_{({\rm Computation}(s+\cdot((\mathbf{while}\ a>0\ \mathbf{do}\ I)+\cdot\,{\rm Start\text{-}At}({\rm insloc}(0)))))(1+k)} = \mathbf{IC}_{({\rm Computation}(s+\cdot(I+\cdot\,{\rm Start\text{-}At}({\rm insloc}(0)))))(k)} +$ 4, and

(iv) $({\rm Computation}(s{+}\cdot((\mathbf{while}\ a>0\ \mathbf{do}\ I){+}\cdot\,{\rm Start\text{-}At}({\rm insloc}(0)))))(1+k)\restriction D = ({\rm Computation}(s{+}\cdot(I{+}\cdot\,{\rm Start\text{-}At}({\rm insloc}(0)))$

Then $\mathbf{IC}_{({\rm Computation}(s+\cdot((\mathbf{while}\ a>0\ \mathbf{do}\ I)+\cdot\,{\rm Start\text{-}At}({\rm insloc}(0)))))(1+k+1)} = \mathbf{IC}_{({\rm Computation}(s+\cdot(I+\cdot\,{\rm Start\text{-}At}({\rm insloc}(0)))))(k+1)} +$ 4 and $({\rm Computation}(s{+}\cdot((\mathbf{while}\ a > 0\ \mathbf{do}\ I){+}\cdot\,{\rm Start\text{-}At}({\rm insloc}(0)))))(1+k+1)\restriction D = ({\rm Computation}(s{+}\cdot(I{+}\cdot\,{\rm Start\text{-}At}({\rm insloc}(0)))))(k+1)\restriction D$, where $D = {\rm Int\text{-}Locations} \cup {\rm FinSeq\text{-}Locations}$.

(45) Let a be an integer location, I be a macro instruction, and s be a state of $\mathbf{SCM}_{\rm FSA}$. Suppose I is closed on s and halting on s and $\mathbf{IC}_{({\rm Computation}(s+\cdot((\mathbf{while}\ a>0\ \mathbf{do}\ I)+\cdot\,{\rm Start\text{-}At}({\rm insloc}(0)))))(1+{\rm LifeSpan}(s+\cdot(I+\cdot\,{\rm Start\text{-}At}({\rm insloc}(0)))))} = \mathbf{IC}_{({\rm Computation}(s+\cdot(I+\cdot\,{\rm Start\text{-}At}({\rm insloc}(0)))))({\rm LifeSpan}(s+\cdot(I+\cdot\,{\rm Start\text{-}At}({\rm insloc}(0)))))} + 4$. Then ${\rm CurInstr}(({\rm Computation}(s{+}\cdot((\mathbf{while}\ a > 0\ \mathbf{do}\ I){+}\cdot\,{\rm Start\text{-}At}({\rm insloc}(0)))))(1+{\rm LifeSpan}(s{+}\cdot(I{+}\cdot\,{\rm Start\text{-}At}({\rm insloc}(0)))))) = {\rm goto}\ {\rm insloc}({\rm card}\,I + 4)$.

(46) For every integer location a and for every macro instruction I holds $(\mathbf{while}\ a > 0\ \mathbf{do}\ I)({\rm insloc}({\rm card}\,I+4)) = {\rm goto}\ {\rm insloc}(0)$.

(47) Let s be a state of $\mathbf{SCM}_{\rm FSA}$, I be a macro instruction, and a be a read-write integer location. Suppose I is closed on s and halting on s and $s(a) > 0$. Then $\mathbf{IC}_{({\rm Computation}(s+\cdot((\mathbf{while}\ a>0\ \mathbf{do}\ I)+\cdot\,{\rm Start\text{-}At}({\rm insloc}(0)))))({\rm LifeSpan}(s+\cdot(I+\cdot\,{\rm Start\text{-}At}({\rm insloc}(0))))+3)} = {\rm insloc}(0)$ and for every natural number k such that $k \leq {\rm LifeSpan}(s{+}\cdot(I{+}\cdot\,{\rm Start\text{-}At}({\rm insloc}(0)))) + 3$ holds $\mathbf{IC}_{({\rm Computation}(s+\cdot((\mathbf{while}\ a>0\ \mathbf{do}\ I)+\cdot\,{\rm Start\text{-}At}({\rm insloc}(0)))))(k)} \in {\rm dom}(\mathbf{while}\ a > 0\ \mathbf{do}\ I)$.

In the sequel s denotes a state of $\mathbf{SCM}_{\rm FSA}$, I denotes a macro instruction, and a denotes a read-write integer location.

Let us consider s, I, a. The functor $\mathit{StepWhile{>}0}(a,I,s)$ yielding a function from $\mathbb{N}$ into $\prod$(the object kind of $\mathbf{SCM}_{\rm FSA}$) is defined by:

(Def. 5) $(\mathit{StepWhile{>}0}(a,I,s))(0) = s$ and for every natural number i holds $(\mathit{StepWhile{>}0}(a,I,s))(i+1) = ({\rm Computation}((\mathit{StepWhile{>}0}(a,I,s))(i){+}\cdot((\mathbf{while}\ a > 0\ \mathbf{do}\ I){+}\cdot s_0)))({\rm LifeSpan}((\mathit{StepWhile{>}0}(a,I,s))(i){+}\cdot(I{+}\cdot s_0))$ 3), where $s_0 = {\rm Start\text{-}At}({\rm insloc}(0))$.

Next we state several propositions:

(50)[2] $(\mathit{StepWhile{>}0}(a,I,s))(k+1) = (\mathit{StepWhile{>}0}(a,I,(\mathit{StepWhile{>}0}(a,I,s))(k)))(1)$.

(51) Let I be a macro instruction, a be a read-write integer location, and s be a state of $\mathbf{SCM}_{\rm FSA}$. Then $(\mathit{StepWhile{>}0}(a,I,s))(0+1) = ({\rm Computation}(s{+}\cdot((\mathbf{while}\ a > 0\ \mathbf{do}\ I){+}\cdot s_0)))({\rm LifeSpan}(s{+}\cdot(I{+}\cdot s_0)) + 3)$, where $s_0 = {\rm Start\text{-}At}({\rm insloc}(0))$.

(52) Let I be a macro instruction, a be a read-write integer location, s be a state of $\mathbf{SCM}_{\rm FSA}$, and k, n be natural numbers. Suppose $\mathbf{IC}_{(\mathit{StepWhile>0}(a,I,s))(k)} = {\rm insloc}(0)$ and $(\mathit{StepWhile{>}0}(a,I,s))(k) = ({\rm Computation}(s{+}\cdot((\mathbf{while}\ a > 0\ \mathbf{do}\ I){+}\cdot\,{\rm Start\text{-}At}({\rm insloc}(0)))))(n)$. Then $(\mathit{StepWhile{>}0}(a,I,s))(k) = (\mathit{StepWhile{>}0}(a,I,s))(k){+}\cdot((\mathbf{while}\ a > 0\ \mathbf{do}\ I){+}\cdot\,{\rm Start\text{-}At}({\rm insloc}(0)))$ and $(\mathit{StepWhile{>}0}(a,I,s))(k+1) = ({\rm Computation}(s{+}\cdot((\mathbf{while}\ a > 0\ \mathbf{do}\ I){+}\cdot\,{\rm Start\text{-}At}({\rm insloc}(0)))))(n + ({\rm LifeSpan}((\mathit{StepWhile{>}0}(a,I,s))(k){+}\cdot(I{+}\cdot\,{\rm Start\text{-}At}({\rm insloc}(0)))) + 3))$.

(53) Let I be a macro instruction, a be a read-write integer location, and s be a state of $\mathbf{SCM}_{\rm FSA}$. Suppose that

(i) for every natural number k holds I is closed on $(\mathit{StepWhile{>}0}(a,I,s))(k)$ and halting on $(\mathit{StepWhile{>}0}(a,I,s))(k)$, and

(ii) there exists a function f from $\prod$(the object kind of $\mathbf{SCM}_{\rm FSA}$) into $\mathbb{N}$ such that for every natural number k holds $f((\mathit{StepWhile{>}0}(a,I,s))(k+1)) < f((\mathit{StepWhile{>}0}(a,I,s))(k))$ or

[2] The propositions (48) and (49) have been removed.

$f((\mathit{StepWhile{>}0}(a,I,s))(k)) = 0$ but $f((\mathit{StepWhile{>}0}(a,I,s))(k)) = 0$ iff $(\mathit{StepWhile{>}0}(a,I,s))(k)(a) \leq 0$.

Then **while** $a > 0$ **do** I is halting on s and **while** $a > 0$ **do** I is closed on s.

(54) Let I be a parahalting macro instruction, a be a read-write integer location, and s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$. Given a function f from $\prod$(the object kind of $\mathbf{SCM}_{\mathrm{FSA}}$) into $\mathbb{N}$ such that let k be a natural number. Then $f((\mathit{StepWhile{>}0}(a,I,s))(k+1)) < f((\mathit{StepWhile{>}0}(a,I,s))(k))$ or $f((\mathit{StepWhile{>}0}(a,I,s))(k)) = 0$ but $f((\mathit{StepWhile{>}0}(a,I,s))(k)) = 0$ iff $(\mathit{StepWhile{>}0}(a,I,s))(k)(a) \leq 0$. Then **while** $a > 0$ **do** I is halting on s and **while** $a > 0$ **do** I is closed on s.

(55) Let I be a parahalting macro instruction and a be a read-write integer location. Given a function f from $\prod$(the object kind of $\mathbf{SCM}_{\mathrm{FSA}}$) into $\mathbb{N}$ such that let s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$. Then $f((\mathit{StepWhile{>}0}(a,I,s))(1)) < f(s)$ or $f(s) = 0$ but $f(s) = 0$ iff $s(a) \leq 0$. Then **while** $a > 0$ **do** I is parahalting.

Let I, J be good macro instructions and let a be an integer location. Note that **if** $a > 0$ **then** I **else** J is good.

Let I be a good macro instruction and let a be an integer location. Note that **while** $a > 0$ **do** I is good.

Acknowledgments

The author wishes to thank Prof. Andrzej Trybulec and Dr. Grzegorz Bancerek for their helpful comments and encouragement during his stay in Białystok.

References

[1] Noriko Asamoto. Conditional branch macro instructions of $\mathbf{SCM}_{\mathrm{FSA}}$. Part I. *Journal of Formalized Mathematics*, 8, 1996. `http://mizar.org/JFM/Vol8/scmfsa8a.html`.

[2] Noriko Asamoto. Conditional branch macro instructions of $\mathbf{SCM}_{\mathrm{FSA}}$. Part II. *Journal of Formalized Mathematics*, 8, 1996. `http://mizar.org/JFM/Vol8/scmfsa8b.html`.

[3] Noriko Asamoto. Constant assignment macro instructions of $\mathbf{SCM}_{\mathrm{FSA}}$. Part II. *Journal of Formalized Mathematics*, 8, 1996. `http://mizar.org/JFM/Vol8/scmfsa7b.html`.

[4] Noriko Asamoto, Yatsuka Nakamura, Piotr Rudnicki, and Andrzej Trybulec. On the composition of macro instructions. Part II. *Journal of Formalized Mathematics*, 8, 1996. `http://mizar.org/JFM/Vol8/scmfsa6b.html`.

[5] Grzegorz Bancerek. Cardinal numbers. *Journal of Formalized Mathematics*, 1, 1989. `http://mizar.org/JFM/Vol1/card_1.html`.

[6] Grzegorz Bancerek. The fundamental properties of natural numbers. *Journal of Formalized Mathematics*, 1, 1989. `http://mizar.org/JFM/Vol1/nat_1.html`.

[7] Grzegorz Bancerek. König's theorem. *Journal of Formalized Mathematics*, 2, 1990. `http://mizar.org/JFM/Vol2/card_3.html`.

[8] Grzegorz Bancerek and Piotr Rudnicki. Development of terminology for **scm**. *Journal of Formalized Mathematics*, 5, 1993. `http://mizar.org/JFM/Vol5/scm_1.html`.

[9] Czesław Byliński. Functions and their basic properties. *Journal of Formalized Mathematics*, 1, 1989. `http://mizar.org/JFM/Vol1/funct_1.html`.

[10] Czesław Byliński. Functions from a set to a set. *Journal of Formalized Mathematics*, 1, 1989. `http://mizar.org/JFM/Vol1/funct_2.html`.

[11] Czesław Byliński. The modification of a function by a function and the iteration of the composition of a function. *Journal of Formalized Mathematics*, 2, 1990. `http://mizar.org/JFM/Vol2/funct_4.html`.

[12] Yatsuka Nakamura and Andrzej Trybulec. A mathematical model of CPU. *Journal of Formalized Mathematics*, 4, 1992. `http://mizar.org/JFM/Vol4/ami_1.html`.

[13] Piotr Rudnicki and Andrzej Trybulec. Memory handling for $\mathbf{SCM}_{\mathrm{FSA}}$. *Journal of Formalized Mathematics*, 8, 1996. `http://mizar.org/JFM/Vol8/sf_mastr.html`.

[14] Yasushi Tanaka. On the decomposition of the states of SCM. *Journal of Formalized Mathematics*, 5, 1993. `http://mizar.org/JFM/Vol5/ami_5.html`.

[15] Andrzej Trybulec. Tarski Grothendieck set theory. *Journal of Formalized Mathematics*, Axiomatics, 1989. `http://mizar.org/JFM/Axiomatics/tarski.html`.

[16] Andrzej Trybulec. Subsets of real numbers. *Journal of Formalized Mathematics*, Addenda, 2003. `http://mizar.org/JFM/Addenda/numbers.html`.

[17] Andrzej Trybulec and Yatsuka Nakamura. Some remarks on the simple concrete model of computer. *Journal of Formalized Mathematics*, 5, 1993. `http://mizar.org/JFM/Vol5/ami_3.html`.

[18] Andrzej Trybulec and Yatsuka Nakamura. Modifying addresses of instructions of $\mathbf{SCM}_{\mathrm{FSA}}$. *Journal of Formalized Mathematics*, 8, 1996. `http://mizar.org/JFM/Vol8/scmfsa_4.html`.

[19] Andrzej Trybulec, Yatsuka Nakamura, and Noriko Asamoto. On the compositions of macro instructions. Part I. *Journal of Formalized Mathematics*, 8, 1996. `http://mizar.org/JFM/Vol8/scmfsa6a.html`.

[20] Andrzej Trybulec, Yatsuka Nakamura, and Piotr Rudnicki. The $\mathbf{SCM}_{\mathrm{FSA}}$ computer. *Journal of Formalized Mathematics*, 8, 1996. `http://mizar.org/JFM/Vol8/scmfsa_2.html`.

[21] Zinaida Trybulec. Properties of subsets. *Journal of Formalized Mathematics*, 1, 1989. `http://mizar.org/JFM/Vol1/subset_1.html`.

[22] Edmund Woronowicz. Relations and their basic properties. *Journal of Formalized Mathematics*, 1, 1989. `http://mizar.org/JFM/Vol1/relat_1.html`.

Received December 10, 1997

Published January 2, 2004