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

# The `loop` and `Times` Macroinstruction for $\mathbf{SCM}_{\rm FSA}$

Noriko Asamoto
Ochanomizu University
Tokyo

**Summary.** We implement two macroinstructions `loop` and `Times` which iterate macroinstructions of $\mathbf{SCM}_{\rm FSA}$. In a `loop` macroinstruction it jumps to the head when the original macroinstruction stops, in a `Times` macroinstruction it behaves as if the original macroinstrucion repeats $n$ times.

MML Identifier: `SCMFSA8C`.

WWW: `http://mizar.org/JFM/Vol9/scmfsa8c.html`

The articles [18], [17], [7], [11], [24], [10], [12], [9], [6], [13], [19], [16], [23], [20], [21], [8], [15], [22], [4], [5], [3], [1], [2], and [14] provide the notation and terminology for this paper.

## 1. Preliminaries

One can check that there exists a macro instruction which is pseudo-paraclosed.

One can prove the following propositions:

(2)[1] Let $s$ be a state of $\mathbf{SCM}_{\rm FSA}$ and $P$ be an initial finite partial state of $\mathbf{SCM}_{\rm FSA}$. Suppose $P$ is pseudo-closed on $s$. Let $k$ be a natural number. Suppose that for every natural number $n$ such that $n \leq k$ holds $\mathbf{IC}_{(\mathrm{Computation}(s+\cdot(P+\cdot\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(n)} \in \mathrm{dom}\, P$. Then $k < \mathrm{pseudo}-\mathrm{LifeSpan}(s,P)$.

(6)[2] For every function $f$ and for every set $x$ such that $x \in \mathrm{dom}\, f$ holds $f+\cdot(x \dot{\longmapsto} f(x)) = f$.

(7) For every instruction-location $l$ of $\mathbf{SCM}_{\rm FSA}$ holds $l+0 = l$.

(8) For every instruction $i$ of $\mathbf{SCM}_{\rm FSA}$ holds $\mathrm{IncAddr}(i,0) = i$.

(9) For every programmed finite partial state $P$ of $\mathbf{SCM}_{\rm FSA}$ holds $\mathrm{ProgramPart}(\mathrm{Relocated}(P,0)) = P$.

(10) For all finite partial states $P$, $Q$ of $\mathbf{SCM}_{\rm FSA}$ such that $P \subseteq Q$ holds $\mathrm{ProgramPart}(P) \subseteq \mathrm{ProgramPart}(Q)$.

(11) For all programmed finite partial states $P$, $Q$ of $\mathbf{SCM}_{\rm FSA}$ and for every natural number $k$ such that $P \subseteq Q$ holds $\mathrm{Shift}(P,k) \subseteq \mathrm{Shift}(Q,k)$.

(12) For all finite partial states $P$, $Q$ of $\mathbf{SCM}_{\rm FSA}$ and for every natural number $k$ such that $P \subseteq Q$ holds $\mathrm{ProgramPart}(\mathrm{Relocated}(P,k)) \subseteq \mathrm{ProgramPart}(\mathrm{Relocated}(Q,k))$.

[1] The proposition (1) has been removed.
[2] The propositions (3)–(5) have been removed.

(13) Let $I$, $J$ be macro instructions and $k$ be a natural number. Suppose $\operatorname{card} I \leq k$ and $k < \operatorname{card} I + \operatorname{card} J$. Let $i$ be an instruction of $\mathbf{SCM}_{\rm FSA}$. If $i = J(\operatorname{insloc}(k -' \operatorname{card} I))$, then $(I;J)(\operatorname{insloc}(k)) = \operatorname{IncAddr}(i, \operatorname{card} I)$.

(14) For every state $s$ of $\mathbf{SCM}_{\rm FSA}$ such that $s(\operatorname{intloc}(0)) = 1$ and $\mathbf{IC}_s = \operatorname{insloc}(0)$ holds $\operatorname{Initialize}(s) = s$.

(15) For every state $s$ of $\mathbf{SCM}_{\rm FSA}$ holds $\operatorname{Initialize}(\operatorname{Initialize}(s)) = \operatorname{Initialize}(s)$.

(16) For every state $s$ of $\mathbf{SCM}_{\rm FSA}$ and for every macro instruction $I$ holds $s{+\cdot}(\operatorname{Initialized}(I){+\cdot}\operatorname{Start-At}(\operatorname{insloc}(0))) = \operatorname{Initialize}(s){+\cdot}(I{+\cdot}\operatorname{Start-At}(\operatorname{insloc}(0)))$.

(17) For every state $s$ of $\mathbf{SCM}_{\rm FSA}$ and for every macro instruction $I$ holds $\operatorname{IExec}(I,s) = \operatorname{IExec}(I, \operatorname{Initialize}(s))$.

(18) For every state $s$ of $\mathbf{SCM}_{\rm FSA}$ and for every macro instruction $I$ such that $s(\operatorname{intloc}(0)) = 1$ holds $s{+\cdot}(I{+\cdot}\operatorname{Start-At}(\operatorname{insloc}(0))) = s{+\cdot}\operatorname{Initialized}(I)$.

(19) For every macro instruction $I$ holds $I{+\cdot}\operatorname{Start-At}(\operatorname{insloc}(0)) \subseteq \operatorname{Initialized}(I)$.

(20) For every instruction-location $l$ of $\mathbf{SCM}_{\rm FSA}$ and for every macro instruction $I$ holds $l \in \operatorname{dom} I$ iff $l \in \operatorname{dom} \operatorname{Initialized}(I)$.

(21) For every state $s$ of $\mathbf{SCM}_{\rm FSA}$ and for every macro instruction $I$ holds $\operatorname{Initialized}(I)$ is closed on $s$ iff $I$ is closed on $\operatorname{Initialize}(s)$.

(22) For every state $s$ of $\mathbf{SCM}_{\rm FSA}$ and for every macro instruction $I$ holds $\operatorname{Initialized}(I)$ is halting on $s$ iff $I$ is halting on $\operatorname{Initialize}(s)$.

(23) For every macro instruction $I$ such that for every state $s$ of $\mathbf{SCM}_{\rm FSA}$ holds $I$ is halting on $\operatorname{Initialize}(s)$ holds $\operatorname{Initialized}(I)$ is halting.

(24) For every macro instruction $I$ such that for every state $s$ of $\mathbf{SCM}_{\rm FSA}$ holds $\operatorname{Initialized}(I)$ is halting on $s$ holds $\operatorname{Initialized}(I)$ is halting.

(25) For every macro instruction $I$ holds $\operatorname{ProgramPart}(\operatorname{Initialized}(I)) = I$.

(26) Let $s$ be a state of $\mathbf{SCM}_{\rm FSA}$, $I$ be a macro instruction, $l$ be an instruction-location of $\mathbf{SCM}_{\rm FSA}$, and $x$ be a set. If $x \in \operatorname{dom} I$, then $I(x) = (s{+\cdot}(I{+\cdot}\operatorname{Start-At}(l)))(x)$.

(27) For every state $s$ of $\mathbf{SCM}_{\rm FSA}$ such that $s(\operatorname{intloc}(0)) = 1$ holds $\operatorname{Initialize}(s)\restriction(\text{Int-Locations} \cup \text{FinSeq-Locations}) = s\restriction(\text{Int-Locations} \cup \text{FinSeq-Locations})$.

(28) Let $s$ be a state of $\mathbf{SCM}_{\rm FSA}$, $I$ be a macro instruction, $a$ be an integer location, and $l$ be an instruction-location of $\mathbf{SCM}_{\rm FSA}$. Then $(s{+\cdot}(I{+\cdot}\operatorname{Start-At}(l)))(a) = s(a)$.

(29) For every programmed finite partial state $I$ of $\mathbf{SCM}_{\rm FSA}$ and for every instruction-location $l$ of $\mathbf{SCM}_{\rm FSA}$ holds $\mathbf{IC}_{\mathbf{SCM}_{\rm FSA}} \in \operatorname{dom}(I{+\cdot}\operatorname{Start-At}(l))$.

(30) For every programmed finite partial state $I$ of $\mathbf{SCM}_{\rm FSA}$ and for every instruction-location $l$ of $\mathbf{SCM}_{\rm FSA}$ holds $(I{+\cdot}\operatorname{Start-At}(l))(\mathbf{IC}_{\mathbf{SCM}_{\rm FSA}}) = l$.

(31) Let $s$ be a state of $\mathbf{SCM}_{\rm FSA}$, $P$ be a finite partial state of $\mathbf{SCM}_{\rm FSA}$, and $l$ be an instruction-location of $\mathbf{SCM}_{\rm FSA}$. Then $\mathbf{IC}_{s{+\cdot}(P{+\cdot}\operatorname{Start-At}(l))} = l$.

(32) For every state $s$ of $\mathbf{SCM}_{\rm FSA}$ and for every instruction $i$ of $\mathbf{SCM}_{\rm FSA}$ such that $\operatorname{InsCode}(i) \in \{0,6,7,8\}$ holds $\operatorname{Exec}(i,s)\restriction(\text{Int-Locations} \cup \text{FinSeq-Locations}) = s\restriction(\text{Int-Locations} \cup \text{FinSeq-Locations})$.

(33) Let $s_1$, $s_2$ be states of $\mathbf{SCM}_{\rm FSA}$. Suppose that

(i) $s_1(\operatorname{intloc}(0)) = s_2(\operatorname{intloc}(0))$,

(ii) for every read-write integer location $a$ holds $s_1(a) = s_2(a)$, and

(iii) for every finite sequence location $f$ holds $s_1(f) = s_2(f)$.

Then $s_1\restriction(\text{Int-Locations} \cup \text{FinSeq-Locations}) = s_2\restriction(\text{Int-Locations} \cup \text{FinSeq-Locations})$.

(34) For every state $s$ of $\mathbf{SCM}_{\mathrm{FSA}}$ and for every programmed finite partial state $P$ of $\mathbf{SCM}_{\mathrm{FSA}}$ holds $(s+\cdot P)\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations}) = s\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations})$.

(35) For all states $s$, $s_3$ of $\mathbf{SCM}_{\mathrm{FSA}}$ holds $(s+\cdot s_3\restriction\text{the instruction locations of }\mathbf{SCM}_{\mathrm{FSA}})\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations}) = s\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations})$.

(36) For every state $s$ of $\mathbf{SCM}_{\mathrm{FSA}}$ holds $\text{Initialize}(s)\restriction\text{the instruction locations of }\mathbf{SCM}_{\mathrm{FSA}} = s\restriction\text{the instruction locations of }\mathbf{SCM}_{\mathrm{FSA}}$.

(37) Let $s$, $s_3$ be states of $\mathbf{SCM}_{\mathrm{FSA}}$ and $I$ be a macro instruction. Then $(s_3+\cdot s\restriction\text{the instruction locations of }\mathbf{SCM}_{\mathrm{FSA}})\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations}) = s_3\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations})$.

(38) For every state $s$ of $\mathbf{SCM}_{\mathrm{FSA}}$ holds $\text{IExec}(\text{Stop}_{\mathbf{SCM}_{\mathrm{FSA}}}, s) = \text{Initialize}(s)+\cdot\text{Start-At}(\text{insloc}(0))$.

(39) For every state $s$ of $\mathbf{SCM}_{\mathrm{FSA}}$ and for every macro instruction $I$ such that $I$ is closed on $s$ holds $\text{insloc}(0) \in \text{dom} I$.

(40) For every state $s$ of $\mathbf{SCM}_{\mathrm{FSA}}$ and for every paraclosed macro instruction $I$ holds $\text{insloc}(0) \in \text{dom} I$.

(41) For every instruction $i$ of $\mathbf{SCM}_{\mathrm{FSA}}$ holds $\text{rng}\,\text{Macro}(i) = \{i, \mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}}\}$.

(42) Let $s_1$, $s_2$ be states of $\mathbf{SCM}_{\mathrm{FSA}}$ and $I$ be a macro instruction. Suppose $I$ is closed on $s_1$ and $I+\cdot\text{Start-At}(\text{insloc}(0)) \subseteq s_1$. Let $n$ be a natural number. Suppose $\text{ProgramPart}(\text{Relocated}(I,n)) \subseteq s_2$ and $\mathbf{IC}_{(s_2)} = \text{insloc}(n)$ and $s_1\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations}) = s_2\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations})$. Let $i$ be a natural number. Then $\mathbf{IC}_{(\text{Computation}(s_1))(i)} + n = \mathbf{IC}_{(\text{Computation}(s_2))(i)}$ and $\text{IncAddr}(\text{CurInstr}((\text{Computation}(s_1))(i)), n) = \text{CurInstr}((\text{Computation}(s_2))(i))$ and $(\text{Computation}(s_1))(i)\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations}) = (\text{Computation}(s_2))(i)\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations})$.

(43) Let $s_1$, $s_2$ be states of $\mathbf{SCM}_{\mathrm{FSA}}$ and $I$ be a macro instruction. Suppose $I$ is closed on $s_1$ and $I+\cdot\text{Start-At}(\text{insloc}(0)) \subseteq s_1$ and $I+\cdot\text{Start-At}(\text{insloc}(0)) \subseteq s_2$ and $s_1\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations}) = s_2\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations})$. Let $i$ be a natural number. Then $\mathbf{IC}_{(\text{Computation}(s_1))(i)} = \mathbf{IC}_{(\text{Computation}(s_2))(i)}$ and $\text{CurInstr}((\text{Computation}(s_1))(i)) = \text{CurInstr}((\text{Computation}(s_2))(i))$ and $(\text{Computation}(s_1))(i)\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations}) = (\text{Computation}(s_2))(i)\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations})$.

(44) Let $s_1$, $s_2$ be states of $\mathbf{SCM}_{\mathrm{FSA}}$ and $I$ be a macro instruction. Suppose $I$ is closed on $s_1$ and halting on $s_1$ and $I+\cdot\text{Start-At}(\text{insloc}(0)) \subseteq s_1$ and $I+\cdot\text{Start-At}(\text{insloc}(0)) \subseteq s_2$ and $s_1\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations}) = s_2\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations})$. Then $\text{LifeSpan}(s_1) = \text{LifeSpan}(s_2)$.

(45) Let $s_1$, $s_2$ be states of $\mathbf{SCM}_{\mathrm{FSA}}$ and $I$ be a macro instruction. Suppose that

(i) $s_1(\text{intloc}(0)) = 1$,

(ii) $I$ is closed on $s_1$ and halting on $s_1$,

(iii) for every read-write integer location $a$ holds $s_1(a) = s_2(a)$, and

(iv) for every finite sequence location $f$ holds $s_1(f) = s_2(f)$.

Then $\text{IExec}(I,s_1)\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations}) = \text{IExec}(I,s_2)\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations})$.

(46) Let $s_1$, $s_2$ be states of $\mathbf{SCM}_{\mathrm{FSA}}$ and $I$ be a macro instruction. Suppose $s_1(\text{intloc}(0)) = 1$ and $I$ is closed on $s_1$ and halting on $s_1$ and $s_1\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations}) = s_2\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations})$. Then $\text{IExec}(I,s_1)\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations}) = \text{IExec}(I,s_2)\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations})$.

Let $I$ be a macro instruction. One can check that $\text{Initialized}(I)$ is initial.

Next we state a number of propositions:

(47) Let $s$ be a state of $\mathbf{SCM}_{\mathrm{FSA}}$ and $I$ be a macro instruction. Then $\text{Initialized}(I)$ is pseudo-closed on $s$ if and only if $I$ is pseudo-closed on $\text{Initialize}(s)$.

(48) For every state $s$ of $\mathbf{SCM}_{\mathrm{FSA}}$ and for every macro instruction $I$ such that $I$ is pseudo-closed on Initialize($s$) holds $\mathrm{pseudo}-\mathrm{LifeSpan}(s,\mathrm{Initialized}(I)) = \mathrm{pseudo}-\mathrm{LifeSpan}(\mathrm{Initialize}(s),I)$.

(49) For every state $s$ of $\mathbf{SCM}_{\mathrm{FSA}}$ and for every macro instruction $I$ such that Initialized($I$) is pseudo-closed on $s$ holds $\mathrm{pseudo}-\mathrm{LifeSpan}(s,\mathrm{Initialized}(I)) = \mathrm{pseudo}-\mathrm{LifeSpan}(\mathrm{Initialize}(s),I)$.

(50) Let $s$ be a state of $\mathbf{SCM}_{\mathrm{FSA}}$ and $I$ be an initial finite partial state of $\mathbf{SCM}_{\mathrm{FSA}}$. Suppose $I$ is pseudo-closed on $s$. Then $I$ is pseudo-closed on $s{+\cdot}(I{+\cdot}\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))$ and $\mathrm{pseudo}-\mathrm{LifeSpan}(s,I) = \mathrm{pseudo}-\mathrm{LifeSpan}(s{+\cdot}(I{+\cdot}\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0))),I)$.

(51) Let $s_1$, $s_2$ be states of $\mathbf{SCM}_{\mathrm{FSA}}$ and $I$ be a macro instruction. Suppose $I{+\cdot}\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0)) \subseteq s_1$ and $I$ is pseudo-closed on $s_1$. Let $n$ be a natural number. Suppose $\mathrm{ProgramPart}(\mathrm{Relocated}(I,n)) \subseteq s_2$ and $\mathbf{IC}_{(s_2)} = \mathrm{insloc}(n)$ and $s_1\restriction(\mathrm{Int\text{-}Locations}\cup\mathrm{FinSeq\text{-}Locations}) = s_2\restriction(\mathrm{Int\text{-}Locations}\cup\mathrm{FinSeq\text{-}Locations})$. Then

(i) for every natural number $i$ such that $i < \mathrm{pseudo}-\mathrm{LifeSpan}(s_1,I)$ holds $\mathrm{IncAddr}(\mathrm{CurInstr}((\mathrm{Computation}(s_1))(i)),n) = \mathrm{CurInstr}((\mathrm{Computation}(s_2))(i))$, and

(ii) for every natural number $i$ such that $i \leq \mathrm{pseudo}-\mathrm{LifeSpan}(s_1,I)$ holds $\mathbf{IC}_{(\mathrm{Computation}(s_1))(i)} + n = \mathbf{IC}_{(\mathrm{Computation}(s_2))(i)}$ and $(\mathrm{Computation}(s_1))(i)\restriction(\mathrm{Int\text{-}Locations}\cup\mathrm{FinSeq\text{-}Locations}) = (\mathrm{Computation}(s_2))(i)\restriction(\mathrm{Int\text{-}Locations}\cup\mathrm{FinSeq\text{-}Locations})$.

(52) Let $s_1$, $s_2$ be states of $\mathbf{SCM}_{\mathrm{FSA}}$ and $I$ be a macro instruction. Suppose $s_1\restriction(\mathrm{Int\text{-}Locations}\cup\mathrm{FinSeq\text{-}Locations}) = s_2\restriction(\mathrm{Int\text{-}Locations}\cup\mathrm{FinSeq\text{-}Locations})$. If $I$ is pseudo-closed on $s_1$, then $I$ is pseudo-closed on $s_2$.

(53) Let $s$ be a state of $\mathbf{SCM}_{\mathrm{FSA}}$ and $I$ be a macro instruction. Suppose $s(\mathrm{intloc}(0)) = 1$. Then $I$ is pseudo-closed on $s$ if and only if $I$ is pseudo-closed on Initialize($s$).

(54) Let $a$ be an integer location and $I$, $J$ be macro instructions. Then $\mathrm{insloc}(0) \in \mathrm{dom}(\mathbf{if}\ a = 0\ \mathbf{then}\ I\ \mathbf{else}\ J)$ and $\mathrm{insloc}(1) \in \mathrm{dom}(\mathbf{if}\ a = 0\ \mathbf{then}\ I\ \mathbf{else}\ J)$ and $\mathrm{insloc}(0) \in \mathrm{dom}(\mathbf{if}\ a > 0\ \mathbf{then}\ I\ \mathbf{else}\ J)$ and $\mathrm{insloc}(1) \in \mathrm{dom}(\mathbf{if}\ a > 0\ \mathbf{then}\ I\ \mathbf{else}\ J)$.

(55) Let $a$ be an integer location and $I$, $J$ be macro instructions. Then $(\mathbf{if}\ a = 0\ \mathbf{then}\ I\ \mathbf{else}\ J)(\mathrm{insloc}(0)) = \mathbf{if}\ a = 0\ \mathbf{goto}\ \mathrm{insloc}(\mathrm{card}J+3)$ and $(\mathbf{if}\ a = 0\ \mathbf{then}\ I\ \mathbf{else}\ J)(\mathrm{insloc}(1)) = \mathrm{goto}\ \mathrm{insloc}(2)$ and $(\mathbf{if}\ a > 0\ \mathbf{then}\ I\ \mathbf{else}\ J)(\mathrm{insloc}(0)) = \mathbf{if}\ a > 0\ \mathbf{goto}\ \mathrm{insloc}(\mathrm{card}J+3)$ and $(\mathbf{if}\ a > 0\ \mathbf{then}\ I\ \mathbf{else}\ J)(\mathrm{insloc}(1)) = \mathrm{goto}\ \mathrm{insloc}(2)$.

(56) Let $a$ be an integer location, $I$, $J$ be macro instructions, and $n$ be a natural number. If $n < \mathrm{card}I + \mathrm{card}J + 3$, then $\mathrm{insloc}(n) \in \mathrm{dom}(\mathbf{if}\ a = 0\ \mathbf{then}\ I\ \mathbf{else}\ J)$ and $(\mathbf{if}\ a = 0\ \mathbf{then}\ I\ \mathbf{else}\ J)(\mathrm{insloc}(n)) \neq \mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}}$.

(57) Let $a$ be an integer location, $I$, $J$ be macro instructions, and $n$ be a natural number. If $n < \mathrm{card}I + \mathrm{card}J + 3$, then $\mathrm{insloc}(n) \in \mathrm{dom}(\mathbf{if}\ a > 0\ \mathbf{then}\ I\ \mathbf{else}\ J)$ and $(\mathbf{if}\ a > 0\ \mathbf{then}\ I\ \mathbf{else}\ J)(\mathrm{insloc}(n)) \neq \mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}}$.

(58) Let $s$ be a state of $\mathbf{SCM}_{\mathrm{FSA}}$ and $I$ be a macro instruction. Suppose Directed($I$) is pseudo-closed on $s$. Then

(i) $I;\ \mathrm{Stop}_{\mathrm{SCM}_{\mathrm{FSA}}}$ is closed on $s$,

(ii) $I;\ \mathrm{Stop}_{\mathrm{SCM}_{\mathrm{FSA}}}$ is halting on $s$,

(iii) $\mathrm{LifeSpan}(s{+\cdot}((I;\ \mathrm{Stop}_{\mathrm{SCM}_{\mathrm{FSA}}}){+\cdot}\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))) = \mathrm{pseudo}-\mathrm{LifeSpan}(s,\mathrm{Directed}(I))$,

(iv) for every natural number $n$ such that $n < \mathrm{pseudo}-\mathrm{LifeSpan}(s,\mathrm{Directed}(I))$ holds $\mathbf{IC}_{(\mathrm{Computation}(s{+\cdot}(I{+\cdot}\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(n)} = \mathbf{IC}_{(\mathrm{Computation}(s{+\cdot}((I;\ \mathrm{Stop}_{\mathrm{SCM}_{\mathrm{FSA}}}){+\cdot}\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(n)}$, and

(v) for every natural number $n$ such that $n \leq \mathrm{pseudo}-\mathrm{LifeSpan}(s,\mathrm{Directed}(I))$ holds $(\mathrm{Computation}(s{+\cdot}(I{+\cdot}\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(n)\restriction D = (\mathrm{Computation}(s{+\cdot}((I;\ \mathrm{Stop}_{\mathrm{SCM}_{\mathrm{FSA}}}){+\cdot}\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(n)\restriction D$,

where $D = \mathrm{Int\text{-}Locations}\cup\mathrm{FinSeq\text{-}Locations}$.

(59) Let $s$ be a state of $\mathbf{SCM}_{\rm FSA}$ and $I$ be a macro instruction. If $\mathrm{Directed}(I)$ is pseudo-closed on $s$, then $\mathrm{Result}(s+\cdot((I;\mathrm{Stop}_{\mathrm{SCM}_{\rm FSA}})+\cdot\mathrm{Start\text{-}At}(\mathrm{insloc}(0))))\restriction D = (\mathrm{Computation}(s+\cdot(I+\cdot\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(\mathrm{pseudo}-\mathrm{LifeSpan}(s,\mathrm{Directed}(I)))\restriction D$, where $D = \text{Int-Locations}\cup\text{FinSeq-Locations}$.

(60) Let $s$ be a state of $\mathbf{SCM}_{\rm FSA}$ and $I$ be a macro instruction. If $s(\mathrm{intloc}(0)) = 1$ and $\mathrm{Directed}(I)$ is pseudo-closed on $s$, then $\mathrm{IExec}(I;\mathrm{Stop}_{\mathrm{SCM}_{\rm FSA}},s)\restriction D = (\mathrm{Computation}(s+\cdot(I+\cdot\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(\mathrm{pseudo}-\mathrm{LifeSpan}(s,\mathrm{Directed}(I)))\restriction D$, where $D = \text{Int-Locations}\cup\text{FinSeq-Locations}$.

(61) For all macro instructions $I$, $J$ and for every integer location $a$ holds $(\mathbf{if}\ a = 0\ \mathbf{then}\ I\ \mathbf{else}\ J)(\mathrm{insloc}(\mathrm{card}\,I+\mathrm{card}\,J+3)) = \mathbf{halt}_{\mathbf{SCM}_{\rm FSA}}$.

(62) For all macro instructions $I$, $J$ and for every integer location $a$ holds $(\mathbf{if}\ a > 0\ \mathbf{then}\ I\ \mathbf{else}\ J)(\mathrm{insloc}(\mathrm{card}\,I+\mathrm{card}\,J+3)) = \mathbf{halt}_{\mathbf{SCM}_{\rm FSA}}$.

(63) For all macro instructions $I$, $J$ and for every integer location $a$ holds $(\mathbf{if}\ a = 0\ \mathbf{then}\ I\ \mathbf{else}\ J)(\mathrm{insloc}(\mathrm{card}\,J+2)) = \mathrm{goto}\ \mathrm{insloc}(\mathrm{card}\,I+\mathrm{card}\,J+3)$.

(64) For all macro instructions $I$, $J$ and for every integer location $a$ holds $(\mathbf{if}\ a > 0\ \mathbf{then}\ I\ \mathbf{else}\ J)(\mathrm{insloc}(\mathrm{card}\,J+2)) = \mathrm{goto}\ \mathrm{insloc}(\mathrm{card}\,I+\mathrm{card}\,J+3)$.

(65) For every macro instruction $J$ and for every integer location $a$ holds $(\mathbf{if}\ a = 0\ \mathbf{then}\ \mathrm{Goto}(\mathrm{insloc}(2))\ \mathbf{else}\ J)(\mathrm{insloc}(\mathrm{card}\,J+3)) = \mathrm{goto}\ \mathrm{insloc}(\mathrm{card}\,J+5)$.

(66) Let $s$ be a state of $\mathbf{SCM}_{\rm FSA}$, $I$, $J$ be macro instructions, and $a$ be a read-write integer location. Suppose $s(a) = 0$ and $\mathrm{Directed}(I)$ is pseudo-closed on $s$. Then $\mathbf{if}\ a = 0\ \mathbf{then}\ I\ \mathbf{else}\ J$ is halting on $s$ and $\mathbf{if}\ a = 0\ \mathbf{then}\ I\ \mathbf{else}\ J$ is closed on $s$ and $\mathrm{LifeSpan}(s+\cdot((\mathbf{if}\ a = 0\ \mathbf{then}\ I\ \mathbf{else}\ J)+\cdot\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))) = \mathrm{LifeSpan}(s+\cdot((I;\mathrm{Stop}_{\mathrm{SCM}_{\rm FSA}})+\cdot\mathrm{Start\text{-}At}(\mathrm{insloc}(0))))+1$.

(67) Let $s$ be a state of $\mathbf{SCM}_{\rm FSA}$, $I$, $J$ be macro instructions, and $a$ be a read-write integer location. Suppose $s(\mathrm{intloc}(0)) = 1$ and $s(a) = 0$ and $\mathrm{Directed}(I)$ is pseudo-closed on $s$. Then $\mathrm{IExec}(\mathbf{if}\ a = 0\ \mathbf{then}\ I\ \mathbf{else}\ J,s)\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations}) = \mathrm{IExec}(I;\mathrm{Stop}_{\mathrm{SCM}_{\rm FSA}},s)\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations})$.

(68) Let $s$ be a state of $\mathbf{SCM}_{\rm FSA}$, $I$, $J$ be macro instructions, and $a$ be a read-write integer location. Suppose $s(a) > 0$ and $\mathrm{Directed}(I)$ is pseudo-closed on $s$. Then $\mathbf{if}\ a > 0\ \mathbf{then}\ I\ \mathbf{else}\ J$ is halting on $s$ and $\mathbf{if}\ a > 0\ \mathbf{then}\ I\ \mathbf{else}\ J$ is closed on $s$ and $\mathrm{LifeSpan}(s+\cdot((\mathbf{if}\ a > 0\ \mathbf{then}\ I\ \mathbf{else}\ J)+\cdot\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))) = \mathrm{LifeSpan}(s+\cdot((I;\mathrm{Stop}_{\mathrm{SCM}_{\rm FSA}})+\cdot\mathrm{Start\text{-}At}(\mathrm{insloc}(0))))+1$.

(69) Let $s$ be a state of $\mathbf{SCM}_{\rm FSA}$, $I$, $J$ be macro instructions, and $a$ be a read-write integer location. Suppose $s(\mathrm{intloc}(0)) = 1$ and $s(a) > 0$ and $\mathrm{Directed}(I)$ is pseudo-closed on $s$. Then $\mathrm{IExec}(\mathbf{if}\ a > 0\ \mathbf{then}\ I\ \mathbf{else}\ J,s)\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations}) = \mathrm{IExec}(I;\mathrm{Stop}_{\mathrm{SCM}_{\rm FSA}},s)\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations})$.

(70) Let $s$ be a state of $\mathbf{SCM}_{\rm FSA}$, $I$, $J$ be macro instructions, and $a$ be a read-write integer location. Suppose $s(a) \neq 0$ and $\mathrm{Directed}(J)$ is pseudo-closed on $s$. Then $\mathbf{if}\ a = 0\ \mathbf{then}\ I\ \mathbf{else}\ J$ is halting on $s$ and $\mathbf{if}\ a = 0\ \mathbf{then}\ I\ \mathbf{else}\ J$ is closed on $s$ and $\mathrm{LifeSpan}(s+\cdot((\mathbf{if}\ a = 0\ \mathbf{then}\ I\ \mathbf{else}\ J)+\cdot\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))) = \mathrm{LifeSpan}(s+\cdot((J;\mathrm{Stop}_{\mathrm{SCM}_{\rm FSA}})+\cdot\mathrm{Start\text{-}At}(\mathrm{insloc}(0))))+3$.

(71) Let $s$ be a state of $\mathbf{SCM}_{\rm FSA}$, $I$, $J$ be macro instructions, and $a$ be a read-write integer location. Suppose $s(\mathrm{intloc}(0)) = 1$ and $s(a) \neq 0$ and $\mathrm{Directed}(J)$ is pseudo-closed on $s$. Then $\mathrm{IExec}(\mathbf{if}\ a = 0\ \mathbf{then}\ I\ \mathbf{else}\ J,s)\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations}) = \mathrm{IExec}(J;\mathrm{Stop}_{\mathrm{SCM}_{\rm FSA}},s)\restriction(\text{Int-Locations}\cup\text{FinSeq-Locations})$.

(72) Let $s$ be a state of $\mathbf{SCM}_{\rm FSA}$, $I$, $J$ be macro instructions, and $a$ be a read-write integer location. Suppose $s(a) \leq 0$ and $\mathrm{Directed}(J)$ is pseudo-closed on $s$. Then $\mathbf{if}\ a > 0\ \mathbf{then}\ I\ \mathbf{else}\ J$ is halting on $s$ and $\mathbf{if}\ a > 0\ \mathbf{then}\ I\ \mathbf{else}\ J$ is closed on $s$ and $\mathrm{LifeSpan}(s+\cdot((\mathbf{if}\ a > 0\ \mathbf{then}\ I\ \mathbf{else}\ J)+\cdot\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))) = \mathrm{LifeSpan}(s+\cdot((J;\mathrm{Stop}_{\mathrm{SCM}_{\rm FSA}})+\cdot\mathrm{Start\text{-}At}(\mathrm{insloc}(0))))+3$.

(73) Let $s$ be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, $I$, $J$ be macro instructions, and $a$ be a read-write integer location. Suppose $s(\mathrm{intloc}(0)) = 1$ and $s(a) \leq 0$ and Directed($J$) is pseudo-closed on $s$. Then IExec(**if** $a > 0$ **then** $I$ **else** $J, s$)$\upharpoonright$(Int-Locations$\cup$FinSeq-Locations) $=$ IExec($J$; $\mathrm{Stop}_{\mathbf{SCM}_{\mathrm{FSA}}}, s$)$\upharpoonright$(Int-Locations$\cup$FinSeq-Locations).

(74) Let $s$ be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, $I$, $J$ be macro instructions, and $a$ be a read-write integer location. Suppose Directed($I$) is pseudo-closed on $s$ and Directed($J$) is pseudo-closed on $s$. Then **if** $a = 0$ **then** $I$ **else** $J$ is closed on $s$ and **if** $a = 0$ **then** $I$ **else** $J$ is halting on $s$.

(75) Let $s$ be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, $I$, $J$ be macro instructions, and $a$ be a read-write integer location. Suppose Directed($I$) is pseudo-closed on $s$ and Directed($J$) is pseudo-closed on $s$. Then **if** $a > 0$ **then** $I$ **else** $J$ is closed on $s$ and **if** $a > 0$ **then** $I$ **else** $J$ is halting on $s$.

(76) Let $I$ be a macro instruction and $a$ be an integer location. If $I$ does not destroy $a$, then Directed($I$) does not destroy $a$.

(77) Let $i$ be an instruction of $\mathbf{SCM}_{\mathrm{FSA}}$ and $a$ be an integer location. If $i$ does not destroy $a$, then Macro($i$) does not destroy $a$.

(78) For every integer location $a$ holds $\mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}}$ does not refer $a$.

(79) For all integer locations $a$, $b$, $c$ such that $a \neq b$ holds AddTo($c, b$) does not refer $a$.

(80) Let $i$ be an instruction of $\mathbf{SCM}_{\mathrm{FSA}}$ and $a$ be an integer location. If $i$ does not refer $a$, then Macro($i$) does not refer $a$.

(81) Let $I$, $J$ be macro instructions and $a$ be an integer location. Suppose $I$ does not destroy $a$ and $J$ does not destroy $a$. Then $I$; $J$ does not destroy $a$.

(82) Let $J$ be a macro instruction, $i$ be an instruction of $\mathbf{SCM}_{\mathrm{FSA}}$, and $a$ be an integer location. Suppose $i$ does not destroy $a$ and $J$ does not destroy $a$. Then $i$; $J$ does not destroy $a$.

(83) Let $I$ be a macro instruction, $j$ be an instruction of $\mathbf{SCM}_{\mathrm{FSA}}$, and $a$ be an integer location. Suppose $I$ does not destroy $a$ and $j$ does not destroy $a$. Then $I$; $j$ does not destroy $a$.

(84) Let $i$, $j$ be instructions of $\mathbf{SCM}_{\mathrm{FSA}}$ and $a$ be an integer location. Suppose $i$ does not destroy $a$ and $j$ does not destroy $a$. Then $i$; $j$ does not destroy $a$.

(85) For every integer location $a$ holds $\mathrm{Stop}_{\mathbf{SCM}_{\mathrm{FSA}}}$ does not destroy $a$.

(86) For every integer location $a$ and for every instruction-location $l$ of $\mathbf{SCM}_{\mathrm{FSA}}$ holds Goto($l$) does not destroy $a$.

(87) Let $s$ be a state of $\mathbf{SCM}_{\mathrm{FSA}}$ and $I$ be a macro instruction. Suppose $I$ is halting on Initialize($s$). Then

(i) for every read-write integer location $a$ holds $(\mathrm{IExec}(I,s))(a) = (\mathrm{Computation}(\mathrm{Initialize}(s)+\cdot(I+\cdot\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(\mathrm{LifeSpan}(\mathrm{Initialize}(s)+\cdot(I+\cdot\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(a)$, and

(ii) for every finite sequence location $f$ holds $(\mathrm{IExec}(I,s))(f) = (\mathrm{Computation}(\mathrm{Initialize}(s)+\cdot(I+\cdot\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(\mathrm{LifeSpan}(\mathrm{Initialize}(s)+\cdot(I+\cdot\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(f)$.

(88) Let $s$ be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, $I$ be a parahalting macro instruction, and $a$ be a read-write integer location. Then $(\mathrm{IExec}(I,s))(a) = (\mathrm{Computation}(\mathrm{Initialize}(s)+\cdot(I+\cdot\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(\mathrm{LifeSpan}(\mathrm{Initialize}(s)+\cdot(I+\cdot\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(a)$.

(89) Let $s$ be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, $I$ be a macro instruction, $a$ be an integer location, and $k$ be a natural number. Suppose $I$ is closed on Initialize($s$) and halting on Initialize($s$) and $I$ does not destroy $a$. Then $(\mathrm{IExec}(I,s))(a) = (\mathrm{Computation}(\mathrm{Initialize}(s)+\cdot(I+\cdot\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(k)(a)$.

(90) Let $s$ be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, $I$ be a parahalting macro instruction, $a$ be an integer location, and $k$ be a natural number. If $I$ does not destroy $a$, then $(\mathrm{IExec}(I,s))(a) = (\mathrm{Computation}(\mathrm{Initialize}(s)+\cdot(I+\cdot\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(k)(a)$.

(91) Let $s$ be a state of $\mathbf{SCM}_{\rm FSA}$, $I$ be a parahalting macro instruction, and $a$ be an integer location. If $I$ does not destroy $a$, then $({\rm IExec}(I,s))(a) = ({\rm Initialize}(s))(a)$.

(92) Let $s$ be a state of $\mathbf{SCM}_{\rm FSA}$ and $I$ be a keeping 0 macro instruction. Suppose $I$ is halting on ${\rm Initialize}(s)$. Then $({\rm IExec}(I,s))({\rm intloc}(0)) = 1$ and for every natural number $k$ holds $({\rm Computation}({\rm Initialize}(s){+\cdot}(I{+\cdot}\,{\rm Start\text{-}At}({\rm insloc}(0)))))(k)({\rm intloc}(0)) = 1$.

(93) Let $s$ be a state of $\mathbf{SCM}_{\rm FSA}$, $I$ be a macro instruction, and $a$ be an integer location. Suppose $I$ does not destroy $a$. Let $k$ be a natural number. If $\mathbf{IC}_{({\rm Computation}(s{+\cdot}(I{+\cdot}\,{\rm Start\text{-}At}({\rm insloc}(0)))))(k)} \in {\rm dom}\,I$, then $({\rm Computation}(s{+\cdot}(I{+\cdot}\,{\rm Start\text{-}At}({\rm insloc}(0)))))(k+1)(a) = ({\rm Computation}(s{+\cdot}(I{+\cdot}\,{\rm Start\text{-}At}({\rm insloc}(0)))))(k)(a)$.

(94) Let $s$ be a state of $\mathbf{SCM}_{\rm FSA}$, $I$ be a macro instruction, and $a$ be an integer location. Suppose $I$ does not destroy $a$. Let $m$ be a natural number. Suppose that for every natural number $n$ such that $n < m$ holds $\mathbf{IC}_{({\rm Computation}(s{+\cdot}(I{+\cdot}\,{\rm Start\text{-}At}({\rm insloc}(0)))))(n)} \in {\rm dom}\,I$. Let $n$ be a natural number. If $n \leq m$, then $({\rm Computation}(s{+\cdot}(I{+\cdot}\,{\rm Start\text{-}At}({\rm insloc}(0)))))(n)(a) = s(a)$.

(95) Let $s$ be a state of $\mathbf{SCM}_{\rm FSA}$, $I$ be a good macro instruction, and $m$ be a natural number. Suppose that for every natural number $n$ such that $n < m$ holds $\mathbf{IC}_{({\rm Computation}(s{+\cdot}(I{+\cdot}\,{\rm Start\text{-}At}({\rm insloc}(0)))))(n)} \in {\rm dom}\,I$. Let $n$ be a natural number. If $n \leq m$, then $({\rm Computation}(s{+\cdot}(I{+\cdot}\,{\rm Start\text{-}At}({\rm insloc}(0)))))(n)({\rm intloc}(0)) = s({\rm intloc}(0))$.

(96) Let $s$ be a state of $\mathbf{SCM}_{\rm FSA}$ and $I$ be a good macro instruction. Suppose $I$ is halting on ${\rm Initialize}(s)$ and closed on ${\rm Initialize}(s)$. Then $({\rm IExec}(I,s))({\rm intloc}(0)) = 1$ and for every natural number $k$ holds $({\rm Computation}({\rm Initialize}(s){+\cdot}(I{+\cdot}\,{\rm Start\text{-}At}({\rm insloc}(0)))))(k)({\rm intloc}(0)) = 1$.

(97) Let $s$ be a state of $\mathbf{SCM}_{\rm FSA}$ and $I$ be a good macro instruction. Suppose $I$ is closed on $s$. Let $k$ be a natural number. Then $({\rm Computation}(s{+\cdot}(I{+\cdot}\,{\rm Start\text{-}At}({\rm insloc}(0)))))(k)({\rm intloc}(0)) = s({\rm intloc}(0))$.

(98) Let $s$ be a state of $\mathbf{SCM}_{\rm FSA}$, $I$ be a keeping 0 parahalting macro instruction, and $a$ be a read-write integer location. Suppose $I$ does not destroy $a$. Then $({\rm Computation}({\rm Initialize}(s){+\cdot}((I;\,{\rm SubFrom}(a,{\rm intloc}(0))){+\cdot}\,{\rm Start\text{-}At}({\rm insloc}(0)))))({\rm LifeSpan}({\rm Initialize}(s){+\cdot}((I;\,{\rm SubFrom}(a,{\rm intloc}(0))){+\cdot}\,{\rm Start\text{-}At}({\rm insloc}(0)))))(a) = s(a) - 1$.

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

(100) Let $s_1$, $s_2$ be states of $\mathbf{SCM}_{\rm FSA}$ and $I$ be a macro instruction. Suppose $I$ is closed on $s_1$ and halting on $s_1$ and $s_1 \restriction D = s_2 \restriction D$. Let $k$ be a natural number. Then

(i) $({\rm Computation}(s_1{+\cdot}(I{+\cdot}\,{\rm Start\text{-}At}({\rm insloc}(0)))))(k)$ and $({\rm Computation}(s_2{+\cdot}(I{+\cdot}\,{\rm Start\text{-}At}({\rm insloc}(0)))))(k)$ are equal outside the instruction locations of $\mathbf{SCM}_{\rm FSA}$, and

(ii) ${\rm CurInstr}(({\rm Computation}(s_1{+\cdot}(I{+\cdot}\,{\rm Start\text{-}At}({\rm insloc}(0)))))(k)) = {\rm CurInstr}(({\rm Computation}(s_2{+\cdot}(I{+\cdot}\,{\rm Start\text{-}At}({\rm insloc}(0)))))(k))$,

where $D = {\rm Int\text{-}Locations} \cup {\rm FinSeq\text{-}Locations}$.

(101) Let $s_1$, $s_2$ be states of $\mathbf{SCM}_{\rm FSA}$ and $I$ be a macro instruction. Suppose $I$ is closed on $s_1$ and halting on $s_1$ and $s_1 \restriction D = s_2 \restriction D$. Then ${\rm LifeSpan}(s_1{+\cdot}(I{+\cdot}\,{\rm Start\text{-}At}({\rm insloc}(0)))) = {\rm LifeSpan}(s_2{+\cdot}(I{+\cdot}\,{\rm Start\text{-}At}({\rm insloc}(0))))$ and ${\rm Result}(s_1{+\cdot}(I{+\cdot}\,{\rm Start\text{-}At}({\rm insloc}(0))))$ and ${\rm Result}(s_2{+\cdot}(I{+\cdot}\,{\rm Start\text{-}At}({\rm insloc}(0))))$ are equal outside the instruction locations of $\mathbf{SCM}_{\rm FSA}$, where $D = {\rm Int\text{-}Locations} \cup {\rm FinSeq\text{-}Locations}$.

(103)[3] Let $s_1$, $s_2$ be states of $\mathbf{SCM}_{\rm FSA}$ and $I$ be a macro instruction. Suppose that

(i) $I$ is closed on $s_1$ and halting on $s_1$,

(ii) $I{+\cdot}\,{\rm Start\text{-}At}({\rm insloc}(0)) \subseteq s_1$,

(iii) $I{+\cdot}\,{\rm Start\text{-}At}({\rm insloc}(0)) \subseteq s_2$, and

(iv) there exists a natural number $k$ such that $({\rm Computation}(s_1))(k)$ and $s_2$ are equal outside the instruction locations of $\mathbf{SCM}_{\rm FSA}$.

Then ${\rm Result}(s_1)$ and ${\rm Result}(s_2)$ are equal outside the instruction locations of $\mathbf{SCM}_{\rm FSA}$.

[3] The proposition (102) has been removed.

## 2. THE loop MACROINSTRUCTION

Let $I$ be a macro instruction and let $k$ be a natural number. One can check that $\mathrm{IncAddr}(I,k)$ is initial and programmed.

Let $I$ be a macro instruction. The functor loop$I$ yielding a halt-free macro instruction is defined by:

(Def. 4)[4] $\mathrm{loop}\,I = (\mathrm{id}_{\text{the instructions of }\mathbf{SCM}_{\mathrm{FSA}}} +\cdot (\mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}} \dot{\longmapsto} \text{goto insloc}(0))) \cdot I$.

One can prove the following two propositions:

(104) For every macro instruction $I$ holds $\mathrm{loop}\,I = \mathrm{Directed}(I, \mathrm{insloc}(0))$.

(105) Let $I$ be a macro instruction and $a$ be an integer location. If $I$ does not destroy $a$, then loop$I$ does not destroy $a$.

Let $I$ be a good macro instruction. Note that loop$I$ is good.

We now state several propositions:

(106) For every macro instruction $I$ holds $\mathrm{dom}\,\mathrm{loop}\,I = \mathrm{dom}\,I$.

(107) For every macro instruction $I$ holds $\mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}} \notin \mathrm{rng}\,\mathrm{loop}\,I$.

(108) For every macro instruction $I$ and for every set $x$ such that $I(x) \neq \mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}}$ holds $(\mathrm{loop}\,I)(x) = I(x)$.

(109) Let $s$ be a state of $\mathbf{SCM}_{\mathrm{FSA}}$ and $I$ be a macro instruction. Suppose $I$ is closed on $s$ and halting on $s$. Let $m$ be a natural number. Suppose $m \leq \mathrm{LifeSpan}(s+\cdot(I+\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0))))$. Then $(\mathrm{Computation}(s+\cdot(I+\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(m)$ and $(\mathrm{Computation}(s+\cdot(\mathrm{loop}\,I+\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(m)$ are equal outside the instruction locations of $\mathbf{SCM}_{\mathrm{FSA}}$.

(110) Let $s$ be a state of $\mathbf{SCM}_{\mathrm{FSA}}$ and $I$ be a macro instruction. Suppose $I$ is closed on $s$ and halting on $s$. Let $m$ be a natural number. If $m < \mathrm{LifeSpan}(s+\cdot(I+\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0))))$, then $\mathrm{CurInstr}((\mathrm{Computation}(s+\cdot(I+\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(m)) = \mathrm{CurInstr}((\mathrm{Computation}(s+\cdot(\mathrm{loop}\,I+\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(m))$.

(111) Let $s$ be a state of $\mathbf{SCM}_{\mathrm{FSA}}$ and $I$ be a macro instruction. Suppose $I$ is closed on $s$ and halting on $s$. Let $m$ be a natural number. If $m \leq \mathrm{LifeSpan}(s+\cdot(I+\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0))))$, then $\mathrm{CurInstr}((\mathrm{Computation}(s+\cdot(\mathrm{loop}\,I+\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(m)) \neq \mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}}$.

(112) Let $s$ be a state of $\mathbf{SCM}_{\mathrm{FSA}}$ and $I$ be a macro instruction. If $I$ is closed on $s$ and halting on $s$, then $\mathrm{CurInstr}((\mathrm{Computation}(s+\cdot(\mathrm{loop}\,I+\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(\mathrm{LifeSpan}(s+\cdot(I+\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))) = \text{goto insloc}(0)$.

(113) Let $s$ be a state of $\mathbf{SCM}_{\mathrm{FSA}}$ and $I$ be a paraclosed macro instruction. Suppose $I+\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)) \subseteq s$ and $s$ is halting. Let $m$ be a natural number. Suppose $m \leq \mathrm{LifeSpan}(s)$. Then $(\mathrm{Computation}(s))(m)$ and $(\mathrm{Computation}(s+\cdot \mathrm{loop}\,I))(m)$ are equal outside the instruction locations of $\mathbf{SCM}_{\mathrm{FSA}}$.

(114) Let $s$ be a state of $\mathbf{SCM}_{\mathrm{FSA}}$ and $I$ be a parahalting macro instruction. Suppose $\mathrm{Initialized}(I) \subseteq s$. Let $k$ be a natural number. If $k \leq \mathrm{LifeSpan}(s)$, then $\mathrm{CurInstr}((\mathrm{Computation}(s+\cdot \mathrm{loop}\,I))(k)) \neq \mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}}$.

## 3. THE Times MACROINSTRUCTION

Let $a$ be an integer location and let $I$ be a macro instruction. The functor $\mathrm{Times}(a,I)$ yields a macro instruction and is defined as follows:

(Def. 5) $\mathrm{Times}(a,I) = \mathbf{if}\ a > 0\ \mathbf{then}\ \mathrm{loop}\,\mathbf{if}\ a = 0\ \mathbf{then}\ \mathrm{Goto}(\mathrm{insloc}(2))\ \mathbf{else}\ (I; \mathrm{SubFrom}(a, \mathrm{intloc}(0)))\ \mathbf{else}\ (\mathrm{Stop}_{\mathbf{SCM}_{\mathrm{FSA}}})$.

[4] The definitions (Def. 1)–(Def. 3) have been removed.

Next we state a number of propositions:

(115) For every good macro instruction $I$ and for every read-write integer location $a$ holds **if** $a = 0$ **then** Goto(insloc(2)) **else** ($I$; SubFrom($a$, intloc(0))) is good.

(116) For all macro instructions $I$, $J$ and for every integer location $a$ holds (**if** $a = 0$ **then** Goto(insloc(2)) **else** ($I$; SubFrom($a$, intloc(0))))(insloc(card($I$; SubFrom($a$, intloc(0))) + 3)) = goto insloc(card($I$; SubFrom($a$, intloc(0))) + 5).

(117) Let $s$ be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, $I$ be a good parahalting macro instruction, and $a$ be a read-write integer location. Suppose $I$ does not destroy $a$ and $s(\mathrm{intloc}(0)) = 1$ and $s(a) > 0$. Then loop **if** $a = 0$ **then** Goto(insloc(2)) **else** ($I$; SubFrom($a$, intloc(0))) is pseudo-closed on $s$.

(118) Let $s$ be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, $I$ be a good parahalting macro instruction, and $a$ be a read-write integer location. Suppose $I$ does not destroy $a$ and $s(a) > 0$. Then Initialized(loop **if** $a = 0$ **then** Goto(insloc(2)) **else** ($I$; SubFrom($a$, intloc(0)))) is pseudo-closed on $s$.

(119) Let $s$ be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, $I$ be a good parahalting macro instruction, and $a$ be a read-write integer location. Suppose $I$ does not destroy $a$ and $s(\mathrm{intloc}(0)) = 1$. Then Times($a$, $I$) is closed on $s$ and Times($a$, $I$) is halting on $s$.

(120) Let $I$ be a good parahalting macro instruction and $a$ be a read-write integer location. If $I$ does not destroy $a$, then Initialized(Times($a$, $I$)) is halting.

(121) Let $I$, $J$ be macro instructions and $a$, $c$ be integer locations. Suppose $I$ does not destroy $c$ and $J$ does not destroy $c$. Then **if** $a = 0$ **then** $I$ **else** $J$ does not destroy $c$ and **if** $a > 0$ **then** $I$ **else** $J$ does not destroy $c$.

(122) Let $s$ be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, $I$ be a good parahalting macro instruction, and $a$ be a read-write integer location. Suppose $I$ does not destroy $a$ and $s(\mathrm{intloc}(0)) = 1$ and $s(a) > 0$. Then there exists a state $s_2$ of $\mathbf{SCM}_{\mathrm{FSA}}$ and there exists a natural number $k$ such that

$s_2 = s +\cdot$ (loop **if** $a = 0$ **then** Goto(insloc(2)) **else** ($I$; SubFrom($a$, intloc(0))) $+\cdot$ Start-At(insloc(0))) and $k$ = LifeSpan($s +\cdot$ ((**if** $a = 0$ **then** Goto(insloc(2)) **else** ($I$; SubFrom($a$, intloc(0)))) $+\cdot$ Start-At(insloc(0)))) + 1 and $(\mathrm{Computation}(s_2))(k)(a) = s(a) - 1$ and $(\mathrm{Computation}(s_2))(k)(\mathrm{intloc}(0)) = 1$ and for every read-write integer location $b$ such that $b \neq a$ holds $(\mathrm{Computation}(s_2))(k)(b) = (\mathrm{IExec}(I, s))(b)$ and for every finite sequence location $f$ holds $(\mathrm{Computation}(s_2))(k)(f) = (\mathrm{IExec}(I, s))(f)$ and $\mathbf{IC}_{(\mathrm{Computation}(s_2))(k)} = \mathrm{insloc}(0)$ and for every natural number $n$ such that $n \leq k$ holds $\mathbf{IC}_{(\mathrm{Computation}(s_2))(n)} \in \mathrm{dom}$ loop **if** $a = 0$ **then** Goto(insloc(2)) **else** ($I$; SubFrom($a$, intloc(0))).

(123) Let $s$ be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, $I$ be a good parahalting macro instruction, and $a$ be a read-write integer location. If $s(\mathrm{intloc}(0)) = 1$ and $s(a) \leq 0$, then $\mathrm{IExec}(\mathrm{Times}(a, I), s) \restriction (\text{Int-Locations} \cup \text{FinSeq-Locations}) = s \restriction (\text{Int-Locations} \cup \text{FinSeq-Locations})$.

(124) Let $s$ be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, $I$ be a good parahalting macro instruction, and $a$ be a read-write integer location. Suppose $I$ does not destroy $a$ and $s(a) > 0$. Then $(\mathrm{IExec}(I; \mathrm{SubFrom}(a, \mathrm{intloc}(0)), s))(a) = s(a) - 1$ and $\mathrm{IExec}(\mathrm{Times}(a, I), s) \restriction (\text{Int-Locations} \cup \text{FinSeq-Locations}) = \mathrm{IExec}(\mathrm{Times}(a, I), \mathrm{IExec}(I; \mathrm{SubFrom}(a, \mathrm{intloc}(0)), s)) \restriction (\text{Int-Locations} \cup \text{FinSeq-Locations})$.

## 4. An Example

We now state the proposition

(125) Let $s$ be a state of $\mathbf{SCM}_{\mathrm{FSA}}$ and $a$, $b$, $c$ be read-write integer locations. If $a \neq b$ and $a \neq c$ and $b \neq c$ and $s(a) \geq 0$, then $(\mathrm{IExec}(\mathrm{Times}(a, \mathrm{Macro}(\mathrm{AddTo}(b, c))), s))(b) = s(b) + s(c) \cdot s(a)$.

## 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] Noriko Asamoto, Yatsuka Nakamura, Piotr Rudnicki, and Andrzej Trybulec. On the composition of macro instructions. Part III. *Journal of Formalized Mathematics*, 8, 1996. http://mizar.org/JFM/Vol8/scmfsa6c.html.

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

[7] Grzegorz Bancerek. The fundamental properties of natural numbers. *Journal of Formalized Mathematics*, 1, 1989. http://mizar.org/JFM/Vol1/nat_1.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] Grzegorz Bancerek and Andrzej Trybulec. Miscellaneous facts about functions. *Journal of Formalized Mathematics*, 8, 1996. http://mizar.org/JFM/Vol8/funct_7.html.

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

[11] Czesław Byliński. A classical first order language. *Journal of Formalized Mathematics*, 2, 1990. http://mizar.org/JFM/Vol2/cqc_lang.html.

[12] 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.

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

[14] Takaya Nishiyama and Yasuho Mizuhara. Binary arithmetics. *Journal of Formalized Mathematics*, 5, 1993. http://mizar.org/JFM/Vol5/binarith.html.

[15] 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.

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

[17] Andrzej Trybulec. Enumerated sets. *Journal of Formalized Mathematics*, 1, 1989. http://mizar.org/JFM/Vol1/enumset1.html.

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

[19] 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.

[20] 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.

[21] Andrzej Trybulec and Yatsuka Nakamura. Relocability for $\mathbf{SCM}_{\mathrm{FSA}}$. *Journal of Formalized Mathematics*, 8, 1996. http://mizar.org/JFM/Vol8/scmfsa_5.html.

[22] 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.

[23] 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.

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


*Received October 29, 1997*

*Published January 2, 2004*