FORMALIZED MATHEMATICS
Volume 6, Number 4, 1997
University of Białystok

The loop and Times Macroinstruction for $\mathbf{SCM}_{\mathrm{FSA}}$

Noriko Asamoto
Ochanomizu University
Tokyo

Summary. We implement two macroinstructions `loop` and `Times` which iterate macroinstructions of $\mathbf{SCM}_{\mathrm{FSA}}$. 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`.

The articles [22], [29], [16], [8], [12], [30], [13], [14], [11], [7], [9], [28], [15], [17], [23], [20], [21], [27], [24], [25], [1], [10], [19], [26], [5], [6], [4], [2], [3], and [18] provide the terminology and notation for this paper.

1. Preliminaries

Let s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$ and let P be an initial finite partial state of $\mathbf{SCM}_{\mathrm{FSA}}$. We say that P is pseudo-closed on s if and only if the condition (Def. 1) is satisfied.

(Def. 1) There exists a natural number k such that $\mathbf{IC}_{(\mathrm{Computation}(s+\cdot(P+\cdot\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(k)} = \mathrm{insloc}(\mathrm{card}\,\mathrm{ProgramPart}(P))$ and for every natural number n such that $n < k$ holds $\mathbf{IC}_{(\mathrm{Computation}(s+\cdot(P+\cdot\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(n)} \in \mathrm{dom}\,P$.

Let P be an initial finite partial state of $\mathbf{SCM}_{\mathrm{FSA}}$. We say that P is pseudo-paraclosed if and only if:

(Def. 2) For every state s of $\mathbf{SCM}_{\mathrm{FSA}}$ holds P is pseudo-closed on s.

ISSN 1426–2630

Let us note that there exists a macro instruction which is pseudo-paraclosed.

Let s be a state of $\mathbf{SCM}_{\rm FSA}$ and let P be an initial finite partial state of $\mathbf{SCM}_{\rm FSA}$. Let us assume that P is pseudo-closed on s.

The functor $\mathrm{pseudo} - \mathrm{LifeSpan}(s, P)$ yielding a natural number is defined as follows:

(Def. 3) $\mathbf{IC}_{(\mathrm{Computation}(s+\cdot(P+\cdot\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(\mathrm{pseudo}-\mathrm{LifeSpan}(s,P))} = \mathrm{insloc}(\mathrm{card}\,\mathrm{ProgramPart}(P))$ and for every natural number n such that $\mathbf{IC}_{(\mathrm{Computation}(s+\cdot(P+\cdot\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(n)} \notin \mathrm{dom}\,P$ holds $\mathrm{pseudo} - \mathrm{LifeSpan}(s, P) \leqslant n$.

We now state a number of propositions:

(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 n be a natural number. If $n < \mathrm{pseudo} - \mathrm{LifeSpan}(s, P)$, then $\mathbf{IC}_{(\mathrm{Computation}(s+\cdot(P+\cdot\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(n)} \in \mathrm{dom}\,P$ and $\mathrm{CurInstr}((\mathrm{Computation}(s+\cdot(P+\cdot\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(n)) \neq \mathbf{halt}_{\mathbf{SCM}_{\rm FSA}}$.

(2) 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 $n \leqslant k$ holds $\mathbf{IC}_{(\mathrm{Computation}(s+\cdot(P+\cdot\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(n)} \in \mathrm{dom}\,P$. Then $k < \mathrm{pseudo} - \mathrm{LifeSpan}(s, P)$.

(3) Let s be a state of $\mathbf{SCM}_{\rm FSA}$ and I, J be macro instructions. Suppose I is pseudo-closed on s. Let k be a natural number. Suppose $k \leqslant \mathrm{pseudo} - \mathrm{LifeSpan}(s, I)$. Then $(\mathrm{Computation}(s+\cdot(I+\cdot\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(k)$ and $(\mathrm{Computation}(s+\cdot((I;J)+\cdot\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(k)$ are equal outside the instruction locations of $\mathbf{SCM}_{\rm FSA}$.

(4) Let s be a state of $\mathbf{SCM}_{\rm FSA}$ and I be a macro instruction. If I is closed on s and halting on s, then $\mathrm{Directed}(I)$ is pseudo-closed on s.

(5) Let s be a state of $\mathbf{SCM}_{\rm FSA}$ and I be a macro instruction. If I is closed on s and halting on s, then $\mathrm{pseudo} - \mathrm{LifeSpan}(s, \mathrm{Directed}(I)) = \mathrm{LifeSpan}(s+\cdot(I+\cdot\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))) + 1$.

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

(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 $\mathbf{SCM}_{\mathrm{FSA}}$ and for every natural number k such that $P \subseteq Q$ holds $\mathrm{ProgramPart}(\mathrm{Relocated}(P, k)) \subseteq \mathrm{ProgramPart}(\mathrm{Relocated}(Q, k))$.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

(33) Let s_1, s_2 be states of $\mathbf{SCM}_{\mathrm{FSA}}$. Suppose that
- (i) $s_1(\mathrm{intloc}(0)) = s_2(\mathrm{intloc}(0))$,
- (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 $(s +\cdot s_3 \restriction$ 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 $\mathrm{Initialize}(s) \restriction$ the instruction locations of $\mathbf{SCM}_{\mathrm{FSA}} = s \restriction$ 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 $(s_3 +\cdot s \restriction$ 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 $\mathrm{IExec}(\mathrm{Stop}_{\mathrm{SCM}_{\mathrm{FSA}}}, s) = \mathrm{Initialize}(s) +\cdot \text{Start-At}(\mathrm{insloc}(0))$.

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

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

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

(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 $I +\cdot \text{Start-At}(\mathrm{insloc}(0)) \subseteq s_1$. 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 $s_1 \restriction (\text{Int-Locations} \cup \text{FinSeq-Locations}) = s_2 \restriction (\text{Int-Locations} \cup \text{FinSeq-Locations})$. Let i be a natural number. Then $\mathbf{IC}_{(\mathrm{Computation}(s_1))(i)} + n = \mathbf{IC}_{(\mathrm{Computation}(s_2))(i)}$ and

IncAddr(CurInstr((Computation(s_1))(i)), n) = CurInstr((Computation(s_2))(i)) and (Computation(s_1))(i)↾(Int-Locations ∪ FinSeq-Locations) = (Computation(s_2))(i)↾(Int-Locations ∪ 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+· Start-At(insloc(0)) ⊆ s_1 and I+· Start-At(insloc(0)) ⊆ s_2 and s_1↾(Int-Locations ∪ FinSeq-Locations) = s_2↾(Int-Locations ∪ FinSeq-Locations). Let i be a natural number. Then $\mathbf{IC}_{(\mathrm{Computation}(s_1))(i)} = \mathbf{IC}_{(\mathrm{Computation}(s_2))(i)}$ and CurInstr((Computation(s_1))(i)) = CurInstr((Computation(s_2))(i)) and (Computation(s_1))(i)↾(Int-Locations ∪ FinSeq-Locations) = (Computation(s_2))(i)↾(Int-Locations ∪ 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+· Start-At(insloc(0)) ⊆ s_1 and I+· Start-At(insloc(0)) ⊆ s_2 and s_1↾(Int-Locations ∪ FinSeq-Locations) = s_2↾(Int-Locations ∪ FinSeq-Locations). Then LifeSpan(s_1) = 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(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 IExec(I, s_1)↾(Int-Locations ∪ FinSeq-Locations) = IExec(I, s_2)↾(Int-Locations ∪ FinSeq-Locations).

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

Let I be a macro instruction. Observe that Initialized(I) is initial.

One can prove the following propositions:

(47) Let s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$ and I be a macro instruction. Then Initialized(I) is pseudo-closed on s if and only if I is pseudo-closed on 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 pseudo − LifeSpan(s, Initialized(I)) = pseudo − LifeSpan(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 pseudo − LifeSpan(s, Initialized(I)) = pseudo − LifeSpan(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 $s+\cdot(I+\cdot \text{Start-At}(\text{insloc}(0)))$ and $\text{pseudo}-\text{LifeSpan}(s, I) = \text{pseudo}-\text{LifeSpan}(s+\cdot(I+\cdot \text{Start-At}(\text{insloc}(0))), I)$.

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

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

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

(52) Let s_1, s_2 be states of $\mathbf{SCM}_{\mathrm{FSA}}$ and I be a macro instruction. Suppose $s_1 \upharpoonright (\text{Int-Locations} \cup \text{FinSeq-Locations}) = s_2 \upharpoonright (\text{Int-Locations} \cup \text{FinSeq-Locations})$. 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 $s(\text{intloc}(0)) = 1$. Then I is pseudo-closed on s if and only if I is pseudo-closed on $\text{Initialize}(s)$.

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

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

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

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

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

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

(ii) $I;\text{Stop}_{\mathrm{SCM}_{\mathrm{FSA}}}$ 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 $n \leqslant \mathrm{pseudo}-\mathrm{LifeSpan}(s, \mathrm{Directed}(I))$ holds $(\mathrm{Computation}(s+\cdot(I+\cdot\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(n)\restriction D = (\mathrm{Computation}(s+\cdot((I;\mathrm{Stop}_{\mathrm{SCM}_{\mathrm{FSA}}})+\cdot\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(n)\restriction D$.

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

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

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

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

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

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

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

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

(67) 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 $s(a) = 0$ and $\mathrm{Directed}(I)$ is pseudo-closed on s. Then $\mathrm{IExec}(if = 0(a, I, J), s)\restriction(\mathrm{Int\text{-}Locations} \cup \mathrm{FinSeq\text{-}Locations}) = \mathrm{IExec}(I;\mathrm{Stop}_{\mathrm{SCM}_{\mathrm{FSA}}}, s)\restriction(\mathrm{Int\text{-}Locations} \cup \mathrm{FinSeq\text{-}Locations})$.

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

(69) 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(intloc(0)) = 1 and $s(a) > 0$ and Directed(I) is pseudo-closed on s. Then IExec($if > 0(a, I, J), s$)↾(Int-Locations ∪ FinSeq-Locations) = IExec(I;$\mathrm{Stop}_{\mathrm{SCM}_{\mathrm{FSA}}}$, s) ↾(Int-Locations ∪ FinSeq-Locations).

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

(71) 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(intloc(0)) = 1 and $s(a) \neq 0$ and Directed(J) is pseudo-closed on s. Then IExec($if = 0(a, I, J), s$)↾(Int-Locations ∪ FinSeq-Locations) = IExec(J;$\mathrm{Stop}_{\mathrm{SCM}_{\mathrm{FSA}}}$, s) ↾(Int-Locations ∪ FinSeq-Locations).

(72) Let s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, I, J be macro instructions, and a be a read-write integer location. Suppose $s(a) \leqslant 0$ and Directed(J) is pseudo-closed on s. Then $if > 0(a, I, J)$ is halting on s and $if > 0(a, I, J)$ is closed on s and LifeSpan(s+·($if > 0(a, I, J)$+· Start-At(insloc(0)))) = LifeSpan(s+·((J;$\mathrm{Stop}_{\mathrm{SCM}_{\mathrm{FSA}}}$)+· Start-At(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(intloc(0)) = 1 and $s(a) \leqslant 0$ and Directed(J) is pseudo-closed on s. Then IExec($if > 0(a, I, J), s$)↾(Int-Locations ∪ FinSeq-Locations) = IExec(J;$\mathrm{Stop}_{\mathrm{SCM}_{\mathrm{FSA}}}$, s) ↾(Int-Locations ∪ 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 $if = 0(a, I, J)$ is closed on s and $if = 0(a, I, J)$ 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 $if > 0(a, I, J)$ is closed on s and $if > 0(a, I, J)$ 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 $\mathbf{SCM}_{\rm FSA}$ 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 $\mathbf{halt}_{\mathbf{SCM}_{\rm FSA}}$ 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 $\mathbf{SCM}_{\rm FSA}$ 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 $\mathbf{SCM}_{\rm FSA}$, 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 $\mathbf{SCM}_{\rm FSA}$, 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 $\mathbf{SCM}_{\rm FSA}$ 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 $\mathrm{Stop}_{\mathrm{SCM}_{\rm FSA}}$ does not destroy a.

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

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

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

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

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

(89) Let s be a state of $\mathbf{SCM}_{\rm FSA}$, 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 $(\mathrm{IExec}(I, s))(a) =$ (Computation(Initialize(s)$+\cdot$($I+\cdot$ Start-At(insloc(0)))))(k)(a).

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

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

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

(94) Let s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, 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 $\mathbf{IC}_{(\mathrm{Computation}(s {+\cdot} (I {+\cdot}\, \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(n)} \in \mathrm{dom}\, I$. Let n be a natural number. If $n \leqslant m$, then
$(\mathrm{Computation}(s {+\cdot} (I {+\cdot}\, \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(n)(a) = s(a)$.

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

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

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

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

(99) For every instruction i of $\mathbf{SCM}_{\mathrm{FSA}}$ such that i does not destroy intloc(0) holds 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) $({\rm Computation}(s_1 + \cdot (I + \cdot \text{Start-At}({\rm insloc}(0)))))(k)$ and $({\rm Computation}(s_2 + \cdot (I + \cdot \text{Start-At}({\rm insloc}(0)))))(k)$ are equal outside the instruction locations of $\mathbf{SCM}_{\rm FSA}$, and

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

(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 ${\rm LifeSpan}(s_1 + \cdot (I + \cdot \text{Start-At}({\rm insloc}(0)))) = {\rm LifeSpan}(s_2 + \cdot (I + \cdot \text{Start-At}({\rm insloc}(0))))$ and ${\rm Result}(s_1 + \cdot (I + \cdot \text{Start-At}({\rm insloc}(0))))$ and ${\rm Result}(s_2 + \cdot (I + \cdot \text{Start-At}({\rm insloc}(0))))$ are equal outside the instruction locations of $\mathbf{SCM}_{\rm FSA}$.

(102) Let N be a non empty set with non empty elements, S be a steady-programmed von Neumann definite AMI over N, and s be a state of S. Suppose s is halting. Then there exists a natural number k such that s halts at $\mathbf{IC}_{({\rm Computation}(s))(k)}$.

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

(iii) $I + \cdot \text{Start-At}({\rm insloc}(0)) \subseteq s_2$, 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 $\mathbf{SCM}_{\rm FSA}$.

2. The loop Macroinstruction

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

Let I be a macro instruction. The functor ${\rm loop}\, I$ yields a halt-free macro instruction and is defined by:

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

Next we state two propositions:

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

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

Let I be a good macro instruction. One can verify that loop I is good.

The following propositions are true:

(106) For every macro instruction I holds dom loop I = dom I.

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

(108) For every macro instruction I and for every set x such that $x \in$ dom I holds if $I(x) \neq \mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}}$, then (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 $m \leqslant$ LifeSpan(s+·(I+· Start-At(insloc(0)))). Then (Computation(s+·(I+· Start-At(insloc(0)))))(m) and (Computation(s+·(loop I+· Start-At(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 $m <$ LifeSpan(s+·(I+· Start-At(insloc(0)))), then CurInstr((Computation(s+·(I+· Start-At(insloc(0)))))(m)) = CurInstr((Computation(s+·(loop I+· Start-At(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 $m \leqslant$ LifeSpan(s+·(I+· Start-At(insloc(0)))), then CurInstr((Computation(s+·(loop I+· Start-At(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 CurInstr((Computation(s+·(loop I+· Start-At (insloc(0)))))(LifeSpan(s+·(I+· Start-At(insloc(0)))))) = goto insloc(0).

(113) Let s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$ and I be a paraclosed macro instruction. Suppose I+· Start-At(insloc(0)) $\subseteq s$ and s is halting. Let m be a natural number. Suppose $m \leqslant$ LifeSpan(s). Then (Computation(s))(m) and (Computation(s+· 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 Initialized(I) $\subseteq s$. Let k be a natural number. If $k \leqslant$ LifeSpan(s), then CurInstr((Computation(s+· 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 Times(a, I) yields a macro instruction and is defined by:

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

The following propositions are true:

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

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

(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 $\mathrm{loop}\, if = 0(a, \mathrm{Goto}(\mathrm{insloc}(2)), I;\mathrm{SubFrom}(a, \mathrm{intloc}(0)))$ 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 $\mathrm{Initialized}(\mathrm{loop}\, if = 0(a, \mathrm{Goto}(\mathrm{insloc}(2)), I;\mathrm{SubFrom}(a, \mathrm{intloc}(0))))$ 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 $\mathrm{Times}(a, I)$ is closed on s and $\mathrm{Times}(a, I)$ 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 $\mathrm{Initialized}(\mathrm{Times}(a, I))$ 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 $if = 0(a, I, J)$ does not destroy c and $if > 0(a, I, J)$ 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

(i) $s_2 = s{+\cdot}(\mathrm{loop}\, if = 0(a, \mathrm{Goto}(\mathrm{insloc}(2)), I;\mathrm{SubFrom}(a, \mathrm{intloc}(0))) {+\cdot}\, \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))$,

(ii) $k = \mathrm{LifeSpan}(s{+\cdot}(if = 0(a, \mathrm{Goto}(\mathrm{insloc}(2)), I;\mathrm{SubFrom}(a, \mathrm{intloc}(0))) {+\cdot}\, \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))) + 1$,

(iii) $(\mathrm{Computation}(s_2))(k)(a) = s(a) - 1$,

(iv) $(\mathrm{Computation}(s_2))(k)(\mathrm{intloc}(0)) = 1$,

(v) for every read-write integer location b such that $b \neq a$ holds $(\mathrm{Computation}(s_2))(k)(b) = (\mathrm{IExec}(I, s))(b)$,

(vi) for every finite sequence location f holds $(\mathrm{Computation}(s_2))(k)(f) = (\mathrm{IExec}(I, s))(f)$,

(vii) $\mathbf{IC}_{(\mathrm{Computation}(s_2))(k)} = \mathrm{insloc}(0)$, and

(viii) for every natural number n such that $n \leqslant k$ holds $\mathbf{IC}_{(\mathrm{Computation}(s_2))(n)} \in \mathrm{dom}\,\mathrm{loop}\,if = 0(a, \mathrm{Goto}(\mathrm{insloc}(2)), I;\mathrm{SubFrom}(a, \mathrm{intloc}(0)))$.

(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 $s(a) \leqslant 0$, 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

One can prove the following 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 $s(a) \geqslant 0$, 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. Some multi instructions defined by sequence of instructions of $\mathbf{SCM}_{\mathrm{FSA}}$. *Formalized Mathematics*, 5(**4**):615–619, 1996.

[2] Noriko Asamoto. Conditional branch macro instructions of $\mathbf{SCM}_{\mathrm{FSA}}$. Part I. *Formalized Mathematics*, 6(**1**):65–72, 1997.

[3] Noriko Asamoto. Conditional branch macro instructions of $\mathbf{SCM}_{\mathrm{FSA}}$. Part II. *Formalized Mathematics*, 6(**1**):73–80, 1997.

[4] Noriko Asamoto. Constant assignment macro instructions of $\mathbf{SCM}_{\mathrm{FSA}}$. Part II. *Formalized Mathematics*, 6(**1**):59–63, 1997.

[5] Noriko Asamoto, Yatsuka Nakamura, Piotr Rudnicki, and Andrzej Trybulec. On the composition of macro instructions. Part II. *Formalized Mathematics*, 6(**1**):41–47, 1997.

[6] Noriko Asamoto, Yatsuka Nakamura, Piotr Rudnicki, and Andrzej Trybulec. On the composition of macro instructions. Part III. *Formalized Mathematics*, 6(**1**):53–57, 1997.

[7] Grzegorz Bancerek. Cardinal numbers. *Formalized Mathematics*, 1(**2**):377–382, 1990.

[8] Grzegorz Bancerek. The fundamental properties of natural numbers. *Formalized Mathematics*, 1(**1**):41–46, 1990.

[9] Grzegorz Bancerek and Krzysztof Hryniewiecki. Segments of natural numbers and finite sequences. *Formalized Mathematics*, 1(**1**):107–114, 1990.

[10] Grzegorz Bancerek and Piotr Rudnicki. Development of terminology for **scm**. *Formalized Mathematics*, 4(**1**):61–67, 1993.

[11] Grzegorz Bancerek and Andrzej Trybulec. Miscellaneous facts about functions. *Formalized Mathematics*, 5(**4**):485–492, 1996.
[12] Czesław Byliński. A classical first order language. *Formalized Mathematics*, 1(**4**):669–676, 1990.
[13] Czesław Byliński. Functions and their basic properties. *Formalized Mathematics*, 1(**1**):55–65, 1990.
[14] Czesław Byliński. The modification of a function by a function and the iteration of the composition of a function. *Formalized Mathematics*, 1(**3**):521–527, 1990.
[15] Czesław Byliński. Products and coproducts in categories. *Formalized Mathematics*, 2(**5**):701–709, 1991.
[16] Krzysztof Hryniewiecki. Basic properties of real numbers. *Formalized Mathematics*, 1(**1**):35–40, 1990.
[17] Yatsuka Nakamura and Andrzej Trybulec. A mathematical model of CPU. *Formalized Mathematics*, 3(**2**):151–160, 1992.
[18] Takaya Nishiyama and Yasuho Mizuhara. Binary arithmetics. *Formalized Mathematics*, 4(**1**):83–86, 1993.
[19] Piotr Rudnicki and Andrzej Trybulec. Memory handling for $\mathbf{SCM}_{\rm FSA}$. *Formalized Mathematics*, 6(**1**):29–36, 1997.
[20] Yasushi Tanaka. On the decomposition of the states of SCM. *Formalized Mathematics*, 5(**1**):1–8, 1996.
[21] Andrzej Trybulec. Enumerated sets. *Formalized Mathematics*, 1(**1**):25–34, 1990.
[22] Andrzej Trybulec. Tarski Grothendieck set theory. *Formalized Mathematics*, 1(**1**):9–11, 1990.
[23] Andrzej Trybulec and Yatsuka Nakamura. Some remarks on the simple concrete model of computer. *Formalized Mathematics*, 4(**1**):51–56, 1993.
[24] Andrzej Trybulec and Yatsuka Nakamura. Modifying addresses of instructions of $\mathbf{SCM}_{\rm FSA}$. *Formalized Mathematics*, 5(**4**):571–576, 1996.
[25] Andrzej Trybulec and Yatsuka Nakamura. Relocability for $\mathbf{SCM}_{\rm FSA}$. *Formalized Mathematics*, 5(**4**):583–586, 1996.
[26] Andrzej Trybulec, Yatsuka Nakamura, and Noriko Asamoto. On the compositions of macro instructions. Part I. *Formalized Mathematics*, 6(**1**):21–27, 1997.
[27] Andrzej Trybulec, Yatsuka Nakamura, and Piotr Rudnicki. The $\mathbf{SCM}_{\rm FSA}$ computer. *Formalized Mathematics*, 5(**4**):519–528, 1996.
[28] Wojciech A. Trybulec. Pigeon hole principle. *Formalized Mathematics*, 1(**3**):575–579, 1990.
[29] Zinaida Trybulec and Halina Święczkowska. Boolean properties of sets. *Formalized Mathematics*, 1(**1**):17–23, 1990.
[30] Edmund Woronowicz. Relations and their basic properties. *Formalized Mathematics*, 1(**1**):73–83, 1990.

Received October 29, 1997