JOURNAL OF FORMALIZED MATHEMATICS
Volume 8, Released 1996, Published 2003
Inst. of Computer Science, Univ. of Białystok

On the Composition of Macro Instructions. Part II[1]

Noriko Asamoto
Ochanomizu University
Tokyo

Yatsuka Nakamura
Shinshu University
Nagano

Piotr Rudnicki
University of Alberta
Edmonton

Andrzej Trybulec
Warsaw University
Białystok

Summary. We define the semantics of macro instructions (introduced in [19]) in terms of executions of $\mathbf{SCM}_{\rm FSA}$. In a similar way, we define the semantics of macro composition. Several attributes of macro instructions are introduced (paraclosed, parahalting, keeping 0) and their usage enables a systematic treatment of the composition of macro intructions. This article is continued in [1].

MML Identifier: SCMFSA6B.

WWW: http://mizar.org/JFM/Vol8/scmfsa6b.html

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

1. PRELIMINARIES

One can prove the following proposition

(3)[1] For all functions f, g and for every set A such that $A \cap \operatorname{dom} f \subseteq A \cap \operatorname{dom} g$ holds $(f {+\cdot} g{\restriction}A){\restriction}A = g{\restriction}A$.

2. PROPERTIES OF START-AT

For simplicity, we adopt the following convention: m, n denote natural numbers, x denotes a set, i denotes an instruction of $\mathbf{SCM}_{\rm FSA}$, I denotes a macro instruction, a denotes an integer location, f denotes a finite sequence location, l, l_1 denote instruction-locations of $\mathbf{SCM}_{\rm FSA}$, and s, s_1, s_2 denote states of $\mathbf{SCM}_{\rm FSA}$.

One can prove the following propositions:

(4) $\text{Start-At}(\text{insloc}(0)) \subseteq \text{Initialized}(I)$.

(5) If $I {+\cdot} \text{Start-At}(\text{insloc}(n)) \subseteq s$, then $I \subseteq s$.

(6) $(I {+\cdot} \text{Start-At}(\text{insloc}(n))){\restriction}$the instruction locations of $\mathbf{SCM}_{\rm FSA} = I$.

(7) If $x \in \operatorname{dom} I$, then $I(x) = (I {+\cdot} \text{Start-At}(\text{insloc}(n)))(x)$.

[1]This work was partially supported by NSERC Grant OGP9207 and NATO CRG 951368.
[1] The propositions (1) and (2) have been removed.

(8) If $\mathrm{Initialized}(I) \subseteq s$, then $I +\cdot \text{Start-At}(\mathrm{insloc}(0)) \subseteq s$.

(9) $a \notin \mathrm{dom}\,\text{Start-At}(l)$.

(10) $f \notin \mathrm{dom}\,\text{Start-At}(l)$.

(11) $l_1 \notin \mathrm{dom}\,\text{Start-At}(l)$.

(12) $a \notin \mathrm{dom}(I +\cdot \text{Start-At}(l))$.

(13) $f \notin \mathrm{dom}(I +\cdot \text{Start-At}(l))$.

(14) $s +\cdot I +\cdot \text{Start-At}(\mathrm{insloc}(0)) = s +\cdot \text{Start-At}(\mathrm{insloc}(0)) +\cdot I$.

3. Properties of AMI structures

In the sequel N is a non empty set with non empty elements.

Next we state two propositions:

(15) If $s = \mathrm{Following}(s)$, then for every n holds $(\mathrm{Computation}(s))(n) = s$.

(16) Let S be a halting IC-Ins-separated definite non empty non void AMI over N and s be a state of S. If s is halting, then $\mathrm{Result}(s) = (\mathrm{Computation}(s))(\mathrm{LifeSpan}(s))$.

Let us consider N, let S be an IC-Ins-separated definite non empty non void AMI over N, let s be a state of S, let l be an instruction-location of S, and let i be an instruction of S. Then $s +\cdot (l, i)$ is a state of S.

Let s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, let l_2 be an integer location, and let k be an integer. Then $s +\cdot (l_2, k)$ is a state of $\mathbf{SCM}_{\mathrm{FSA}}$.

We now state the proposition

(17) Let S be a steady-programmed IC-Ins-separated definite non empty non void AMI over N, s be a state of S, and given n. Then $s \restriction$the instruction locations of $S = (\mathrm{Computation}(s))(n) \restriction$the instruction locations of S.

4. Execution of macro instructions

Let I be a macro instruction and let s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$. The functor $\mathrm{IExec}(I, s)$ yields a state of $\mathbf{SCM}_{\mathrm{FSA}}$ and is defined by:

(Def. 1) $\mathrm{IExec}(I, s) = \mathrm{Result}(s +\cdot \mathrm{Initialized}(I)) +\cdot s \restriction$the instruction locations of $\mathbf{SCM}_{\mathrm{FSA}}$.

Let I be a macro instruction. We say that I is paraclosed if and only if:

(Def. 2) For every state s of $\mathbf{SCM}_{\mathrm{FSA}}$ and for every natural number n such that $I +\cdot \text{Start-At}(\mathrm{insloc}(0)) \subseteq s$ holds $\mathbf{IC}_{(\mathrm{Computation}(s))(n)} \in \mathrm{dom}\, I$.

We say that I is parahalting if and only if:

(Def. 3) $I +\cdot \text{Start-At}(\mathrm{insloc}(0))$ is halting.

We say that I is keeping 0 if and only if:

(Def. 4) For every state s of $\mathbf{SCM}_{\mathrm{FSA}}$ such that $I +\cdot \text{Start-At}(\mathrm{insloc}(0)) \subseteq s$ and for every natural number k holds $(\mathrm{Computation}(s))(k)(\mathrm{intloc}(0)) = s(\mathrm{intloc}(0))$.

Let us note that there exists a macro instruction which is parahalting.

We now state two propositions:

(18) For every parahalting macro instruction I such that $I +\cdot \text{Start-At}(\mathrm{insloc}(0)) \subseteq s$ holds s is halting.

(19) For every parahalting macro instruction I such that $\mathrm{Initialized}(I) \subseteq s$ holds s is halting.

Let I be a parahalting macro instruction. One can verify that $\mathrm{Initialized}(I)$ is halting.

One can prove the following two propositions:

(20) $s_2 + \cdot\, (\mathbf{IC}_{(s_2)}, \mathrm{goto}\ (\mathbf{IC}_{(s_2)}))$ is not halting.

(21) Suppose that

(i) s_1 and s_2 are equal outside the instruction locations of $\mathbf{SCM}_{\mathrm{FSA}}$,

(ii) $I \subseteq s_1$,

(iii) $I \subseteq s_2$, and

(iv) for every m such that $m < n$ holds $\mathbf{IC}_{(\mathrm{Computation}(s_2))(m)} \in \mathrm{dom}\, I$.

Let given m. Suppose $m \leq n$. Then $(\mathrm{Computation}(s_1))(m)$ and $(\mathrm{Computation}(s_2))(m)$ are equal outside the instruction locations of $\mathbf{SCM}_{\mathrm{FSA}}$.

Let us observe that every macro instruction which is parahalting is also paraclosed and every macro instruction which is keeping 0 is also paraclosed.

One can prove the following three propositions:

(22) Let I be a parahalting macro instruction and a be a read-write integer location. If $a \notin \mathrm{UsedIntLoc}(I)$, then $(\mathrm{IExec}(I, s))(a) = s(a)$.

(23) For every parahalting macro instruction I such that $f \notin \mathrm{UsedInt}^{*}\,\mathrm{Loc}(I)$ holds $(\mathrm{IExec}(I, s))(f) = s(f)$.

(24) If $\mathbf{IC}_s = l$ and $s(l) = \mathrm{goto}\ l$, then s is not halting.

Let us observe that every macro instruction which is parahalting is also non empty.

We now state a number of propositions:

(25) For every parahalting macro instruction I holds $\mathrm{dom}\, I \neq \emptyset$.

(26) For every parahalting macro instruction I holds $\mathrm{insloc}(0) \in \mathrm{dom}\, I$.

(27) Let J be a parahalting macro instruction. Suppose $J + \cdot\, \mathrm{Start\text{-}At}(\mathrm{insloc}(0)) \subseteq s_1$. Let n be a natural number. Suppose $\mathrm{ProgramPart}(\mathrm{Relocated}(J, n)) \subseteq s_2$ and $\mathbf{IC}_{(s_2)} = \mathrm{insloc}(n)$ and $s_1 \restriction (\mathrm{Int\text{-}Locations} \cup \mathrm{FinSeq\text{-}Locations}) = s_2 \restriction (\mathrm{Int\text{-}Locations} \cup \mathrm{FinSeq\text{-}Locations})$. Let i be a natural number. Then $\mathbf{IC}_{(\mathrm{Computation}(s_1))(i)} + n = \mathbf{IC}_{(\mathrm{Computation}(s_2))(i)}$ and $\mathrm{IncAddr}(\mathrm{CurInstr}((\mathrm{Computation}(s_1))(i)), n) = \mathrm{CurInstr}((\mathrm{Computation}(s_2))(i))$ and $(\mathrm{Computation}(s_1))(i) \restriction (\mathrm{Int\text{-}Locatic}$ $(\mathrm{Computation}(s_2))(i) \restriction (\mathrm{Int\text{-}Locations} \cup \mathrm{FinSeq\text{-}Locations})$.

(28) Let I be a parahalting macro instruction. Suppose $I + \cdot\, \mathrm{Start\text{-}At}(\mathrm{insloc}(0)) \subseteq s_1$ and $I + \cdot\, \mathrm{Start\text{-}At}(\mathrm{insloc}(0)) \subseteq s_2$ and s_1 and s_2 are equal outside the instruction locations of $\mathbf{SCM}_{\mathrm{FSA}}$. Let k be a natural number. Then $(\mathrm{Computation}(s_1))(k)$ and $(\mathrm{Computation}(s_2))(k)$ are equal outside the instruction locations of $\mathbf{SCM}_{\mathrm{FSA}}$ and $\mathrm{CurInstr}((\mathrm{Computation}(s_1))(k)) = \mathrm{CurInstr}((\mathrm{Computation}(s_2))(k))$.

(29) Let I be a parahalting macro instruction. Suppose $I + \cdot\, \mathrm{Start\text{-}At}(\mathrm{insloc}(0)) \subseteq s_1$ and $I + \cdot\, \mathrm{Start\text{-}At}(\mathrm{insloc}(0)) \subseteq s_2$ and s_1 and s_2 are equal outside the instruction locations of $\mathbf{SCM}_{\mathrm{FSA}}$. Then $\mathrm{LifeSpan}(s_1) = \mathrm{LifeSpan}(s_2)$ and $\mathrm{Result}(s_1)$ and $\mathrm{Result}(s_2)$ are equal outside the instruction locations of $\mathbf{SCM}_{\mathrm{FSA}}$.

(30) For every parahalting macro instruction I holds $\mathbf{IC}_{\mathrm{IExec}(I, s)} = \mathbf{IC}_{\mathrm{Result}(s + \cdot\, \mathrm{Initialized}(I))}$.

(31) For every non empty macro instruction I holds $\mathrm{insloc}(0) \in \mathrm{dom}\, I$ and $\mathrm{insloc}(0) \in \mathrm{dom}\, \mathrm{Initialized}(I)$ and $\mathrm{insloc}(0) \in \mathrm{dom}(I + \cdot\, \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))$.

(32) $x \in \mathrm{dom}\, \mathrm{Macro}(i)$ iff $x = \mathrm{insloc}(0)$ or $x = \mathrm{insloc}(1)$.

(33) $(\mathrm{Macro}(i))(\mathrm{insloc}(0)) = i$ and $(\mathrm{Macro}(i))(\mathrm{insloc}(1)) = \mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}}$ and $(\mathrm{Initialized}(\mathrm{Macro}(i)))(\mathrm{insloc}(0)) = i$ and $(\mathrm{Initialized}(\mathrm{Macro}(i)))(\mathrm{insloc}(1)) = \mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}}$ and $(\mathrm{Macro}(i) +\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))(\mathrm{insloc}(0)) = i$.

(34) If $\mathrm{Initialized}(I) \subseteq s$, then $\mathbf{IC}_s = \mathrm{insloc}(0)$.

One can verify that there exists a macro instruction which is keeping 0 and parahalting.
We now state the proposition

(35) For every keeping 0 parahalting macro instruction I holds $(\mathrm{IExec}(I,s))(\mathrm{intloc}(0)) = 1$.

5. The Composition of Macro Instructions

Next we state several propositions:

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

(37) For every paraclosed macro instruction I such that $s +\cdot I$ is halting and $\mathrm{Directed}(I) \subseteq s$ and $\mathrm{Start\text{-}At}(\mathrm{insloc}(0)) \subseteq s$ holds $\mathbf{IC}_{(\mathrm{Computation}(s))(\mathrm{LifeSpan}(s +\cdot I)+1)} = \mathrm{insloc}(\mathrm{card}\, I)$.

(38) Let I be a paraclosed macro instruction. If $s +\cdot I$ is halting and $\mathrm{Directed}(I) \subseteq s$ and $\mathrm{Start\text{-}At}(\mathrm{insloc}(0)) \subseteq s$, then $(\mathrm{Computation}(s))(\mathrm{LifeSpan}(s +\cdot I)) \restriction (\mathrm{Int\text{-}Locations} \cup \mathrm{FinSeq\text{-}Locations}) = (\mathrm{Computation}(s))(\mathrm{LifeSpan}(s +\cdot I) + 1) \restriction (\mathrm{Int\text{-}Locations} \cup \mathrm{FinSeq\text{-}Locations})$.

(39) Let 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 $\mathrm{CurInstr}((\mathrm{Computation}(s +\cdot \mathrm{Directed}(I)))(k)) \neq \mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}}$.

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

Let I, J be parahalting macro instructions. Observe that $I;\, J$ is parahalting.
The following two propositions are true:

(41) Let I be a keeping 0 macro instruction. Suppose $s +\cdot (I +\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))$ is not halting. Let J be a macro instruction and k be a natural number. 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 $\mathbf{SCM}_{\mathrm{FSA}}$.

(42) Let I be a keeping 0 macro instruction. Suppose $s +\cdot I$ is halting. Let J be a paraclosed macro instruction. Suppose $(I;\, J) +\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)) \subseteq s$. Let k be a natural number. Then $(\mathrm{Computation}(\mathrm{Result}(s +\cdot I) +\cdot (J +\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(k) +\cdot \mathrm{Start\text{-}At}(\mathbf{IC}_{(\mathrm{Computation}(\mathrm{Result}(s +\cdot I) +\cdot (J +\cdot \mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))}$ $\mathrm{card}\, I)$ and $(\mathrm{Computation}(s +\cdot (I;\, J)))(\mathrm{LifeSpan}(s +\cdot I) + 1 + k)$ are equal outside the instruction locations of $\mathbf{SCM}_{\mathrm{FSA}}$.

Let I, J be keeping 0 macro instructions. Note that $I;\, J$ is keeping 0.
The following two propositions are true:

(43) Let I be a keeping 0 parahalting macro instruction and J be a parahalting macro instruction. Then $\mathrm{LifeSpan}(s +\cdot \mathrm{Initialized}(I;\, J)) = \mathrm{LifeSpan}(s +\cdot \mathrm{Initialized}(I)) + 1 + \mathrm{LifeSpan}(\mathrm{Result}(s +\cdot \mathrm{Initialized}(I)) +\cdot \mathrm{Initialized}(J))$.

(44) Let I be a keeping 0 parahalting macro instruction and J be a parahalting macro instruction. Then $\mathrm{IExec}(I;\, J, s) = \mathrm{IExec}(J, \mathrm{IExec}(I,s)) +\cdot \mathrm{Start\text{-}At}(\mathbf{IC}_{\mathrm{IExec}(J,\mathrm{IExec}(I,s))} + \mathrm{card}\, I)$.

REFERENCES

[1] 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`.

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

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

[4] Grzegorz Bancerek and Krzysztof Hryniewiecki. Segments of natural numbers and finite sequences. *Journal of Formalized Mathematics*, 1, 1989. `http://mizar.org/JFM/Vol1/finseq_1.html`.

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

[6] Grzegorz Bancerek and Andrzej Trybulec. Miscellaneous facts about functions. *Journal of Formalized Mathematics*, 8, 1996. `http://mizar.org/JFM/Vol8/funct_7.html`.

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

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

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

[10] 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`.

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

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

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

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

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

[16] 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`.

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

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

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

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

[21] Michał J. Trybulec. Integers. *Journal of Formalized Mathematics*, 2, 1990. `http://mizar.org/JFM/Vol2/int_1.html`.

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

Received July 22, 1996

Published January 2, 2004