FORMALIZED MATHEMATICS
Volume 6, Number 1, 1997
Warsaw University - Białystok

Constant Assignment Macro Instructions of $\mathbf{SCM}_{\mathrm{FSA}}$. Part II

Noriko Asamoto
Ochanomizu University
Tokyo

MML Identifier: `SCMFSA7B`.

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

In this paper m is a natural number.

Next we state two propositions:

(1) For every finite sequence p of elements of the instructions of $\mathbf{SCM}_{\mathrm{FSA}}$ holds $\mathrm{dom}\,\mathrm{Load}(p) = \{\mathrm{insloc}(m) : m < \mathrm{len}\,p\}$.

(2) For every finite sequence p of elements of the instructions of $\mathbf{SCM}_{\mathrm{FSA}}$ holds $\mathrm{rng}\,\mathrm{Load}(p) = \mathrm{rng}\,p$.

Let p be a finite sequence of elements of the instructions of $\mathbf{SCM}_{\mathrm{FSA}}$. Observe that $\mathrm{Load}(p)$ is initial and programmed.

We now state several propositions:

(3) For every instruction i of $\mathbf{SCM}_{\mathrm{FSA}}$ holds $\mathrm{Load}(\langle i \rangle) = \mathrm{insloc}(0) \dot{\longmapsto} i$.

(4) For every instruction i of $\mathbf{SCM}_{\mathrm{FSA}}$ holds $\mathrm{dom}\,\mathrm{Macro}(i) = \{\mathrm{insloc}(0), \mathrm{insloc}(1)\}$.

(5) For every instruction i of $\mathbf{SCM}_{\mathrm{FSA}}$ holds $\mathrm{Macro}(i) = \mathrm{Load}(\langle i, \mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}} \rangle)$.

(6) For every instruction i of $\mathbf{SCM}_{\mathrm{FSA}}$ holds $\mathrm{card}\,\mathrm{Macro}(i) = 2$.

(7) For every instruction i of $\mathbf{SCM}_{\mathrm{FSA}}$ holds if $i = \mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}}$, then $(\mathrm{Directed}(\mathrm{Macro}(i)))(\mathrm{insloc}(0)) = \mathbf{goto}\ \mathrm{insloc}(2)$ and if $i \neq \mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}}$, then $(\mathrm{Directed}(\mathrm{Macro}(i)))(\mathrm{insloc}(0)) = i$.

(8) For every instruction i of $\mathbf{SCM}_{\mathrm{FSA}}$ holds $(\mathrm{Directed}(\mathrm{Macro}(i)))(\mathrm{insloc}(1)) = \mathbf{goto}\ \mathrm{insloc}(2)$.

Let a be an integer location and let k be an integer. Observe that $a{:=}k$ is initial and programmed.

ISSN 1426–2630

Let a be an integer location and let k be an integer. Observe that $a{:=}k$ is parahalting.

We now state the proposition

(9) Let s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, and let a be a read-write integer location, and let k be an integer. Then

(i) $(\mathrm{IExec}(a{:=}k, s))(a) = k$,

(ii) for every read-write integer location b such that $b \neq a$ holds $(\mathrm{IExec}(a{:=}k, s))(b) = s(b)$, and

(iii) for every finite sequence location f holds $(\mathrm{IExec}(a{:=}k, s))(f) = s(f)$.

Let f be a finite sequence location and let p be a finite sequence of elements of $\mathbb{Z}$. One can check that $f{:=}p$ is initial and programmed.

Let f be a finite sequence location and let p be a finite sequence of elements of $\mathbb{Z}$. Observe that $f{:=}p$ is parahalting.

The following proposition is true

(10) Let s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$, and let f be a finite sequence location, and let p be a finite sequence of elements of $\mathbb{Z}$. Then

(i) $(\mathrm{IExec}(f{:=}p, s))(f) = p$,

(ii) for every read-write integer location a such that $a \neq \mathrm{intloc}(1)$ and $a \neq \mathrm{intloc}(2)$ holds $(\mathrm{IExec}(f{:=}p, s))(a) = s(a)$, and

(iii) for every finite sequence location g such that $g \neq f$ holds $(\mathrm{IExec}(f{:=}p, s))(g) = s(g)$.

Let i be an instruction of $\mathbf{SCM}_{\mathrm{FSA}}$ and let a be an integer location. We say that i does not refer a if and only if the condition (Def. 1) is satisfied.

(Def. 1) Let b be an integer location, and let l be an instruction-location of $\mathbf{SCM}_{\mathrm{FSA}}$, and let f be a finite sequence location. Then

(i) $b{:=}a \neq i$,

(ii) $\mathrm{AddTo}(b, a) \neq i$,

(iii) $\mathrm{SubFrom}(b, a) \neq i$,

(iv) $\mathrm{MultBy}(b, a) \neq i$,

(v) $\mathrm{Divide}(b, a) \neq i$,

(vi) $\mathrm{Divide}(a, b) \neq i$,

(vii) $\mathbf{if}\ a = 0\ \mathbf{goto}\ l \neq i$,

(viii) $\mathbf{if}\ a > 0\ \mathbf{goto}\ l \neq i$,

(ix) $b{:=}f_a \neq i$,

(x) $f_b{:=}a \neq i$,

(xi) $f_a{:=}b \neq i$, and

(xii) $f{:=}\langle \underbrace{0, \ldots, 0}_{a} \rangle \neq i$.

Let I be a programmed finite partial state of $\mathbf{SCM}_{\mathrm{FSA}}$ and let a be an integer location. We say that I does not refer a if and only if:

(Def. 2) For every instruction i of $\mathbf{SCM}_{\mathrm{FSA}}$ such that $i \in \mathrm{rng}\, I$ holds i does not refer a.

Let i be an instruction of $\mathbf{SCM}_{\mathrm{FSA}}$ and let a be an integer location. We say that i does not destroy a if and only if the condition (Def. 3) is satisfied.

(Def. 3) Let b be an integer location and let f be a finite sequence location. Then $a{:=}b \neq i$ and $\mathrm{AddTo}(a,b) \neq i$ and $\mathrm{SubFrom}(a,b) \neq i$ and $\mathrm{MultBy}(a,b) \neq i$ and $\mathrm{Divide}(a,b) \neq i$ and $\mathrm{Divide}(b,a) \neq i$ and $a{:=}f_b \neq i$ and $a{:=}\mathrm{len} f \neq i$.

Let I be a finite partial state of $\mathbf{SCM}_{\mathrm{FSA}}$ and let a be an integer location. We say that I does not destroy a if and only if:

(Def. 4) For every instruction i of $\mathbf{SCM}_{\mathrm{FSA}}$ such that $i \in \mathrm{rng}\, I$ holds i does not destroy a.

Let I be a finite partial state of $\mathbf{SCM}_{\mathrm{FSA}}$. We say that I is good if and only if:

(Def. 5) I does not destroy intloc(0).

Let I be a finite partial state of $\mathbf{SCM}_{\mathrm{FSA}}$. We say that I is halt-free if and only if:

(Def. 6) $\mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}} \notin \mathrm{rng}\, I$.

Let us observe that there exists a macro instruction which is halt-free and good.

The following propositions are true:

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

(12) For all integer locations a, b, c such that $a \neq b$ holds $b{:=}c$ does not destroy a.

(13) For all integer locations a, b, c such that $a \neq b$ holds $\mathrm{AddTo}(b,c)$ does not destroy a.

(14) For all integer locations a, b, c such that $a \neq b$ holds $\mathrm{SubFrom}(b,c)$ does not destroy a.

(15) For all integer locations a, b, c such that $a \neq b$ holds $\mathrm{MultBy}(b,c)$ does not destroy a.

(16) For all integer locations a, b, c such that $a \neq b$ and $a \neq c$ holds $\mathrm{Divide}(b,c)$ does not destroy a.

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

(18) For all integer locations a, b and for every instruction-location l of $\mathbf{SCM}_{\mathrm{FSA}}$ holds **if** $b = 0$ **goto** l does not destroy a.

(19) For all integer locations a, b and for every instruction-location l of $\mathbf{SCM}_{\mathrm{FSA}}$ holds **if** $b > 0$ **goto** l does not destroy a.

(20) Let a, b, c be integer locations and let f be a finite sequence location. If $a \neq b$, then $b{:=}f_c$ does not destroy a.

(21) For all integer locations a, b, c and for every finite sequence location f holds $f_c{:=}b$ does not destroy a.

(22) Let a, b be integer locations and let f be a finite sequence location. If $a \neq b$, then $b{:=}\mathrm{len} f$ does not destroy a.

(23) For all integer locations a, b and for every finite sequence location f holds $f{:=}\langle \underbrace{0, \ldots, 0}_{b} \rangle$ does not destroy a.

Let I be a finite partial state of $\mathbf{SCM}_{\mathrm{FSA}}$ and let s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$. We say that I is closed on s if and only if:

(Def. 7) For every natural number k holds $\mathbf{IC}_{(\mathrm{Computation}(s+\cdot(I+\cdot\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))))(k)} \in \mathrm{dom}\, I$.

We say that I is halting on s if and only if:

(Def. 8) $s+\cdot(I+\cdot\,\mathrm{Start\text{-}At}(\mathrm{insloc}(0)))$ is halting.

We now state several propositions:

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

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

(26) Let i be an instruction of $\mathbf{SCM}_{\mathrm{FSA}}$, and let a be an integer location, and let s be a state of $\mathbf{SCM}_{\mathrm{FSA}}$. If i does not destroy a then $(\mathrm{Exec}(i, s))(a) = s(a)$.

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

(28) $\mathrm{Stop}_{\mathrm{SCM}_{\mathrm{FSA}}}$ does not destroy $\mathrm{intloc}(0)$.

One can verify that there exists a macro instruction which is parahalting and good.

One can check that $\mathrm{Stop}_{\mathrm{SCM}_{\mathrm{FSA}}}$ is parahalting and good.

One can check that every macro instruction which is paraclosed and good is also keeping 0.

One can prove the following two propositions:

(29) For every integer location a and for every integer k holds $\mathrm{rng}\,\mathrm{aSeq}(a, k) \subseteq \{a{:=}\,\mathrm{intloc}(0), \mathrm{AddTo}(a, \mathrm{intloc}(0)), \mathrm{SubFrom}(a, \mathrm{intloc}(0))\}$.

(30) For every integer location a and for every integer k holds $\mathrm{rng}(a{:=}k) \subseteq \{\mathbf{halt}_{\mathbf{SCM}_{\mathrm{FSA}}}, a{:=}\,\mathrm{intloc}(0), \mathrm{AddTo}(a, \mathrm{intloc}(0)), \mathrm{SubFrom}(a, \mathrm{intloc}(0))\}$.

Let a be a read-write integer location and let k be an integer. One can check that $a{:=}k$ is good.

Let a be a read-write integer location and let k be an integer. Observe that $a{:=}k$ is keeping 0.

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

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

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

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

[6] Grzegorz Bancerek and Piotr Rudnicki. On defining functions on trees. *Formalized Mathematics*, 4(**1**):91–101, 1993.
[7] Grzegorz Bancerek and Andrzej Trybulec. Miscellaneous facts about functions. *Formalized Mathematics*, 5(**4**):485–492, 1996.
[8] Czesław Byliński. A classical first order language. *Formalized Mathematics*, 1(**4**):669–676, 1990.
[9] Czesław Byliński. Finite sequences and tuples of elements of a non-empty sets. *Formalized Mathematics*, 1(**3**):529–536, 1990.
[10] Czesław Byliński. Functions and their basic properties. *Formalized Mathematics*, 1(**1**):55–65, 1990.
[11] 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.
[12] Krzysztof Hryniewiecki. Basic properties of real numbers. *Formalized Mathematics*, 1(**1**):35–40, 1990.
[13] Yatsuka Nakamura and Andrzej Trybulec. A mathematical model of CPU. *Formalized Mathematics*, 3(**2**):151–160, 1992.
[14] Yatsuka Nakamura and Andrzej Trybulec. On a mathematical model of programs. *Formalized Mathematics*, 3(**2**):241–250, 1992.
[15] Takaya Nishiyama and Yasuho Mizuhara. Binary arithmetics. *Formalized Mathematics*, 4(**1**):83–86, 1993.
[16] Piotr Rudnicki and Andrzej Trybulec. Memory handling for $\mathbf{SCM}_{\mathrm{FSA}}$. *Formalized Mathematics*, 6(**1**):29–36, 1997.
[17] Yasushi Tanaka. On the decomposition of the states of SCM. *Formalized Mathematics*, 5(**1**):1–8, 1996.
[18] Andrzej Trybulec. Binary operations applied to functions. *Formalized Mathematics*, 1(**2**):329–334, 1990.
[19] Andrzej Trybulec. Enumerated sets. *Formalized Mathematics*, 1(**1**):25–34, 1990.
[20] Andrzej Trybulec. Tarski Grothendieck set theory. *Formalized Mathematics*, 1(**1**):9–11, 1990.
[21] Andrzej Trybulec and Yatsuka Nakamura. Modifying addresses of instructions of $\mathbf{SCM}_{\mathrm{FSA}}$. *Formalized Mathematics*, 5(**4**):571–576, 1996.
[22] Andrzej Trybulec and Yatsuka Nakamura. Some remarks on the simple concrete model of computer. *Formalized Mathematics*, 4(**1**):51–56, 1993.
[23] Andrzej Trybulec, Yatsuka Nakamura, and Noriko Asamoto. On the compositions of macro instructions. Part I. *Formalized Mathematics*, 6(**1**):21–27, 1997.
[24] Andrzej Trybulec, Yatsuka Nakamura, and Piotr Rudnicki. The $\mathbf{SCM}_{\mathrm{FSA}}$ computer. *Formalized Mathematics*, 5(**4**):519–528, 1996.
[25] Michał J. Trybulec. Integers. *Formalized Mathematics*, 1(**3**):501–505, 1990.
[26] Wojciech A. Trybulec. Binary operations on finite sequences. *Formalized Mathematics*, 1(**5**):979–981, 1990.
[27] Wojciech A. Trybulec. Pigeon hole principle. *Formalized Mathematics*, 1(**3**):575–579, 1990.
[28] Zinaida Trybulec and Halina Święczkowska. Boolean properties of sets. *Formalized Mathematics*, 1(**1**):17–23, 1990.
[29] Edmund Woronowicz. Relations and their basic properties. *Formalized Mathematics*, 1(**1**):73–83, 1990.

Received August 27, 1996