JOURNAL OF FORMALIZED MATHEMATICS
Volume 13, Released 2001, Published 2003
Inst. of Computer Science, Univ. of Białystok

On the Instructions of SCM[1]

Artur Korniłowicz
University of Białystok

MML Identifier: AMI_6.

WWW: http://mizar.org/JFM/Vol13/ami_6.html

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

For simplicity, we follow the rules: a, b denote data-locations, i_1, i_2, i_3 denote instruction-locations of **SCM**, s_1, s_2 denote states of **SCM**, T denotes an instruction type of **SCM**, and k denotes a natural number.

The following propositions are true:

(1) $a \notin$ the instruction locations of **SCM**.

(2) $\text{Data-Loc}_{\mathbf{SCM}} \neq$ the instruction locations of **SCM**.

(3) For every object o of **SCM** holds $o = \mathbf{IC}_{\mathbf{SCM}}$ or $o \in$ the instruction locations of **SCM** or o is a data-location.

(4) If $i_2 \neq i_3$, then $\text{Next}(i_2) \neq \text{Next}(i_3)$.

(5) If s_1 and s_2 are equal outside the instruction locations of **SCM**, then $s_1(a) = s_2(a)$.

(6) Let N be a set with non empty elements, S be a realistic IC-Ins-separated definite non empty non void AMI over N, t, u be states of S, i_1 be an instruction-location of S, e be an element of $\text{ObjectKind}(\mathbf{IC}_S)$, and I be an element of $\text{ObjectKind}(i_1)$. If $e = i_1$ and $u = t {+\cdot} [\mathbf{IC}_S \longmapsto e, i_1 \longmapsto I]$, then $u(i_1) = I$ and $\mathbf{IC}_u = i_1$ and $\mathbf{IC}_{\text{Following}(u)} = (\text{Exec}(u(\mathbf{IC}_u), u))(\mathbf{IC}_S)$.

(7) $\text{AddressPart}(\mathbf{halt}_{\mathbf{SCM}}) = \emptyset$.

(8) $\text{AddressPart}(a{:=}b) = \langle a, b \rangle$.

(9) $\text{AddressPart}(\text{AddTo}(a,b)) = \langle a, b \rangle$.

(10) $\text{AddressPart}(\text{SubFrom}(a,b)) = \langle a, b \rangle$.

(11) $\text{AddressPart}(\text{MultBy}(a,b)) = \langle a, b \rangle$.

(12) $\text{AddressPart}(\text{Divide}(a,b)) = \langle a, b \rangle$.

(13) $\text{AddressPart}(\text{goto } i_2) = \langle i_2 \rangle$.

(14) $\text{AddressPart}(\mathbf{if}\ a = 0\ \mathbf{goto}\ i_2) = \langle i_2, a \rangle$.

(15) $\text{AddressPart}(\mathbf{if}\ a > 0\ \mathbf{goto}\ i_2) = \langle i_2, a \rangle$.

[1]This work has been partially supported by TYPES grant IST-1999-29001.

(16) If $T = 0$, then AddressParts $T = \{0\}$.

Let us consider T. Observe that AddressParts T is non empty.
We now state a number of propositions:

(17) If $T = 1$, then $\mathrm{dom} \prod_{\mathrm{AddressParts}\, T} = \{1,2\}$.

(18) If $T = 2$, then $\mathrm{dom} \prod_{\mathrm{AddressParts}\, T} = \{1,2\}$.

(19) If $T = 3$, then $\mathrm{dom} \prod_{\mathrm{AddressParts}\, T} = \{1,2\}$.

(20) If $T = 4$, then $\mathrm{dom} \prod_{\mathrm{AddressParts}\, T} = \{1,2\}$.

(21) If $T = 5$, then $\mathrm{dom} \prod_{\mathrm{AddressParts}\, T} = \{1,2\}$.

(22) If $T = 6$, then $\mathrm{dom} \prod_{\mathrm{AddressParts}\, T} = \{1\}$.

(23) If $T = 7$, then $\mathrm{dom} \prod_{\mathrm{AddressParts}\, T} = \{1,2\}$.

(24) If $T = 8$, then $\mathrm{dom} \prod_{\mathrm{AddressParts}\, T} = \{1,2\}$.

(25) $\prod_{\mathrm{AddressParts}\,\mathrm{InsCode}(a:=b)}(1) = \text{Data-Loc}_{\mathrm{SCM}}$.

(26) $\prod_{\mathrm{AddressParts}\,\mathrm{InsCode}(a:=b)}(2) = \text{Data-Loc}_{\mathrm{SCM}}$.

(27) $\prod_{\mathrm{AddressParts}\,\mathrm{InsCode}(\mathrm{AddTo}(a,b))}(1) = \text{Data-Loc}_{\mathrm{SCM}}$.

(28) $\prod_{\mathrm{AddressParts}\,\mathrm{InsCode}(\mathrm{AddTo}(a,b))}(2) = \text{Data-Loc}_{\mathrm{SCM}}$.

(29) $\prod_{\mathrm{AddressParts}\,\mathrm{InsCode}(\mathrm{SubFrom}(a,b))}(1) = \text{Data-Loc}_{\mathrm{SCM}}$.

(30) $\prod_{\mathrm{AddressParts}\,\mathrm{InsCode}(\mathrm{SubFrom}(a,b))}(2) = \text{Data-Loc}_{\mathrm{SCM}}$.

(31) $\prod_{\mathrm{AddressParts}\,\mathrm{InsCode}(\mathrm{MultBy}(a,b))}(1) = \text{Data-Loc}_{\mathrm{SCM}}$.

(32) $\prod_{\mathrm{AddressParts}\,\mathrm{InsCode}(\mathrm{MultBy}(a,b))}(2) = \text{Data-Loc}_{\mathrm{SCM}}$.

(33) $\prod_{\mathrm{AddressParts}\,\mathrm{InsCode}(\mathrm{Divide}(a,b))}(1) = \text{Data-Loc}_{\mathrm{SCM}}$.

(34) $\prod_{\mathrm{AddressParts}\,\mathrm{InsCode}(\mathrm{Divide}(a,b))}(2) = \text{Data-Loc}_{\mathrm{SCM}}$.

(35) $\prod_{\mathrm{AddressParts}\,\mathrm{InsCode}(\mathrm{goto}\ i_2)}(1) =$ the instruction locations of **SCM**.

(36) $\prod_{\mathrm{AddressParts}\,\mathrm{InsCode}(\mathbf{if}\ a=0\ \mathbf{goto}\ i_2)}(1) =$ the instruction locations of **SCM**.

(37) $\prod_{\mathrm{AddressParts}\,\mathrm{InsCode}(\mathbf{if}\ a=0\ \mathbf{goto}\ i_2)}(2) = \text{Data-Loc}_{\mathrm{SCM}}$.

(38) $\prod_{\mathrm{AddressParts}\,\mathrm{InsCode}(\mathbf{if}\ a>0\ \mathbf{goto}\ i_2)}(1) =$ the instruction locations of **SCM**.

(39) $\prod_{\mathrm{AddressParts}\,\mathrm{InsCode}(\mathbf{if}\ a>0\ \mathbf{goto}\ i_2)}(2) = \text{Data-Loc}_{\mathrm{SCM}}$.

(40) $\mathrm{NIC}(\mathbf{halt}_{\mathbf{SCM}}, i_1) = \{i_1\}$.

Let us note that JUMP($\mathbf{halt}_{\mathbf{SCM}}$) is empty.
We now state the proposition

(41) $\mathrm{NIC}(a:=b, i_1) = \{\mathrm{Next}(i_1)\}$.

Let us consider a, b. Observe that JUMP($a:=b$) is empty.
We now state the proposition

(42) $\mathrm{NIC}(\mathrm{AddTo}(a,b), i_1) = \{\mathrm{Next}(i_1)\}$.

Let us consider a, b. One can verify that JUMP(AddTo(a,b)) is empty.
We now state the proposition

(43) $\mathrm{NIC}(\mathrm{SubFrom}(a,b),i_1) = \{\mathrm{Next}(i_1)\}$.

Let us consider a, b. Observe that $\mathrm{JUMP}(\mathrm{SubFrom}(a,b))$ is empty.
We now state the proposition

(44) $\mathrm{NIC}(\mathrm{MultBy}(a,b),i_1) = \{\mathrm{Next}(i_1)\}$.

Let us consider a, b. Note that $\mathrm{JUMP}(\mathrm{MultBy}(a,b))$ is empty.
We now state the proposition

(45) $\mathrm{NIC}(\mathrm{Divide}(a,b),i_1) = \{\mathrm{Next}(i_1)\}$.

Let us consider a, b. One can verify that $\mathrm{JUMP}(\mathrm{Divide}(a,b))$ is empty.
One can prove the following propositions:

(46) $\mathrm{NIC}(\mathrm{goto}\ i_2,i_1) = \{i_2\}$.

(47) $\mathrm{JUMP}(\mathrm{goto}\ i_2) = \{i_2\}$.

Let us consider i_2. One can verify that $\mathrm{JUMP}(\mathrm{goto}\ i_2)$ is non empty and trivial.
We now state two propositions:

(48) $\mathrm{NIC}(\mathbf{if}\ a=0\ \mathbf{goto}\ i_2,i_1) = \{i_2,\mathrm{Next}(i_1)\}$.

(49) $\mathrm{JUMP}(\mathbf{if}\ a=0\ \mathbf{goto}\ i_2) = \{i_2\}$.

Let us consider a, i_2. Observe that $\mathrm{JUMP}(\mathbf{if}\ a=0\ \mathbf{goto}\ i_2)$ is non empty and trivial.
One can prove the following propositions:

(50) $\mathrm{NIC}(\mathbf{if}\ a>0\ \mathbf{goto}\ i_2,i_1) = \{i_2,\mathrm{Next}(i_1)\}$.

(51) $\mathrm{JUMP}(\mathbf{if}\ a>0\ \mathbf{goto}\ i_2) = \{i_2\}$.

Let us consider a, i_2. Observe that $\mathrm{JUMP}(\mathbf{if}\ a>0\ \mathbf{goto}\ i_2)$ is non empty and trivial.
We now state two propositions:

(52) $\mathrm{SUCC}(i_1) = \{i_1,\mathrm{Next}(i_1)\}$.

(53) Let f be a function from $\mathbb{N}$ into the instruction locations of **SCM**. Suppose that for every natural number k holds $f(k) = \mathbf{i}_k$. Then

(i) f is bijective, and

(ii) for every natural number k holds $f(k+1) \in \mathrm{SUCC}(f(k))$ and for every natural number j such that $f(j) \in \mathrm{SUCC}(f(k))$ holds $k \leq j$.

Let us observe that **SCM** is standard.
We now state three propositions:

(54) $\mathrm{il}_{\mathbf{SCM}}(k) = \mathbf{i}_k$.

(55) $\mathrm{Next}(\mathrm{il}_{\mathbf{SCM}}(k)) = \mathrm{il}_{\mathbf{SCM}}(k+1)$.

(56) $\mathrm{Next}(i_1) = \mathrm{NextLoc}\, i_1$.

Let us mention that $\mathrm{InsCode}(\mathbf{halt}_{\mathbf{SCM}})$ is jump-only.
Let us note that $\mathbf{halt}_{\mathbf{SCM}}$ is jump-only.
Let us consider i_2. Note that $\mathrm{InsCode}(\mathrm{goto}\ i_2)$ is jump-only.
Let us consider i_2. Observe that goto i_2 is jump-only, non sequential, and non instruction location free.

Let us consider a, i_2. Note that $\mathrm{InsCode}(\mathbf{if}\ a=0\ \mathbf{goto}\ i_2)$ is jump-only and $\mathrm{InsCode}(\mathbf{if}\ a>0\ \mathbf{goto}\ i_2)$ is jump-only.

Let us consider a, i_2. One can verify that **if** $a=0$ **goto** i_2 is jump-only, non sequential, and non instruction location free and **if** $a>0$ **goto** i_2 is jump-only, non sequential, and non instruction location free.

Let us consider a, b. One can verify the following observations:

* InsCode(a:=b) is non jump-only,
* InsCode(AddTo(a,b)) is non jump-only,
* InsCode(SubFrom(a,b)) is non jump-only,
* InsCode(MultBy(a,b)) is non jump-only, and
* InsCode(Divide(a,b)) is non jump-only.

Let us consider a, b. One can check the following observations:

* a:=b is non jump-only and sequential,
* AddTo(a,b) is non jump-only and sequential,
* SubFrom(a,b) is non jump-only and sequential,
* MultBy(a,b) is non jump-only and sequential, and
* Divide(a,b) is non jump-only and sequential.

Let us observe that **SCM** is homogeneous and has explicit jumps and no implicit jumps.
Let us note that **SCM** is regular.
We now state three propositions:

(57) IncAddr(goto $i_2,k) =$ goto $\mathrm{il}_{\mathbf{SCM}}(\mathrm{locnum}(i_2)+k)$.

(58) IncAddr(**if** $a=0$ **goto** $i_2,k) =$ **if** $a=0$ **goto** $\mathrm{il}_{\mathbf{SCM}}(\mathrm{locnum}(i_2)+k)$.

(59) IncAddr(**if** $a>0$ **goto** $i_2,k) =$ **if** $a>0$ **goto** $\mathrm{il}_{\mathbf{SCM}}(\mathrm{locnum}(i_2)+k)$.

Let us note that **SCM** is IC-good and Exec-preserving.

References

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

[2] Grzegorz Bancerek. The ordinal numbers. *Journal of Formalized Mathematics*, 1, 1989. http://mizar.org/JFM/Vol1/ordinal1.html.

[3] Grzegorz Bancerek. Sequences of ordinal numbers. *Journal of Formalized Mathematics*, 1, 1989. http://mizar.org/JFM/Vol1/ordinal2.html.

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

[5] 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.

[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] Artur Korniłowicz. On the composition of macro instructions of standard computers. *Journal of Formalized Mathematics*, 12, 2000. http://mizar.org/JFM/Vol12/amistd_2.html.

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

[13] Yatsuka Nakamura and Andrzej Trybulec. On a mathematical model of programs. *Journal of Formalized Mathematics*, 4, 1992. http://mizar.org/JFM/Vol4/ami_2.html.

[14] Beata Padlewska. Families of sets. *Journal of Formalized Mathematics*, 1, 1989. `http://mizar.org/JFM/Vol1/setfam_1.html`.

[15] Dariusz Surowik. Cyclic groups and some of their properties — part I. *Journal of Formalized Mathematics*, 3, 1991. `http://mizar.org/JFM/Vol3/gr_cy_1.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. Tarski Grothendieck set theory. *Journal of Formalized Mathematics*, Axiomatics, 1989. `http://mizar.org/JFM/Axiomatics/tarski.html`.

[18] Andrzej Trybulec. Tuples, projections and Cartesian products. *Journal of Formalized Mathematics*, 1, 1989. `http://mizar.org/JFM/Vol1/mcart_1.html`.

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

[20] 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`.

[21] Andrzej Trybulec, Piotr Rudnicki, and Artur Korniłowicz. Standard ordering of instruction locations. *Journal of Formalized Mathematics*, 12, 2000. `http://mizar.org/JFM/Vol12/amistd_1.html`.

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

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

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

Received May 8, 2001

Published January 2, 2004