JOURNAL OF FORMALIZED MATHEMATICS
Volume 11, Released 1999, Published 2003
Inst. of Computer Science, Univ. of Białystok

Computation of Two Consecutive Program Blocks for SCMPDS[1]

Jing-Chao Chen
Shanghai Jiaotong University

Summary. In this article, a program block without halting instructions is called No-StopCode program block. If a program consists of two blocks, where the first block is parahalting (i.e. halt for all states) and No-StopCode, and the second block is parahalting and shiftable, it can be computed by combining the computation results of the two blocks. For a program which consists of a instruction and a block, we obtain a similar conclusion. For a large amount of programs, the computation method given in the article is useful, but it is not suitable to recursive programs.

MML Identifier: SCMPDS_5.

WWW: http://mizar.org/JFM/Vol11/scmpds_5.html

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

1. PRELIMINARIES

For simplicity, we adopt the following convention: x denotes a set, m, n denote natural numbers, a, b denote Int positions, i denotes an instruction of SCMPDS, s, s_1, s_2 denote states of SCMPDS, k_1, k_2 denote integers, l_1 denotes an instruction-location of SCMPDS, I, J denote Program-blocks, and N denotes a set with non empty elements.

The following propositions are true:

(1) Let S be a halting IC-Ins-separated definite non empty non void AMI over N and s be a state of S. If $s = \mathrm{Following}(s)$, then for every n holds $(\mathrm{Computation}(s))(n) = s$.

(2) $x \in \mathrm{dom}\,\mathrm{Load}(i)$ iff $x = \mathrm{inspos}\,0$.

(3) If $l_1 \in \mathrm{dom}\,\mathrm{stop}\,I$ and $(\mathrm{stop}\,I)(l_1) \neq \mathbf{halt}_{\mathrm{SCMPDS}}$, then $l_1 \in \mathrm{dom}\,I$.

(4) $\mathrm{dom}\,\mathrm{Load}(i) = \{\mathrm{inspos}\,0\}$ and $(\mathrm{Load}(i))(\mathrm{inspos}\,0) = i$.

(5) $\mathrm{inspos}\,0 \in \mathrm{dom}\,\mathrm{Load}(i)$.

(6) $\mathrm{card}\,\mathrm{Load}(i) = 1$.

(7) $\mathrm{card}\,\mathrm{stop}\,I = \mathrm{card}\,I + 1$.

(8) $\mathrm{card}\,\mathrm{stop}\,\mathrm{Load}(i) = 2$.

[1]This research is partially supported by the National Natural Science Foundation of China Grant No. 69873033.

(9) inspos 0 $\in$ dom stop Load(i) and inspos 1 $\in$ dom stop Load(i).

(10) (stop Load(i))(inspos 0) $= i$ and (stop Load(i))(inspos 1) $=$ $\mathbf{halt}_{\mathrm{SCMPDS}}$.

(11) $x \in$ dom stop Load(i) iff $x =$ inspos 0 or $x =$ inspos 1.

(12) dom stop Load(i) $= \{$inspos 0, inspos 1$\}$.

(13) inspos 0 $\in$ dom Initialized(stop Load(i)) and inspos 1 $\in$ dom Initialized(stop Load(i)) and (Initialized(stop Load(i)))(inspos 0) $= i$ and (Initialized(stop Load(i)))(inspos 1) $=$ $\mathbf{halt}_{\mathrm{SCMPDS}}$.

(14) For all Program-blocks I, J holds Initialized(stop I; J) $= (I$; (J; SCMPDS $-$ Stop))$+\cdot$ Start-At(inspos 0).

(15) For all Program-blocks I, J holds Initialized(I) $\subseteq$ Initialized(stop I; J).

(16) dom stop $I \subseteq$ dom stop I; J.

(17) For all Program-blocks I, J holds Initialized(stop I)$+\cdot$ Initialized(stop I; J) $=$ Initialized(stop I; J).

(18) If Initialized(I) $\subseteq s$, then $\mathbf{IC}_s =$ inspos 0.

(19) ($s+\cdot$ Initialized(I))(a) $= s(a)$.

(20) Let I be a parahalting Program-block. Suppose Initialized(stop I) $\subseteq s_1$ and Initialized(stop I) $\subseteq s_2$ and s_1 and s_2 are equal outside the instruction locations of SCMPDS. Let k be a natural number. Then (Computation(s_1))(k) and (Computation(s_2))(k) are equal outside the instruction locations of SCMPDS and CurInstr((Computation(s_1))(k)) $=$ CurInstr((Computation(s_2))(k)).

(21) Let I be a parahalting Program-block. Suppose Initialized(stop I) $\subseteq s_1$ and Initialized(stop I) $\subseteq s_2$ and s_1 and s_2 are equal outside the instruction locations of SCMPDS. Then LifeSpan(s_1) $=$ LifeSpan(s_2) and Result(s_1) and Result(s_2) are equal outside the instruction locations of SCMPDS.

(22) For every Program-block I holds $\mathbf{IC}_{\mathrm{IExec}(I,s)} = \mathbf{IC}_{\mathrm{Result}(s+\cdot \mathrm{Initialized}(\mathrm{stop}\, I))}$.

(23) Let I be a parahalting Program-block and J be a Program-block. Suppose Initialized(stop I) $\subseteq s$. Let given m. Suppose $m \leq$ LifeSpan(s). Then (Computation(s))(m) and (Computation($s+\cdot(I$; J)))(m) are equal outside the instruction locations of SCMPDS.

(24) Let I be a parahalting Program-block and J be a Program-block. Suppose Initialized(stop I) $\subseteq s$. Let given m. Suppose $m \leq$ LifeSpan(s). Then (Computation(s))(m) and (Computation($s+\cdot$ Initialized(stop I; J)))(m) are equal outside the instruction locations of SCMPDS.

2. Non Halting Instructions and Parahalting Instructions

Let i be an instruction of SCMPDS. We say that i is No-StopCode if and only if:

(Def. 1) $i \neq$ $\mathbf{halt}_{\mathrm{SCMPDS}}$.

Let i be an instruction of SCMPDS. We say that i is parahalting if and only if:

(Def. 2) Load(i) is parahalting.

One can check that there exists an instruction of SCMPDS which is No-StopCode, shiftable, and parahalting.

One can prove the following proposition

(25) If $k_1 \neq 0$, then goto k_1 is No-StopCode.

Let us consider a. One can verify that return a is No-StopCode.
Let us consider a, k_1. Note that a:=k_1 is No-StopCode and saveIC(a,k_1) is No-StopCode.
Let us consider a, k_1, k_2. One can check the following observations:

* $(a,k_1) <> 0$_goto k_2 is No-StopCode,
* $(a,k_1) <= 0$_goto k_2 is No-StopCode,
* $(a,k_1) >= 0$_goto k_2 is No-StopCode, and
* a_{k_1}:=k_2 is No-StopCode.

Let us consider a, k_1, k_2. Observe that AddTo(a,k_1,k_2) is No-StopCode.
Let us consider a, b, k_1, k_2. One can verify the following observations:

* AddTo(a,k_1,b,k_2) is No-StopCode,
* SubFrom(a,k_1,b,k_2) is No-StopCode,
* MultBy(a,k_1,b,k_2) is No-StopCode,
* Divide(a,k_1,b,k_2) is No-StopCode, and
* $(a,k_1) := (b,k_2)$ is No-StopCode.

Let us note that $\mathbf{halt}_{\rm SCMPDS}$ is parahalting.
Let i be a parahalting instruction of SCMPDS. Note that Load(i) is parahalting.
Let us consider a, k_1. Observe that a:=k_1 is parahalting.
Let us consider a, k_1, k_2. Note that a_{k_1}:=k_2 is parahalting and AddTo(a,k_1,k_2) is parahalting.
Let us consider a, b, k_1, k_2. One can verify the following observations:

* AddTo(a,k_1,b,k_2) is parahalting,
* SubFrom(a,k_1,b,k_2) is parahalting,
* MultBy(a,k_1,b,k_2) is parahalting,
* Divide(a,k_1,b,k_2) is parahalting, and
* $(a,k_1) := (b,k_2)$ is parahalting.

One can prove the following proposition

(26) If InsCode$(i) = 1$, then i is not parahalting.

Let I_1 be a finite partial state of SCMPDS. We say that I_1 is No-StopCode if and only if:

(Def. 3) For every instruction-location x of SCMPDS such that $x \in \mathrm{dom}\, I_1$ holds $I_1(x) \neq \mathbf{halt}_{\rm SCMPDS}$.

One can check that there exists a Program-block which is parahalting, shiftable, and No-StopCode.
Let I, J be No-StopCode Program-blocks. Note that I; J is No-StopCode.
Let i be a No-StopCode instruction of SCMPDS. Observe that Load(i) is No-StopCode.
Let i be a No-StopCode instruction of SCMPDS and let J be a No-StopCode Program-block. One can verify that i; J is No-StopCode.
Let I be a No-StopCode Program-block and let j be a No-StopCode instruction of SCMPDS. Note that I; j is No-StopCode.
Let i, j be No-StopCode instructions of SCMPDS. One can check that i; j is No-StopCode.
The following propositions are true:

(27) For every parahalting No-StopCode Program-block I such that Initialized(stop I) $\subseteq s$ holds $\mathbf{IC}_{(\mathrm{Computation}(s))(\mathrm{LifeSpan}(s+\cdot\,\mathrm{Initialized}(\mathrm{stop}\,I)))} = \mathrm{inspos}\,\mathrm{card}\,I$.

(28) For every parahalting Program-block I and for every natural number k such that $k <$ LifeSpan$(s+\cdot$ Initialized(stop I)) holds $\mathbf{IC}_{(\mathrm{Computation}(s+\cdot\,\mathrm{Initialized}(\mathrm{stop}\,I)))(k)} \in \mathrm{dom}\, I$.

(29) Let I be a parahalting Program-block and k be a natural number. Suppose $\mathrm{Initialized}(I) \subseteq s$ and $k \leq \mathrm{LifeSpan}(s+\cdot\, \mathrm{Initialized}(\mathrm{stop}\, I))$. Then $(\mathrm{Computation}(s))(k)$ and $(\mathrm{Computation}(s+\cdot\, \mathrm{Initialized}(\mathrm{stop}\, I)))(k)$ are equal outside the instruction locations of SCMPDS.

(30) For every parahalting No-StopCode Program-block I such that $\mathrm{Initialized}(I) \subseteq s$ holds $\mathbf{IC}_{(\mathrm{Computation}(s))(\mathrm{LifeSpan}(s+\cdot\, \mathrm{Initialized}(\mathrm{stop}\, I)))} = \mathrm{inspos}\, \mathrm{card}\, I$.

(31) For every parahalting Program-block I such that $\mathrm{Initialized}(I) \subseteq s$ holds $\mathrm{CurInstr}((\mathrm{Computation}(s))(\mathrm{LifeSpan}(s+\cdot\, \mathrm{Ini}$ $\mathbf{halt}_{\mathrm{SCMPDS}}$ or $\mathbf{IC}_{(\mathrm{Computation}(s))(\mathrm{LifeSpan}(s+\cdot\, \mathrm{Initialized}(\mathrm{stop}\, I)))} = \mathrm{inspos}\, \mathrm{card}\, I$.

(32) Let I be a parahalting No-StopCode Program-block and k be a natural number. If $\mathrm{Initialized}(I) \subseteq s$ and $k < \mathrm{LifeSpan}(s+\cdot\, \mathrm{Initialized}(\mathrm{stop}\, I))$, then $\mathrm{CurInstr}((\mathrm{Computation}(s))(k)) \neq \mathbf{halt}_{\mathrm{SCMPDS}}$.

(33) Let I be a parahalting Program-block, J be a Program-block, and k be a natural number. Suppose $k \leq \mathrm{LifeSpan}(s+\cdot\, \mathrm{Initialized}(\mathrm{stop}\, I))$. Then $(\mathrm{Computation}(s+\cdot\, \mathrm{Initialized}(\mathrm{stop}\, I)))(k)$ and $(\mathrm{Computation}(s+\cdot((I;\, J)+\cdot\, \mathrm{Start\text{-}At}(\mathrm{inspos}\, 0))))(k)$ are equal outside the instruction locations of SCMPDS.

(34) Let I be a parahalting Program-block, J be a Program-block, and k be a natural number. Suppose $k \leq \mathrm{LifeSpan}(s+\cdot\, \mathrm{Initialized}(\mathrm{stop}\, I))$. Then $(\mathrm{Computation}(s+\cdot\, \mathrm{Initialized}(\mathrm{stop}\, I)))(k)$ and $(\mathrm{Computation}(s+\cdot\, \mathrm{Initialized}(\mathrm{stop}\, I;\, J)))(k)$ are equal outside the instruction locations of SCMPDS.

Let I be a parahalting Program-block and let J be a parahalting shiftable Program-block. One can verify that $I;\, J$ is parahalting.

Let i be a parahalting instruction of SCMPDS and let J be a parahalting shiftable Program-block. Note that $i;\, J$ is parahalting.

Let I be a parahalting Program-block and let j be a parahalting shiftable instruction of SCMPDS. Note that $I;\, j$ is parahalting.

Let i be a parahalting instruction of SCMPDS and let j be a parahalting shiftable instruction of SCMPDS. Observe that $i;\, j$ is parahalting.

One can prove the following proposition

(35) Let s, s_1 be states of SCMPDS and J be a parahalting shiftable Program-block. If $s = (\mathrm{Computation}(s_1+\cdot\, \mathrm{Initialized}(\mathrm{stop}\, J)))(m)$, then $\mathrm{Exec}(\mathrm{CurInstr}(s), s+\cdot\, \mathrm{Start\text{-}At}(\mathbf{IC}_s + n)) = \mathrm{Following}(s)+\cdot\, \mathrm{Start\text{-}At}(\mathbf{IC}_{\mathrm{Following}(s)} + n)$.

3. Computation of two Consecutive Program Blocks

One can prove the following propositions:

(36) Let I be a parahalting No-StopCode Program-block, J be a parahalting shiftable Program-block, and k be a natural number. Suppose $\mathrm{Initialized}(\mathrm{stop}\, I;\, J) \subseteq s$. Then $(\mathrm{Computation}(\mathrm{Result}(s+\cdot\, \mathrm{Initialized}(\mathrm{stop}\, I))+\cdot\, \mathrm{Initialized}(\mathrm{stop}\, J)))(k)+\cdot\, \mathrm{Start\text{-}At}(\mathbf{IC}_{(\mathrm{Computation}(\mathrm{Result}(s+\cdot\, \mathrm{Initialized}(\mathrm{stop}\, I))}$ $\mathrm{card}\, I)$ and $(\mathrm{Computation}(s+\cdot\, \mathrm{Initialized}(\mathrm{stop}\, I;\, J)))(\mathrm{LifeSpan}(s+\cdot\, \mathrm{Initialized}(\mathrm{stop}\, I)) + k)$ are equal outside the instruction locations of SCMPDS.

(37) Let I be a parahalting No-StopCode Program-block and J be a parahalting shiftable Program-block. Then $\mathrm{LifeSpan}(s+\cdot\, \mathrm{Initialized}(\mathrm{stop}\, I;\, J)) = \mathrm{LifeSpan}(s+\cdot\, \mathrm{Initialized}(\mathrm{stop}\, I)) + \mathrm{LifeSpan}(\mathrm{Result}(s+\cdot\, \mathrm{Initialized}(\mathrm{stop}\, I))+\cdot\, \mathrm{Initialized}(\mathrm{stop}\, J))$.

(38) Let I be a parahalting No-StopCode Program-block and J be a parahalting shiftable Program-block. Then $\mathrm{IExec}(I;\, J, s) = \mathrm{IExec}(J, \mathrm{IExec}(I, s))+\cdot\, \mathrm{Start\text{-}At}(\mathbf{IC}_{\mathrm{IExec}(J, \mathrm{IExec}(I, s))} + \mathrm{card}\, I)$.

(39) Let I be a parahalting No-StopCode Program-block and J be a parahalting shiftable Program-block. Then $(\mathrm{IExec}(I;\, J, s))(a) = (\mathrm{IExec}(J, \mathrm{IExec}(I, s)))(a)$.

4. Computation of the Program Consisting of a Instruction and a Block

Let s be a state of SCMPDS. The functor Initialized(s) yielding a state of SCMPDS is defined by:

(Def. 4) $\mathrm{Initialized}(s) = s {+\cdot}\, \mathrm{Start\text{-}At}(\mathrm{inspos}\,0)$.

Next we state several propositions:

(40) $\mathbf{IC}_{\mathrm{Initialized}(s)} = \mathrm{inspos}\,0$ and $(\mathrm{Initialized}(s))(a) = s(a)$ and $(\mathrm{Initialized}(s))(l_1) = s(l_1)$.

(41) s_1 and s_2 are equal outside the instruction locations of SCMPDS iff $s_1 \restriction (\mathrm{Data\text{-}Loc}_{\mathrm{SCM}} \cup \{\mathbf{IC}_{\mathrm{SCMPDS}}\}) = s_2 \restriction (\mathrm{Data\text{-}Loc}_{\mathrm{SCM}} \cup \{\mathbf{IC}_{\mathrm{SCMPDS}}\})$.

(43)[1] If $s_1 \restriction \mathrm{Data\text{-}Loc}_{\mathrm{SCM}} = s_2 \restriction \mathrm{Data\text{-}Loc}_{\mathrm{SCM}}$ and $\mathrm{InsCode}(i) \neq 3$, then $\mathrm{Exec}(i, s_1) \restriction \mathrm{Data\text{-}Loc}_{\mathrm{SCM}} = \mathrm{Exec}(i, s_2) \restriction \mathrm{Data\text{-}Loc}_{\mathrm{SCM}}$.

(44) For every shiftable instruction i of SCMPDS such that $s_1 \restriction \mathrm{Data\text{-}Loc}_{\mathrm{SCM}} = s_2 \restriction \mathrm{Data\text{-}Loc}_{\mathrm{SCM}}$ holds $\mathrm{Exec}(i, s_1) \restriction \mathrm{Data\text{-}Loc}_{\mathrm{SCM}} = \mathrm{Exec}(i, s_2) \restriction \mathrm{Data\text{-}Loc}_{\mathrm{SCM}}$.

(45) For every parahalting instruction i of SCMPDS holds $\mathrm{Exec}(i, \mathrm{Initialized}(s)) = \mathrm{IExec}(\mathrm{Load}(i), s)$.

(46) Let I be a parahalting No-StopCode Program-block and j be a parahalting shiftable instruction of SCMPDS. Then $(\mathrm{IExec}(I;\, j, s))(a) = (\mathrm{Exec}(j, \mathrm{IExec}(I, s)))(a)$.

(47) Let i be a No-StopCode parahalting instruction of SCMPDS and j be a shiftable parahalting instruction of SCMPDS. Then $(\mathrm{IExec}(i;\, j, s))(a) = (\mathrm{Exec}(j, \mathrm{Exec}(i, \mathrm{Initialized}(s))))(a)$.

References

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

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

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

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

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

[6] 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.

[7] Jing-Chao Chen. Computation and program shift in the SCMPDS computer. *Journal of Formalized Mathematics*, 11, 1999. http://mizar.org/JFM/Vol11/scmpds_3.html.

[8] Jing-Chao Chen. The construction and shiftability of program blocks for SCMPDS. *Journal of Formalized Mathematics*, 11, 1999. http://mizar.org/JFM/Vol11/scmpds_4.html.

[9] Jing-Chao Chen. The SCMPDS computer and the basic semantics of its instructions. *Journal of Formalized Mathematics*, 11, 1999. http://mizar.org/JFM/Vol11/scmpds_2.html.

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

[11] 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.

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

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

[14] 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.

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

[1] The proposition (42) has been removed.

[16] Wojciech A. Trybulec. Groups. *Journal of Formalized Mathematics*, 2, 1990. http://mizar.org/JFM/Vol2/group_1.html.

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

Received June 15, 1999

Published January 2, 2004