記号処理 12－16
（1980. 6. 21）

ハッシング・ハードウェアの試作とその応用

井田哲雄（理化学研究所）

§1. はじめに.

ハッシングを記号処理に適用する手法は、今迄数多く論じられてきた。[1, 2, 3] ハッシングを単一ヒットの連想記憶の実現手段と考える使用法と、複雑なデータ構造の共有と高速同定法が筆者らの研究グループで採用され、ソフトウェアのインプリメンテーションがなされている。

本稿では、既に練られてきたいくつかのハッシングの応用アルゴリズムを高速に実行するために、設計・製作されたハッシングハードウェアの機能と、具体的使用法の概略を紹介する。

§2. ハードウェアの設計方針

- データベースを格納する表（ハッシュ表）は、原則として、主記憶にとられるものとし、並列ハッシングアルゴリズム [4,5] を実現する。
- 記憶装置とCPUのインタフェイスに、ハッシュ番地によるアクセス機構と、鍵および特殊ビットパターンと読み出しデータとの比較機構を設ける。
- ハッシュ番地列を生成するハードウェア機構を、従来のアドレスマッピング機構と並置し、概念的には、ハッシュアクセスを、一つのアドレスマッピング法と見なしうるようにする。

§3. 試作装置概要

図1にハッシングハードウェアを持ったシステムを示す。太線で囲まれた部分が、ハッシングハードウェアとして付加された部分である。他の部分は、既存の16ビットのミニコンで、マイクロプログラムにより、PDP 11のアーキテクチャのエミュレートができるようになっている。

ハッシングハードウェアは、大別して、HAU（Hash Addressing Unit）とHM（Hash table Memories）に分割される。HAUはさらにHAG（Hash Address Generator）, HCG（Hash Code Generator）とHTDU（Hash Table Descriptor Unit）に分れる。（図2）

- HCGは鍵からHAGの入力となるビットパターン（ハッシュコード）を生成する。
- HAGはハッシュ番地列を生成するもので、次のアルゴリズムを実行する。

  c, c' をハッシュコードとし、P をハッシュ表のサイズとする。P は素数とする。m を $2^m > P > 2^{m-1}$ にとり、生成すべき、番地列 $h_0, h_1 \cdots h_{P-1}$ を次のように生成する。

```
h0 ← c ∧ (2^m − 1) , Δh ← c' ∧ (2^m − 1)
if h0 ≥ P, h0 ← h0 − P
if Δh ≥ P, Δh ← Δh − P
if Δh = 0, Δh ← 1
for i = 1, 2, ···, P − 1
    hi ← hi + Δh
    if hi ≥ P, hi ← hi − P
```

・HMは、通常の記憶装置としての機能の他、次の機能を合せ持つ。
HM1～4はHATによって起動され、同時に読み出し動作が行なわれる。
削除語(全1)，空語(全0)と鍵の一致検出を行なう。
・HTDUはハッシュ表のdescriptorを格納する。各descriptorはハッシュ表のベース，サイズ等の情報を格納する。これら情報はμプログラムの制御，HAG等に用いられる。

§4. ハッシング命令

表1にハッシング関連命令を示す。これらの命令はPDP 11の拡張命令として使用できるようになっている。表探索命令は，図3に示すような，5種類の構造のハッシュ表に対して、作用する。(表1に可能な組合せを示す)。図3で、single, double, quadruple表は各々，1，2，4語長(一語は16ビット)の鍵の表である。pairとはsingle鍵と、それに付随した値との対である。virtual型表は、複雑なデータ構造あるいは、4語長より長い鍵の処理を行なうために設けられたもので、virtual鍵と，実際の鍵へのポインタの対よりなる。

§5. 応用例.

(1) 記号表の操作

図4に記号表のデータ構造を示す。図中で、HT1はpair型ハッシュ表である。鍵が16ビットの時，鍵は表の鍵部に格納される。鍵が2倍長以上の場合はポインタが格納される。

(2) 複雑な構造の唯一コピーの生成

複雑な構造を、ハッシング命令で扱う方法として、次の2つの方法が考えられる。

リストによる実現

2倍長鍵型ハッシュ表を使用し、ポインタそのものを鍵として利用するものである。(後藤らのhconsによるリスト生成手法と同じ)

virtual鍵を用いる方法(図5参照)

ソフトウェアにより，データ構造を反映させたvirtual鍵を作り、その鍵をハッシングの鍵として利用する。virtual鍵では，データ構造と鍵の対応が一対一でない。このため，同一virtual鍵が、表中に2個以上登録されうる。ハッシングのハードウェアは，マルチヒットに対処し、しかも，ハッシュアドレス列の途中から探索を再開する機能を持つ必要がある。このため，実際には，実現されたハードウェアは、図1,2で示す機構にさらに付加機構が設けられている。

§6. 性能評価

表2に、ハッシングの最も基本的な命令であるハッシュ探索命令HSRの実行速度を示す。表2は、ハードウェアでな、純ソフトウェアによるハッシングに比べ約10倍の速度の向上が得られたことを示している。

§7. 終りに

本稿では、16ビットの既存アーキテクチャ(特にPDP11)に、ハッシングハードウェアを実現する方法を具体例の製作によって示した。本試作機の主たる目的は、非数値処理にハッシングを導入し、高速化する方法をハードウェアインプリメンテーションの側から探るものであり，かつ，より大規模計算機[6]のモデルインプリメンテーションである。(しかしながら，16ビットの計算機で十分な

仕事ができ、しかもハッシングを多用する応用に対しては、ここで示したミスラムは十分に実用的であると考えられる。

§8 参考文献

[1] Goto, E. Monocopy and associative algorithms in an extended Lisp, Tech. Rept. 74-30, Department of Information Science, University of Tokyo (1974)

[2] Goto, E. and Kanada, Y. Hashing lemmas on time complexity with application to formula manipulation, Proc. ACM-SYMSAC, 1976

[3] Feldman, J.A. and Rovner, P.D. An Algol-based associative language, CACM, Vol. 12, No. 8 (1968)

[4] Goto, E., Ida, T. and Gunji, T. Parallel hashing algorithms, Information Processing Letters, Vol. 6, No. 1 (1977)

[5] Ida, T. and Goto, E. Analysis of parallel hashing algorithms with key deletion, Journal of Information Processing, Vol. 1, No. 1 (1978)

[6] Goto, E. et. al. FLATS, a machine for numerical, symbolic and associative processing, Proc. 6th annual symposium on computer architecture (1979)

[7] Ida, T. Hashing Hardware and its application to symbol manipulation Proc. international workshop on high-level language computer architecture (1980)
(本文には引用されていない)

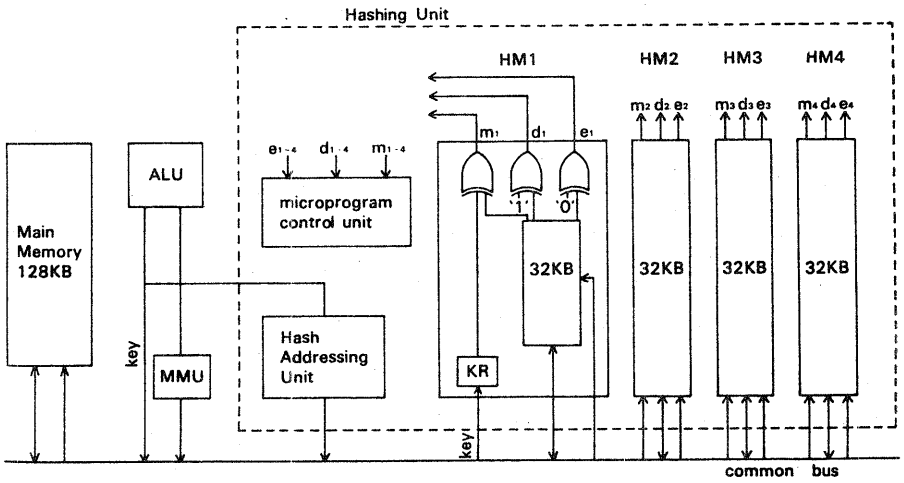


図 1. System with Hashing Hardware

図2 **Block Diagram of Hash Addressing Unit**

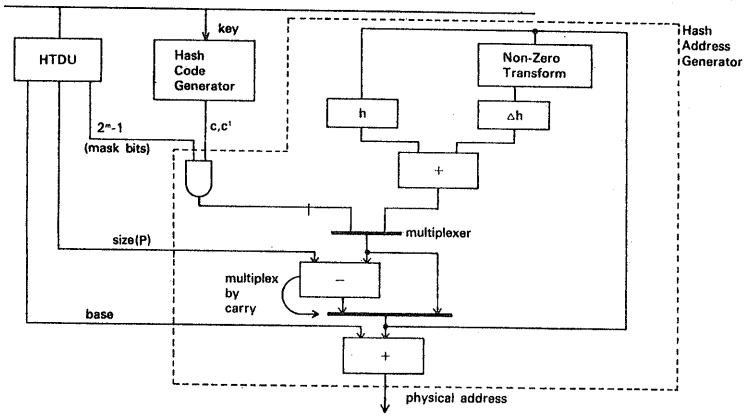


図3. ハッシュ表の型と構造

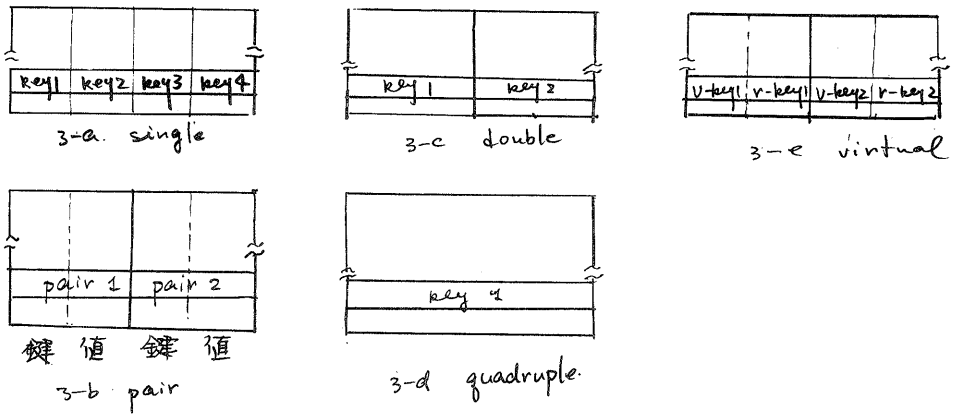
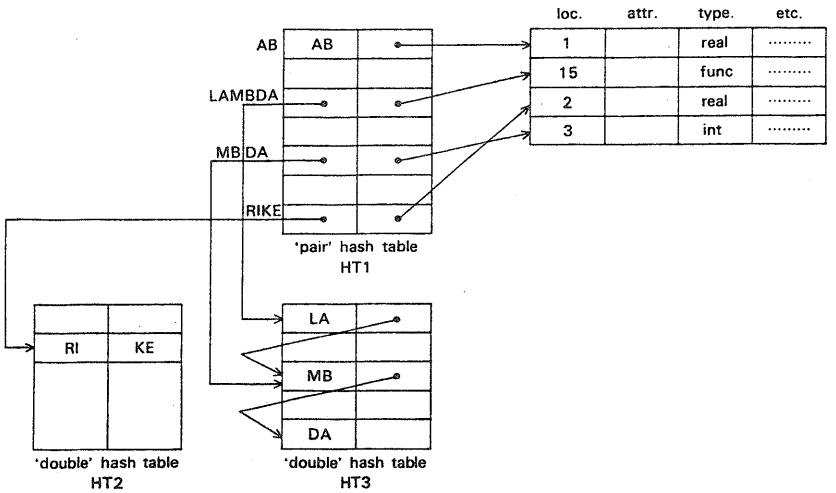


図4 **Representations of a Symbol Table**



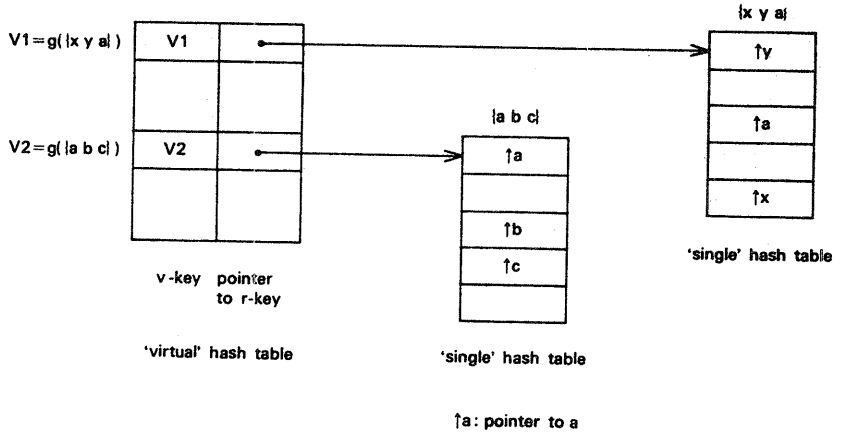


図5 virtual ハッシュ表を用いた集合の表現法

命令	機能	適用する表の型				
		single	double	quadruple	pair	virtual
HSR	Search key	×	×	×	×	
HGV	Get value of pair				×	
HPV	Put value of pair				×	
HNI	New key insert	×	×	×	×	
HSI	Search and insert	×	×	×	×	
HSD	Search and delete	×	×	×	×	
HGR	Get real-key					×
HGRN	Get real-key next					×
HPR	Put real-key					×
HDX	Delete existing v-key					×
HRTI	Return from hash interrupt					
PTHT	Put in hash table descriptor					
GTHT	Get from hash table descriptor					

表1. ハッシング命令一覧

	case 1H	case 2H	case 1S	case 2S
HSR for single keys	6.1	6.6	5.5×10	8.3×10
HSR for double keys	1.1×10	1.2×10	1.2×10^2	1.7×10^2
HSR for quad. keys	1.8×10	2.0×10	2.0×10^2	2.3×10^2

in μsec

1H: 50%詰った表のすべての鍵を参照した時のHSRの平均実行時間
2H: 80%
1S; 2Sは、PDP11の命令を利用して、ソフトウェアにより実行した時の平均実行時間
この値は、命令のフェッチ、デコードおよび、割り込みが生じた場合には、その処理時間
をも含む(1H, 2Hの場合)

表2. HSRの平均実行時間