Transactions Briefs

Encoding Don't Cares in Static and Dynamic Content-Addressable Memories

Sergio R. Ramírez-Chávez

Abstract—**A new encoding scheme for don't cares using two binary static content-addressable memory cells is presented. No extra hardware is required for the storage and retrieval of don't-cares beyond the hardware necessary for binary write and match operations found in most commercial static CAM's. Besides the availability of proven hardware for binary CAM chips, this implementation has the advantage that both binary and ternary values can coexist in a single CAM word.**

I. Introduction

Content addressable memories (CAM's) have been the subject of research since the early 1960's [1]. The extension of their ability to store three values, zero, one, and don't care, was reported in the early seventies [2]. More recently various associative processors have been implemented using this feature [6], [8].

The purpose of this paper is to present a new encoding scheme for zeros, ones, and don't-cares that can be used in a conventional static CAM with masking capability. This scheme is derived from the known fact that two bits are needed to encode three symbols (0, 1, and *), and that the behavior of the match line is a combinational function of the two bits stored in the CAM cells and the search argument.

The consequence of this encoding scheme is that currently available CAM chips with mask capability can be used for the storage and retrieval of don't-cares without any modification, and that specialized cells with don't-care capability [2], [7], [8] are not necessary as it was previously believed. While this conclusion might be a disappointment to researchers involved in finding ways to efficiently implement CAM arrays with don't-care capabilities, it is also a blessing since it makes available a variety of commercial CAM chips for use in their research without the need for access to VLSI fabrication facilities.

II. Need for Don't-Cares

Various uses of the storage of don't-cares have been reported in the literature. Applications range from image compression using quad-trees [3] to gate level simulation [6]. For illustrative purposes, we discuss how a large set of integers can be compressed using a CAM with don't care capability.

Set Storage and Interval Searching

The set operations used in Pascal can be easily supported by any binary CAM. Further compression of the stored data can be achieved if the CAM has the capability of storing wild cards (don't cares). Fig. 1 shows a Pascal set and a search of its contents using the **in** statement. Fig. 2(a) shows the encoding of this set in an 8-bit binary CAM and its response to the query in Fig. 1. Fig. 2(b) shows the encoding for a ternary CAM and its response to the same query.

In a conventional CAM the storage of don't cares is not

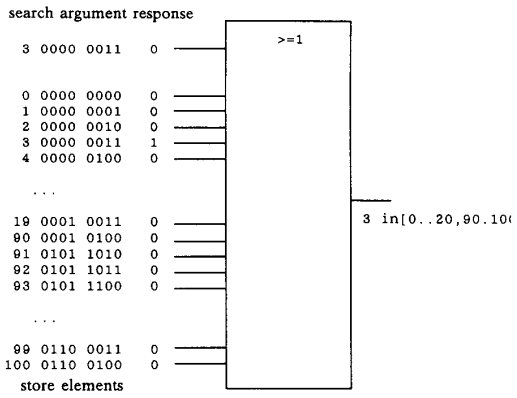
Manuscript received December 16, 1991; June 4, 1992. This paper was recommended by Associate Editor G. DeMicheli.

The author is with the Department of Electrical Engineering, Bucknell University, Lewisburg, PA 17837.

IEEE Log Number 9202967.

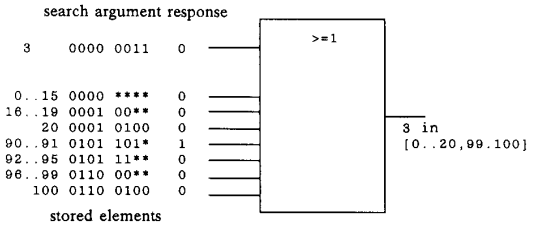
```
program  setexample (input,output)

type  intset = set of integer;

var x: intset;
var y: integer;

begin
   x := [0..20,90..100];

            ...

   readln (y);

            ...

if y in x then writeln('y found');

            ...

end.
```

Fig. 1. Pascal program illustrating a search on a set.



total storage = 32 words x 8 bits/word
= 256 bits

(a)



total storage = 7 words x 16 bits/word = 112 bits

(b)

Fig. 2. (a) Storage of sets in a binary CAM. (b) Storage of sets in a ternary CAM.

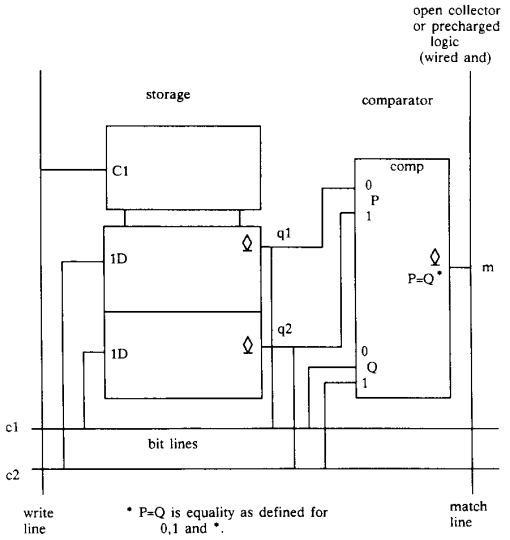


Fig. 3. IEEE-IEC standard diagram for a CAM cell with don't-care capability.

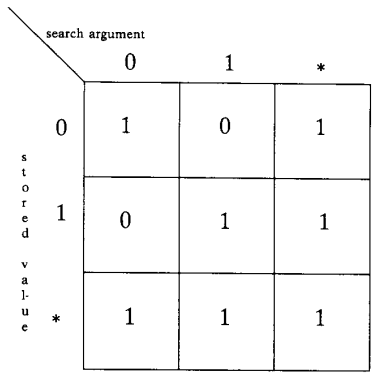


Fig. 4. Definition of equality (m, $p = Q$) for ternary values (zero, one, and don't care).

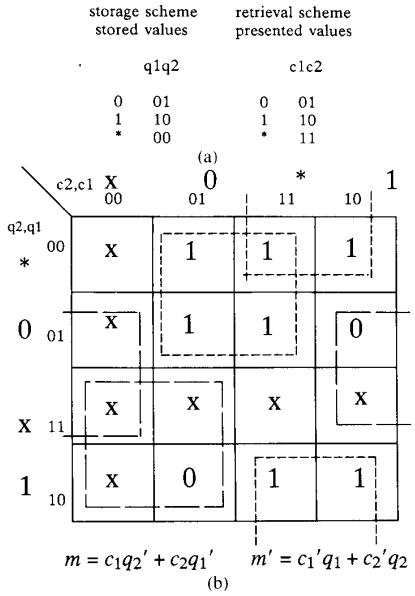


Fig. 5. Possible encoding scheme for: (a) CAM cell with don't-care capability, (b) resulting comparator Karnaugh map.

allowed, therefore the 32 members of the set [0..20, 90..100] must be stored. Since the word size is 8 bits, we need 256 bits to store the whole set. If the storage of don't-cares is allowed, the set can be split into intervals which exhibit regularity in the most significant bits (see Fig. 2(b)), then encoded using the ternary CAM's don't care capability. This capability allows us to store a subset such as [1..15] in a descriptive form like a number x such that x's four most significant bits are zero, rather than in an enumerative way. This compresses the interval from 16 8-bit words to one 8-trit (or 16-bit) word.

If large intervals are involved in the set contents, the number of bits required to store the set is significantly reduced if the storage of don't cares is allowed. For this particular case a 56% reduction in the required storage is achieved. Large reductions can be achieved if sets such as the whole integer set for a 32-bit computer must be stored (one word versus 2^{32} words). For small intervals the price paid by the ternary encoding, as well as by the software overhead incurred in decomposing the interval, might be too high to make this approach feasible.

III. A Formal Approach to the Don't-Care Encoding Scheme in a Conventional Static CAM

The architecture of a CAM with don't-care capability is determined by three factors. First, the fact that two bits of storage are required for the storage of three states; second, the particular encoding chosen for zero, one, and don't care; and third, the encoding of zero, one, and don't care when presented for comparison in a match operation. These two encoding schemes need not be the same. Fig. 3 shows the general architecture of a CAM with don't care capability. The stored values, $(q_1\, q_2)$ are compared against the search arguments (c_1,c_2) and a match is reported by the match output. Fig. 4 shows the truth table of the match operation. Word level searches can be performed by "wire anding" multiple match outputs.

The reader should notice the use of don't-cares both as comparison and as stored values. Normal binary CAM's support don't-cares only as comparison values. This operation is called masking.

Let us now design a static content addressable memory based on a CMOS static RAM. We choose the encoding schemes shown in Fig. 5(a) to represent the stored zero, one, and don't care as well as the values presented for comparison.

Fig. 5(b) shows the Karnaugh map resulting from such encoding scheme and the definition of the match operation. Minimization of the match function "m" results in

$$m = c_1 q_2' + c_2 q_1' \tag{1}$$

which is very readable but is not suitable for CMOS implementation. A more suitable form for CMOS precharged logic is the following:

$$m' = c_1' q_1 + c_2' q_2 \tag{2}$$

$$m = (c_1' q_1 + c_2' q_2)' \tag{3}$$

which can be directly implemented in CMOS precharged logic as shown in Fig. 6(a) and (b). Precharged logic is chosen for CAM

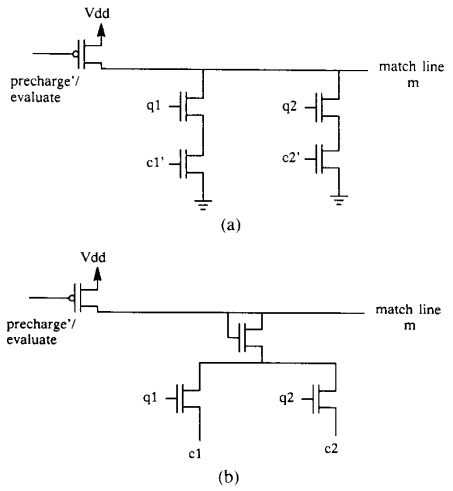


Fig. 6. (a) Resulting comparator designs from the encoding scheme in Fig. 5(a) precharged CMOS. (b) Precharged CMOS with pass transistors.

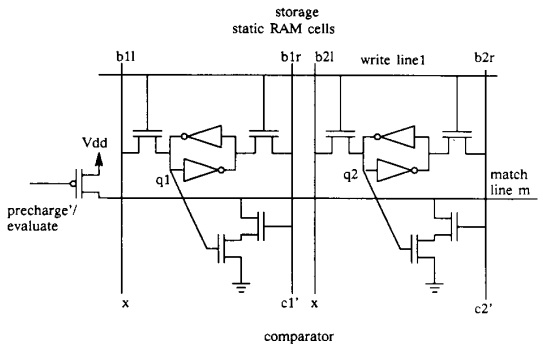


Fig. 7. Possible static CAM cell with don't care capability (comparator in Fig. 6(a) is used).

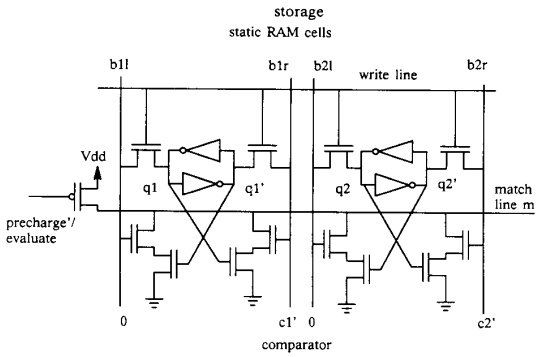


Fig. 8. Two adjacent static binary CAM cells.

storage scheme
stored values

	q1q2
0	01
1	10
*	00

(a)

retrieval scheme
presented values

presented ternary value	encoded value in the bit lines of two binary static CAMS.					binary CAM equivalent operation
	c1c2	b1l	b1r	b2l	b2r	l r
0	01	0	1	0	0	0 M*
1	10	0	0	0	1	M 0
*	11	0	0	0	0	M M

*M is the masking of a bit operation common in commercial binary CAMS.

(b)

Fig. 9. Encoding and retrieval schemes for don't-care in two static binary CAM's cells with masking capability. (a) Encoding scheme. (b) Retrieval scheme.

designs because of its ability to be "wire anded" for match operations.

Once the comparator is designed and the storage RAM cell is chosen it is easy to implement a specialized 2-bit static CAM with don't-care capability such as the one shown in Fig. 7. Other cell topologies are obtained if the storage or retrieval codes are changed; a different comparator (such as in Fig. 6(b)) is used; or the designer decides to share not only the match lines between the two cells but also the bit lines, resulting in a scheme that requires two RAM words plus the comparator to implement a ternary CAM word [8].

Since the purpose of this paper is not to design a new specialized CAM cell with don't-care capabilities but to show that currently available static CAM cells can perform the match operation defined in Fig. 4 and implemented by (1), (2), and (3), let us modify (3) to include the two extra minterms present in the comparators of two conventional static CAM's. Rewriting (3), we obtain

$$m = (0 \bullet q_1' + c_1' q_1 + 0 \bullet q_2' + c_2' q_2)' \qquad (4)$$

which can be easily implemented by two conventional static CAM's with don't care capability by keeping the left bit lines of both cells low during a match operation. The resulting configuration is shown in Fig. 8 and the retrieval scheme is interpreted in terms of the conventional binary CAM's mask operation in Fig. 9. The codes for presenting a zero, one and a don't care for a binary CAM which has encoded trits in cell pairs are 0M, M0, and MM where M indicates that the particular bit is not included in the comparison. This is the encoding scheme that we were searching for.

IV. Dynamic CAM's

If, instead of choosing a static RAM for storage, a one-transistor dynamic RAM is chosen and the comparator in Fig. 6(b) is used, the same design procedure would have led us to Mundy's cell [2] (see Fig. 10(a)). The same choices but with a retrieval scheme where 0 = 10 and 1 = 01 produces Wade and Sodini's cell [7] (see Fig. 10(b)), which has better electrical performance than Mundy's cell at the expense of an inversion in the comparison valve. These cells are the dynamic memory counterparts of the application presented in this paper. There is no one transistor dynamic RAM counterpart for a single static CAM cell because both the stored value (q) and its complement (q') are necessary for implementing the binary match function.

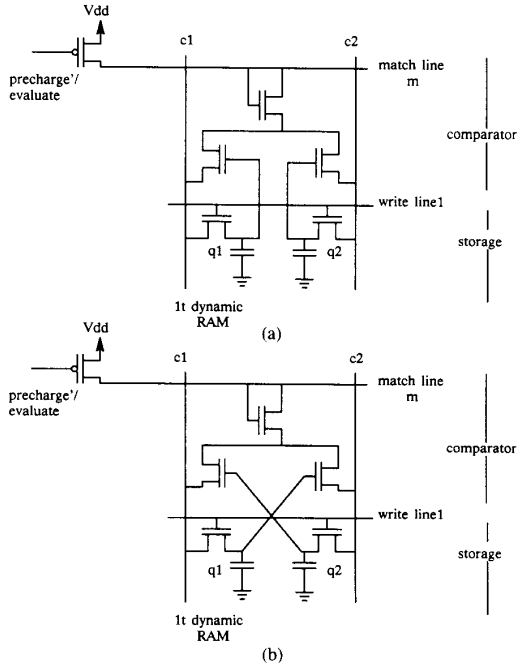


Fig. 10. Dynamic CAMs with don't-care capability. (a) Mundy's cell. (b) Wade and Sodini's cell.

The design procedure presented in this paper when applied to dynamic memories can be considered an *a posteriori* redesign of Mundy's and Wade's cells. The detailed synthesis procedure is left to the reader.

V. Conclusions

The result presented in this paper has the following consequences and advantages.

First, the CAM hardware necessary for storing and retrieving don't-cares is readily available as an off-the-shelf component. This expands the possibilities of many researchers and designers interested in CAM algorithms who may not have access to VLSI fabrication facilities.

Second, normal bit values and don't-care encoded values (trits) can coexist in the same word that gives some space savings when compared to specialized don't-care cells. This also adds versatility to the CAM system.

Third, no specialized hardware is required. Any standard CAM hardware which supports masking operations can store and retrieve don't-cares.

Fourth, the two cells representing three values need not be adjacent,[1] therefore the encoding scheme can be dynamic.

Fifth, don't-cares can be searched if treated as binary rather than ternary values.

References

[1] G. Estrin and R. Fuller, "Algorithms for content-addressable memories," in *Proc. IEEE Pacific Computer Conf.*, pp. 118–130, 1963.

[2] J. L. Mundy, "High Density Four Transistor MOS Content Addressable Memory," US patent 3 701 980, Oct. 31, 1972.

[3] J. V. Oldfield *et al.*, "Content addressable memories for quadtree-based images," *CASE Center Report*, Syracuse University, Syracuse, NY, 1988.

[4] S. R. Ramírez-Chávez, "Application of standard content addressable memory cells in the storage and retrieval of don't cares," *Disclosure of Invention*, 1990.

[5] H. Samet, "The quadtree and related hierarchical data structures," *ACM Computing Surveys*, vol. 16, pp. 187–260, June 1984.

[6] C. Sodini *et al.*, "The MIT database accelerator: A novel content addressable memory," *Proc. Wescon*, 1986.

[7] J. Wade and C. Sodini, "Dynamic cross-coupled bit-line content-addressable memory cell for high-density arrays," *IEEE J. Solid State Circuits*, vol. SC-22, Feb. 1987.

[8] Coherent Research Inc., "Coherent processor," *Ref. Manual*, Syracuse, NY, 1990.

[1] They are shown as adjacent cells just for illustrative purposes. The only requirement is that they share the same match line.

An l_1-Approximation Based Method for Synthesizing FIR Filters

Wen-Shyong Yu, I-Kong Fong, and Kuang-Chiung Chang

***Abstract*—In this paper, a method is proposed for the synthesis of digital filters having approximately the desired linear phase frequency responses. A mathematical optimization problem is formulated from the synthesis objective, and a theorem from the theory of l_1-approximation is used to convert the optimization problem such that it can be solved by the linear programming technique. The method is successfully applied to the synthesis of a digital FIR equalizer for a given analog antialiasing filter.**

I. Introduction

In the last three decades, many significant results and effective algorithms have been developed in the l_1-approximation theory [2]–[4]. Recently, these results are applied to the filter design problems for the purposes of smoothing and deconvolution [5]–[7], and to the optimal control problems for the purposes of stability robustification and disturbance rejection [8]. In this paper, an l_1-approximation based method is proposed for the synthesis of digital FIR filters. The objective is to determine the parameters of the filters such that their frequency responses are close to the specified ones. More specifically, we want to minimize the l_1-norm of the difference between the frequency response of the filter and the desired response. Hence we form a mathematical optimization problem whose equivalent problem, through the use of l_1-approximation theory, can be shown to be solvable by the linear programming technique. This makes the solution of the original problem feasible and effective. Besides, it is known [5]–[8] that filters synthesized by the method of l_1-norm approximation have the ability to reject bounded exogenous noise with persistent duration, while those synthesized by l_q-norm approximations, $1 < q \leq \infty$, have not. Finally, we implement the proposed method and apply it to the synthesis of an equalizer for a given analog antialiasing filter. The result shows its advantage when compared with the result given in [1].

II. Problem Formulation and Main Results

Let $D(\Omega)$, $|\Omega| \leq \pi$, be a complex-valued frequency response with linear phase characteristic which we want a digital filter to

Manuscript received March 5, 1991; revised January 27, 1992 and June 1, 1992. This paper was recommended by Associate Editor M. H. Etzel.

The authors are with the Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan 106.

IEEE Log Number 9202966.