Content-Addressable Memory (CAM) Circuits and Architectures: A Tutorial and Survey

Kostas Pagiamtzis, *Student Member, IEEE,* and Ali Sheikholeslami, *Senior Member, IEEE*

***Abstract*—We survey recent developments in the design of large-capacity content-addressable memory (CAM). A CAM is a memory that implements the lookup-table function in a single clock cycle using dedicated comparison circuitry. CAMs are especially popular in network routers for packet forwarding and packet classification, but they are also beneficial in a variety of other applications that require high-speed table lookup. The main CAM-design challenge is to reduce power consumption associated with the large amount of parallel active circuitry, without sacrificing speed or memory density. In this paper, we review CAM-design techniques at the circuit level and at the architectural level. At the circuit level, we review low-power matchline sensing techniques and searchline driving approaches. At the architectural level we review three methods for reducing power consumption.**

***Index Terms*—Bank selection, content-addressable memory (CAM), matchline pipelining, matchline sensing, NAND cell, NOR cell, review, searchline power.**

I. Introduction

A CONTENT-ADDRESSABLE memory (CAM) compares input search data against a table of stored data, and returns the address of the matching data [1]–[5]. CAMs have a single clock cycle throughput making them faster than other hardware- and software-based search systems. CAMs can be used in a wide variety of applications requiring high search speeds. These applications include parametric curve extraction [6], Hough transformation [7], Huffman coding/decoding [8], [9], Lempel–Ziv compression [10]–[13], and image coding [14]. The primary commercial application of CAMs today is to classify and forward Internet protocol (IP) packets in network routers [15]–[20]. In networks like the Internet, a message such an as e-mail or a Web page is transferred by first breaking up the message into small data packets of a few hundred bytes, and, then, sending each data packet individually through the network. These packets are routed from the source, through the intermediate nodes of the network (called routers), and reassembled at the destination to reproduce the original message. The function of a router is to compare the destination address of a packet to all possible routes, in order to choose the appropriate one. A CAM is a good choice for implementing this lookup operation due to its fast search capability.

Manuscript received May 19, 2005; revised October 12, 2005. This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) and an Ontario Graduate Scholarship in Science and Technology (OGSST).

The authors are with the Department of Electrical and Computer Engineering, University of Toronto, Toronto, ON M5S 3G4, Canada (e-mail: pagiamt@eecg.toronto.edu; ali@eecg.toronto.edu).

Digital Object Identifier 10.1109/JSSC.2005.864128

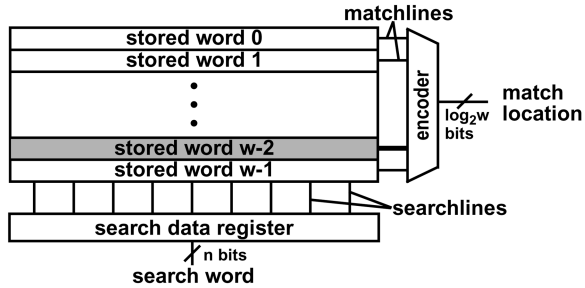

Fig. 1. Conceptual view of a content-addressable memory containing w words. In this example, the search word matches location $(w-2)$ as indicated by the shaded box. The matchlines provide the row match results. The encoder outputs an encoded version of the match location using $log_2 w$ bits.

However, the speed of a CAM comes at the cost of increased silicon area and power consumption, two design parameters that designers strive to reduce. As CAM applications grow, demanding larger CAM sizes, the power problem is further exacerbated. Reducing power consumption, without sacrificing speed or area, is the main thread of recent research in large-capacity CAMs. In this paper, we survey developments in the CAM area at two levels: circuits and architectures. Before providing an outline of this paper at the end of this section, we first briefly introduce the operation of CAM and also describe the CAM application of packet forwarding.

Fig. 1 shows a simplified block diagram of a CAM. The input to the system is the *search word* that is broadcast onto the *searchlines* to the table of stored data. The number of bits in a CAM word is usually large, with existing implementations ranging from 36 to 144 bits. A typical CAM employs a table size ranging between a few hundred entries to 32K entries, corresponding to an address space ranging from 7 bits to 15 bits. Each stored word has a *matchline* that indicates whether the search word and stored word are identical (the match case) or are different (a mismatch case, or miss). The matchlines are fed to an encoder that generates a binary *match location* corresponding to the matchline that is in the match state. An encoder is used in systems where only a single match is expected. In CAM applications where more than one word may match, a priority encoder is used instead of a simple encoder. A priority encoder selects the highest priority matching location to map to the match result, with words in lower address locations receiving higher priority. In addition, there is often a *hit* signal (not shown in the figure) that flags the case in which there is no matching location in the CAM. The overall function of a CAM is to take a search word and return the matching memory location. One can think of this operation as a fully programmable arbitrary mapping of the large space of the input search word to the smaller space of the output match location.

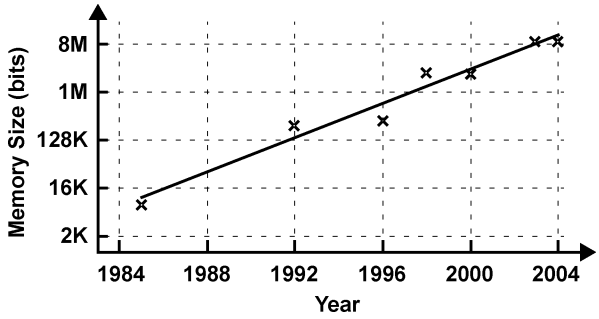

Fig. 2. Plot of CAM capacity (log scale) versus year of publication [21]–[27].

The operation of a CAM is like that of the tag portion of a fully associative cache. The tag portion of a cache compares its input, which is an address, to all addresses stored in the tag memory. In the case of match, a single matchline goes high, indicating the location of a match. Unlike CAMs, caches do not use priority encoders since only a single match occurs; instead, the matchline directly activates a read of the data portion of the cache associated with the matching tag. Many circuits are common to both CAMs and caches; however, we focus on large-capacity CAMs rather than on fully associative caches, which target smaller capacity and higher speed.

Today's largest commercially available single-chip CAMs are 18 Mbit implementations, although the largest CAMs reported in the literature are 9 Mbit in size [21], [22]. As a rule of thumb, the largest available CAM chip is usually about half the size of the largest available SRAM chip. This rule of thumb comes from the fact that a typical CAM cell consists of two SRAM cells, as we will see shortly. Fig. 2 plots (on a logarithmic scale) the capacity of published CAM [21]–[27] chips versus time from 1985 to 2004, revealing an exponential growth rate typical of semiconductor memory circuits and the factor-of-two relationship between SRAM and CAM.

A. Packet Forwarding Using CAM

We describe the application of CAMs to packet forwarding in network routers. First, we briefly summarize packet forwarding and then show how a CAM implements the required operations. Further examples of approaches that use CAMs for the purposes of sorting and searching are provided in [28], [29].

Network routers forward data packets from an incoming port to an outgoing port, using an address-lookup function. The address-lookup function examines the destination address of the packet and selects the output port associated with that address. The router maintains a list, called the routing table, that contains destination addresses and their corresponding output ports. An example of a simplified routing table is displayed in Table I. All four entries in the table are 5-bit words, with the *don't care bit*, "X", matching both a 0 and a 1 in that position. Because of the "X" bits, the first three entries in the Table represent a range of input addresses, i.e., entry 1 maps all addresses in the range 10100 to 10111 to port A. The router searches this table for the destination address of each incoming packet, and selects the appropriate output port. For example, if the router receives a packet with the destination address 10100, the packet is forwarded to port A. In the case of the incoming address 01101, the address

TABLE I
EXAMPLE ROUTING TABLE

Entry No.	Address (Binary)	Output Port
1	101XX	A
2	0110X	B
3	011XX	C
4	10011	D

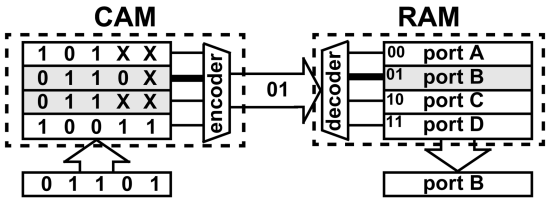

Fig. 3. CAM-based implementation of the routing table of Table I.

lookup matches both entry 2 and entry 3 in the table. Entry 2 is selected since it has the fewest "X" bits, or, alternatively, it has the longest prefix, indicating that it is the most direct route to the destination. This lookup method is called longest-prefix matching.

Fig. 3 illustrates how a CAM accomplishes address lookup by implementing the routing table shown in Table I. On the left of Fig. 3, the packet destination-address of 01101 is the input to the CAM. As in the table, two locations match, with the (priority) encoder choosing the upper entry and generating the match location 01, which corresponds to the most-direct route. This match location is the input address to a RAM that contains a list of output ports, as depicted in Fig. 3. A RAM read operation outputs the port designation, port B, to which the incoming packet is forwarded. We can view the match location output of the CAM as a pointer that retrieves the associated word from the RAM. In the particular case of packet forwarding the associated word is the designation of the output port. This CAM/RAM system is a complete implementation of an address-lookup engine for packet forwarding.

B. CAM Basics

We now take a more detailed look at CAM architecture. A small model is shown in Fig. 4. The figure shows a CAM consisting of 4 words, with each word containing 3 bits arranged horizontally (corresponding to 3 CAM cells). There is a matchline corresponding to each word (ML_0, ML_1, etc.) feeding into matchline sense amplifiers (MLSAs), and there is a differential searchline pair corresponding to each bit of the search word (SL_0, $\overline{\mathrm{SL}_0}$, SL_1, $\overline{\mathrm{SL}_1}$, etc.). A CAM search operation begins with loading the search-data word into the search-data registers followed by precharging all matchlines high, putting them all temporarily in the match state. Next, the searchline drivers broadcast the search word onto the differential searchlines, and each CAM core cell compares its stored bit against the bit on its corresponding searchlines. Matchlines on which all bits match remain in the precharged-high state. Matchlines that have at least one bit that misses, discharge to ground. The MLSA then detects whether its matchline has a matching condition or

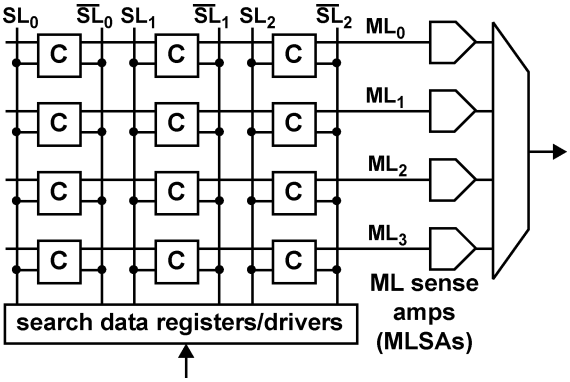

Fig. 4. Simple schematic of a model CAM with 4 words having 3 bits each. The schematic shows individual core cells, differential searchlines, and matchline sense amplifiers (MLSAs).

miss condition. Finally, the encoder maps the matchline of the matching location to its encoded address.

We organize the remainder of this paper by expanding upon the structure of the model in Fig. 4. In the next section, Section II, we begin our survey by looking at the two common CAM cells, the NOR cell and the NAND cell, each of which can be used as the basic building block in Fig. 4. The section continues by examining the combination of CAM cells to construct a full matchline. Section III describes techniques for detecting whether a matchline has a match or a miss. This section also discusses matchline power consumption, and compares the reviewed techniques in terms of the power savings they provide. In Section IV, we turn our attention to the searchlines, and examine proposals that save searchline power. Section V reviews three architectural techniques that save power. Finally, in Section VII, we explore future directions for CAM research.

II. Core Cells and Matchline Structure

A CAM cell serves two basic functions: bit storage (as in RAM) and bit comparison (unique to CAM). Fig. 5 shows a NOR-type CAM cell [Fig. 5(a)] and the NAND-type CAM cell [Fig. 5(b)]. The bit storage in both cases is an SRAM cell where cross-coupled inverters implement the bit-storage nodes D and $\overline{\mathrm{D}}$. To simplify the schematic, we omit the nMOS access transistors and bitlines which are used to read and write the SRAM storage bit. Although some CAM cell implementations use lower area DRAM cells [27], [30], typically, CAM cells use SRAM storage. The bit comparison, which is logically equivalent to an XOR of the stored bit and the search bit is implemented in a somewhat different fashion in the NOR and the NAND cells.

A. NOR Cell

The NOR cell implements the comparison between the complementary stored bit, D (and $\overline{\mathrm{D}}$), and the complementary search data on the complementary searchline, SL (and $\overline{\mathrm{SL}}$), using four comparison transistors, M_1 through M_4, which are all typically minimum-size to maintain high cell density. These transistors implement the pulldown path of a dynamic XNOR logic gate with inputs SL and D. Each pair of transistors, $\mathrm{M}_1/\mathrm{M}_3$ and $\mathrm{M}_2/\mathrm{M}_4$, forms a pulldown path from the matchline, ML, such that a mismatch of SL and D activates least one of the pulldown paths,

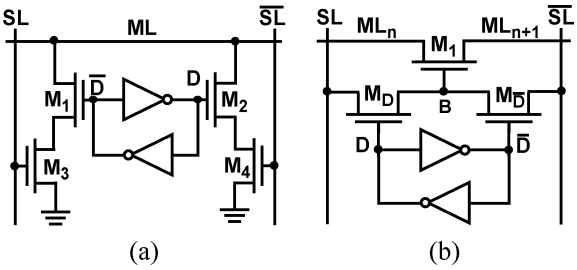

Fig. 5. CAM core cells for (a) 10-T NOR-type CAM and (b) 9-T NAND-type CAM [26]. The cells are shown using SRAM-based data-storage cells. For simplicity, the figure omits the usual SRAM access transistors and associated bitlines. The SRAM storage and access transistors account for six of the cell transistors.

connecting ML to ground. A match of SL and D disables both pulldown paths, disconnecting ML from ground. The NOR nature of this cell becomes clear when multiple cells are connected in parallel to form a CAM word by shorting the ML of each cell to the ML of adjacent cells. The pulldown paths connect in parallel resembling the pulldown path of a CMOS NOR logic gate. There is a match condition on a given ML only if every individual cell in the word has a match.

B. NAND Cell

The NAND cell implements the comparison between the stored bit, D, and corresponding search data on the corresponding searchlines, (SL, $\overline{\mathrm{SL}}$), using the three comparison transistors M_1, M_D, and $\mathrm{M}_{\overline{\mathrm{D}}}$, which are all typically minimum-size to maintain high cell density. We illustrate the bit-comparison operation of a NAND cell through an example. Consider the case of a match when $\mathrm{SL} = 1$ and $\mathrm{D} = 1$. Pass transistor M_D is ON and passes the logic "1" on the SL to node B. Node B is the *bit-match* node which is logic "1" if there is a match in the cell. The logic "1" on node B turns ON transistor M_1. Note that M_1 is also turned ON in the other match case when $\mathrm{SL} = 0$ and $\mathrm{D} = 0$. In this case, the transistor $\mathrm{M}_{\overline{\mathrm{D}}}$ passes a logic high to raise node B. The remaining cases, where $\mathrm{SL} \neq \mathrm{D}$, result in a miss condition, and accordingly node B is logic "0" and the transistor M_1 is OFF. Node B is a pass-transistor implementation of the XNOR function $\mathrm{SL} \odot \mathrm{D}$. The NAND nature of this cell becomes clear when multiple NAND cells are serially connected. In this case, the ML_n and ML_{n+1} nodes are joined to form a word. A serial nMOS chain of all the M_i transistors resembles the pulldown path of a CMOS NAND logic gate. A match condition for the entire word occurs only if every cell in a word is in the match condition.

An important property of the NOR cell is that it provides a full rail voltage at the gates of all comparison transistors. On the other hand, a deficiency of the NAND cell is that it provides only a reduced logic "1" voltage at node B, which can reach only $V_{\mathrm{DD}} - V_{tn}$ when the searchlines are driven to V_{DD} (where V_{DD} is the supply voltage and V_{tn} is the nMOS threshold voltage).

C. Cell Variants

Fig. 6 shows a variant of the NOR cell [Fig. 6(a)] and a variant of the NAND cell [Fig. 6(b)]. The NOR cell variant uses only 9-transistors compared to the previous 10-T NOR cell. The bit comparison uses pass transistors (as in the previous 9-T NAND

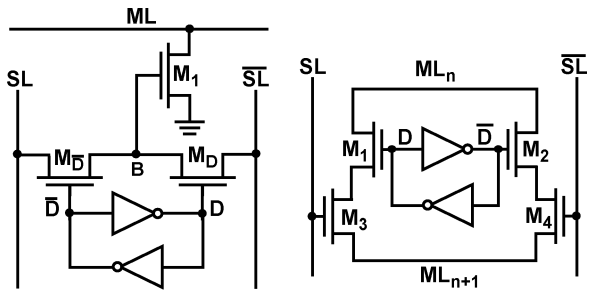

Fig. 6. CAM core cells variations for (a) 9-T NOR-type CAM and (b) 10-T NAND-type CAM. The cells are shown using SRAM-based data-storage cells. For simplicity, the figure omits the usual SRAM access transistors and associated bitlines. The SRAM storage and access transistors account for six of the cell transistors.

cell), however, the NOR property of this cell is apparent when multiple cells are connected in parallel to form a CAM word by shorting the ML of each cell to the ML of adjacent cells. The pulldown paths are in parallel just as in a CMOS NOR logic gate. Although, this 9-T NOR cell has only a single transistor in the pulldown path, the node B can only rise to $V_{\mathrm{DD}} - V_{tn}$ so usually the 10-T NOR cell is preferred.

The 10-T NAND cell [Fig. 6(b)] is a variant of the previous 9-T NAND cell. When the bit comparison succeeds in this cell, one of the transistor paths between $\mathrm{ML_n}$ and $\mathrm{ML_{n+1}}$ is ON. Thus, when multiple cells are shorted together these transistor paths appear in series just as in the pulldown network of a CMOS NAND gate. Since this NAND cell doubles the number of transistors in series, the 9-T NAND is usually preferred.

For the remainder of this paper we discuss only the 9-T NAND cell and the 10-T NOR cells as they are in predominant use today.

D. Ternary Cells

The NOR and NAND cells that have been presented are binary CAM cells. Such cells store either a logic "0" or a logic "1". Ternary cells, in addition, store an "X" value. The "X" value is a *don't care*, that represents both "0" and "1", allowing a wildcard operation. Wildcard operation means that an "X" value stored in a cell causes a match regardless of the input bit. As discussed earlier, this is a feature used in packet forwarding in Internet routers.

A ternary symbol can be encoded into two bits according to Table II. We represent these two bits as D and $\overline{\mathrm{D}}$. Note that although the D and $\overline{\mathrm{D}}$ are not necessarily complementary, we maintain the complementary notation for consistency with the binary CAM cell. Since two bits can represent 4 possible states, but ternary storage requires only three states, we disallow the state where D and $\overline{\mathrm{D}}$ are both zero. To store a ternary value in a NOR cell, we add a second SRAM cell, as shown in Fig. 7. One bit, D, connects to the left pulldown path and the other bit, $\overline{\mathrm{D}}$, connects to the right pulldown path, making the pulldown paths independently controlled. We store an "X" by setting both D and $\overline{\mathrm{D}}$ equal to logic "1", which disables both pulldown paths and forces the cell to match regardless in the inputs. We store a logic "1" by setting $\mathrm{D} = 1$ and $\overline{\mathrm{D}} = 0$ and store a logic "0" by setting $\mathrm{D} = 0$ and $\overline{\mathrm{D}} = 1$. In addition to storing an "X", the cell allows searching for an "X" by setting both SL and $\overline{\mathrm{SL}}$ to logic "0". This is an external *don't care* that forces a match of a bit regardless of the stored bit. Although storing an "X" is possible only in ternary CAMs, an external "X" symbol possible in both binary and ternary CAMs. In cases where ternary operation is needed but only binary CAMs are available, it is possible to emulate ternary operation using two binary cells per ternary symbol [31].

TABLE II
TERNARY ENCODING FOR NOR CELL

	Stored		Search Bit	
Stored Value	D	$\overline{\mathrm{D}}$		
0	0	1	0	1
1	1	0	1	0
X	1	1	0	0

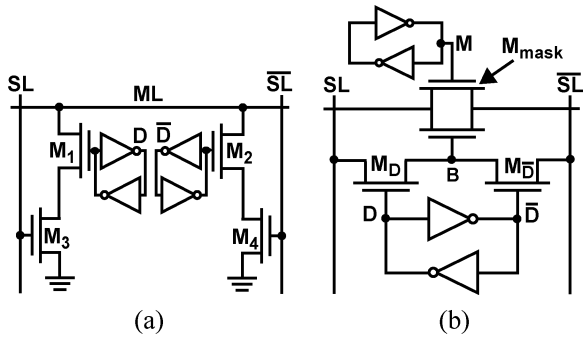

Fig. 7. Ternary core cells for (a) NOR-type CAM and (b) NAND-type CAM [32], [33].

As a modification to the ternary NOR cell of Fig. 7(a), Roth *et al.* [22] propose implementing the pulldown transistors $\mathrm{M_1}$–$\mathrm{M_4}$ using pMOS devices and complementing the logic levels of the searchlines and matchlines accordingly. Using pMOS transistors (instead of nMOS transistors) for the comparison circuitry allows for a more compact layout, due to reducing the number of spacings of p-diffusions to n-diffusions in the cell. In addition to increased density, the smaller area of the cell reduces wiring capacitance and therefore reduces power consumption. The tradeoff that results from using minimum-size pMOS transistors, rather than minimum-size nMOS transistors, is that the pulldown path will have a higher equivalent resistance, slowing down the search operation. We discuss power and operating speed in detail in Sections III and IV.

A NAND cell can be modified for ternary storage by adding storage for a mask bit at node M, as depicted in Fig. 7(b) [32], [33]. When storing an "X", we set this mask bit to "1". This forces transistor $\mathrm{M_{mask}}$ ON, regardless of the value of D, ensuring that the cell always matches. In addition to storing an "X", the cell allows searching for an "X" by setting both SL and $\overline{\mathrm{SL}}$ to logic "1". Table III lists the stored encoding and search-bit encoding for the ternary NAND cell.

Further minor modifications to CAM cells include mixing parts of the NAND and NOR cells, using dynamic-threshold techniques in silicon-on-insulator (SOI) processes, and alternating the logic level of the pulldown path to ground in the NOR cell [34]–[36].

Currently, the NOR cell and the NAND cell are the prevalent core cells for providing storage and comparison circuitry in

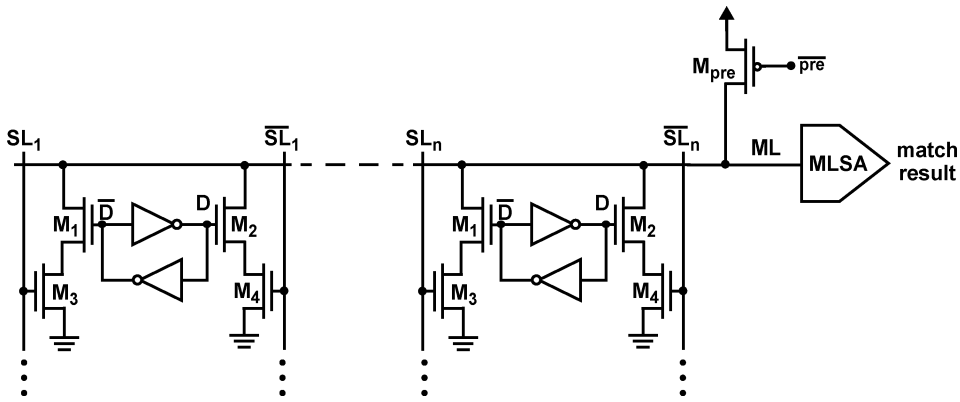

Fig. 8. Structure of a NOR matchline with n cells. Transistor $\mathrm{M_{pre}}$ precharges the matchline and the MLSA evaluates the state of the matchline, generating the match result.

TABLE III
TERNARY ENCODING FOR NAND CELL

	Stored		Search Bit	
Value	D	M	SL	$\overline{\mathrm{SL}}$
0	0	0	0	1
1	1	0	1	0
X	0	1	1	1
X	1	1	1	1

CMOS CAMs. For a comprehensive survey of the precursors of CMOS CAM cells refer to [37].

E. Matchline Structures

We now demonstrate the NAND cell and NOR cell in constructing a CAM matchline. The matchline is one of the two key structures in CAMs. The other is the searchline, which we will examine in Section IV.

1) NOR Matchline: Fig. 8 depicts, in schematic form, how NOR cells are connected in parallel to form a NOR matchline, ML. While we show binary cells in the figure, the description of matchline operation applies to both binary and ternary CAM.

A typical NOR search cycle operates in three phases: searchline precharge, matchline precharge, and matchline evaluation. First, the searchlines are precharged low to disconnect the matchlines from ground by disabling the pulldown paths in each CAM cell. Second, with the pulldown paths disconnected, the $\mathrm{M_{pre}}$ transistor precharges the matchlines high. Finally, the searchlines are driven to the search word values, triggering the matchline evaluation phase. In the case of a match, the ML voltage, V_{ML}, stays high as there is no discharge path to ground. In the case of a miss, there is at least one path to ground that discharges the matchline. The matchline sense amplifier (MLSA) senses the voltage on ML, and generates a corresponding full-rail output *match result*. We will see several variations of this scheme for evaluating the state of NOR matchlines in Section III.

The main feature of the NOR matchline is its high speed of operation. In the slowest case of a one-bit miss in a word, the critical evaluation path is through the two series transistors in the cell that form the pulldown path. Even in this worst case, NOR-cell evaluation is faster than the NAND case, where between 8 and 16 transistors form the evaluation path.

2) NAND Matchline: Fig. 9 shows the structure of the NAND matchline. A number of cells, n, are cascaded to form the matchline (this is, in fact, a match *node*, but for consistency we will refer to it as ML). For the purpose of explanation, we use the binary version of the NAND cell, but the same description applies to the case of a ternary cell.

On the right of the figure, the precharge pMOS transistor, $\mathrm{M_{pre}}$, sets the initial voltage of the matchline, ML, to the supply voltage, V_{DD}. Next, the evaluation nMOS transistor, $\mathrm{M_{eval}}$, turns ON. In the case of a match, all nMOS transistors $\mathrm{M_1}$ through M_n are ON, effectively creating a path to ground from the ML node, hence discharging ML to ground. In the case of a miss, at least one of the series nMOS transistors, $\mathrm{M_1}$ through $\mathrm{M_n}$, is OFF, leaving the ML voltage high. A sense amplifier, MLSA, detects the difference between the match (low) voltage and the miss (high) voltage. The NAND matchline has an explicit evaluation transistor, $\mathrm{M_{eval}}$, unlike the NOR matchline, where the CAM cells themselves perform the evaluation.

There is a potential charge-sharing problem in the NAND matchline. Charge sharing can occur between the ML node and the intermediate $\mathrm{ML_i}$ nodes. For example, in the case where all bits match except for the leftmost bit in Fig. 9, during evaluation there is charge sharing between the ML node and nodes $\mathrm{ML_1}$ through $\mathrm{ML_{n-1}}$. This charge sharing may cause the ML node voltage to drop sufficiently low such that the MLSA detects a false match. A technique that eliminates charge sharing is to precharge high, in addition to ML, the intermediate match nodes $\mathrm{ML_1}$ through $\mathrm{ML_{n-1}}$. This is accomplished by setting to V_{DD} the searchlines, $\mathrm{SL_1}$–$\mathrm{SL_n}$, and their complements $\overline{\mathrm{SL}}_1$–$\overline{\mathrm{SL}}_n$, which forces all transistors in the chain, $\mathrm{M_1}$–$\mathrm{M_n}$, to turn ON and precharge the intermediate nodes. When this precharge of the intermediate match nodes is complete, the searchlines are set to the data values corresponding to the incoming search word. This procedure eliminates charge sharing, since the intermediate match nodes and the ML node are initially shorted. However, there is an increase in the power consumption due to the searchline precharge which we will discuss Section IV.

A feature of the NAND matchline is that a miss stops signal propagation such that there is no consumption of power past the

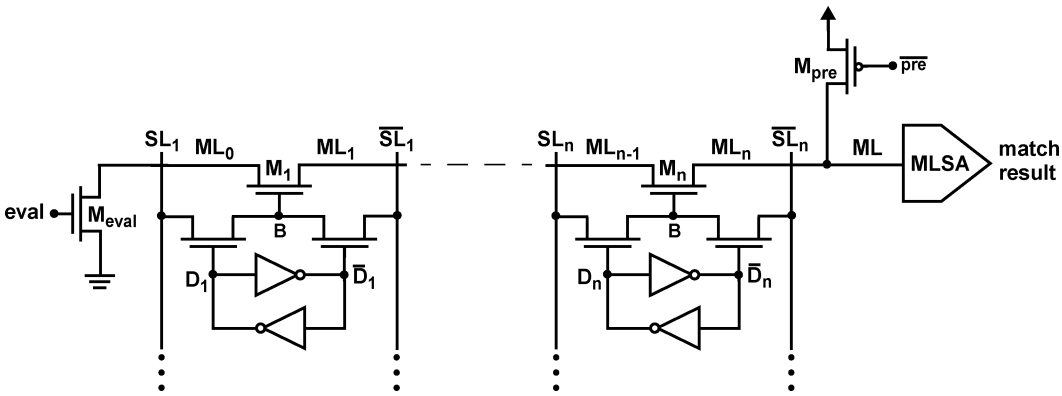

Fig. 9. NAND matchline structure with precharge and evaluate transistors.

final matching transistor in the serial nMOS chain. Typically, only one matchline is in the match state, consequently most matchlines have only a small number of transistors in the chain that are ON and thus only a small amount of power is consumed. Two drawbacks of the NAND matchline are a quadratic delay dependence on the number of cells, and a low noise margin. The quadratic delay-dependence comes from the fact that adding a NAND cell to a NAND matchline adds both a series resistance due to the series nMOS transistor and a capacitance to ground due to the nMOS diffusion capacitance. These elements form an RC ladder structure whose overall time constant has a quadratic dependence on the number of NAND cells. Most implementations limit the number of cells on a NAND matchline to 8 to 16 in order to limit the quadratic degradation in speed [38]. The low noise margin is caused by the use of nMOS pass transistors for the comparison circuitry. Since the gate voltage of the NAND matchline transistors ($\mathrm{M_1}$ through $\mathrm{M_n}$) when conducting, in Fig. 9, is $(V_{\mathrm{DD}} - V_{tn})$, the highest voltage that is passed on the matchline is $(V_{\mathrm{DD}} - 2V_{tn})$, (where V_{tn} is the threshold voltage of the nMOS transistor, augmented by the body effect). NOR cells avoid this problem by applying maximum gate voltage to all CAM cell transistors when conducting. One implementation of a NAND-based CAM reclaims some noise margin by employing the bootstrap effect by reversing the polarity of the matchline precharge and evaluate [26], [39].

Since NOR cells are more prevalent in today's CAM, we use the NOR cell and NOR matchline in the remainder of this paper, except where otherwise noted.

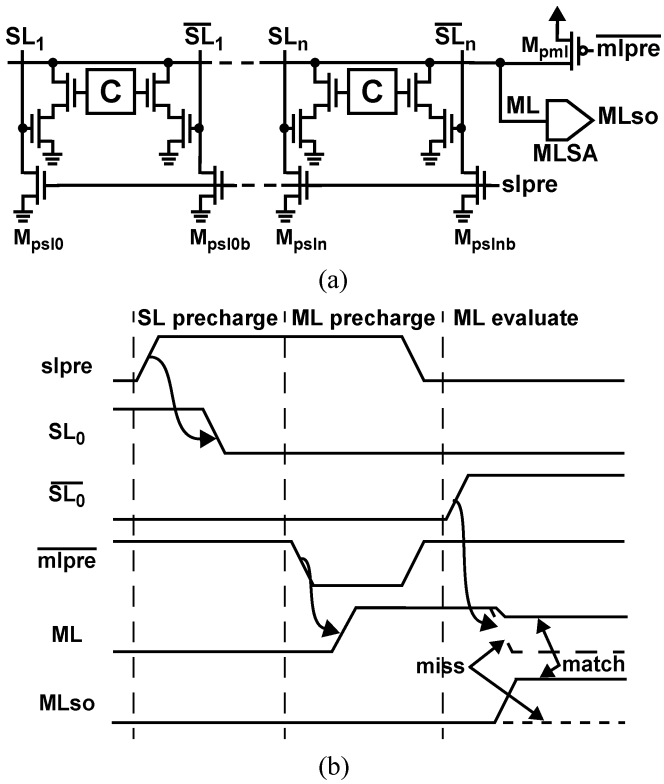

Fig. 10. This figure shows: (a) the schematic with precharge circuitry for matchline sensing using the precharge-high scheme, and (b) the corresponding timing diagram showing relative signal transitions.

III. Matchline Sensing Schemes

This section reviews matchline sensing schemes that generate the match result. First, we review the conventional precharge-high scheme, then introduce several variations that save power.

A. Conventional (Precharge-High) Matchline Sensing

We review the basic operation of the conventional precharge-high scheme and look at sensing speed, charge sharing, timing control and power consumption.

1) Basic Operation: The basic scheme for sensing the state of the NOR matchline is first to precharge high the matchline and then evaluate by allowing the NOR cells to pull down the matchlines in the case of a miss, or leave the matchline high in the case of a match. Fig. 10(a) shows, in schematic form, an implementation of this matchline-sensing scheme. The figure augments Fig. 8 by explicitly adding the searchline precharge transistors. Fig. 10(b) shows the signal timing which is divided into three phases: SL precharge, ML precharge, and ML evaluation. The operation begins by asserting slpre to precharge the searchlines low, disconnecting all the pull down paths in the NOR cells. With the pull down paths disconnected, the operation continues by asserting $\overline{\mathrm{mlpre}}$ to precharge the matchline high. Once the matchline is high, both slpre and $\overline{\mathrm{mlpre}}$ are de-asserted. The ML evaluate phase begins by placing the search word on the searchlines. If there is at least one single-bit miss on the matchline, a path (or multiple paths) to ground will discharge the matchline, ML, indicating a miss for the entire word, which is output on the MLSA

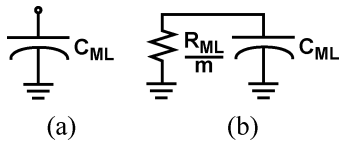

Fig. 11. Matchline circuit model for (a) the match state and (b) the miss state.

sense-output node, called MLso. If all bits on the matchline match, the matchline will remain high indicating a match for the entire word. Using this sketch of the precharge high scheme, we will investigate the performance of matchline in terms of speed, robustness, and power consumption. The matchline power dissipation is one of the major sources of power consumption in CAM. The other major source of power dissipation, the searchlines, is discussed in Section IV.

2) Matchline Model: To facilitate the analysis of matchline-sensing schemes, we desire a simple matchline circuit model. As shown in Fig. 11, the model for the match state of the matchline is a capacitor $C_{\rm ML}$ and the model for the miss state of the matchline is a capacitor, $C_{\rm ML}$, in parallel with a pulldown resistor $R_{\rm ML}/m$, where m is the number of bits that miss on the matchline. Referring to Fig. 10(a), the matchline capacitance, $C_{\rm ML}$, consists of the matchline wiring capacitance, the NOR-cell diffusion capacitance of transistors $\rm M_1$ and $\rm M_2$ [of Fig. 7(a)], the diffusion capacitance of precharge transistors, and the input capacitance of the MLSA. For misses, the equivalent matchline resistance $R_{\rm ML}$ varies with the number of bits, m, that miss in a word; however, for the purpose of analysis we use the worst-case (largest) resistance, which occurs when there is a 1-bit miss (that is $m = 1$).

3) Matchline Delay: Using the simple matchline model, we can find the time required to precharge and evaluate the matchline. The time to precharge the matchline (which we define as the 10% to 90% risetime of ML) through the precharge device $\rm M_{pml}$ of Fig. 10(a) is given by

$$t_{\rm MLpre} = 2.2\tau_{\rm MLpre} \tag{1}$$

$$= 2.2R_{\rm EQpre}C_{\rm ML} \tag{2}$$

where $R_{\rm EQpre}$ is the equivalent resistance of the precharge transistor.

The time to evaluate the matchline depends on the matchline capacitance and the matchline pulldown resistance. The worst-case matchline pulldown resistance occurs when only a single bit misses, activating only a single pulldown path. Referring to Fig. 11, for a single miss, $m = 1$, and the time for the evaluation, which we define as the time for the matchline to fall to 50% of the precharge voltage, is given by

$$t_{\text{ML-eval}} = 0.69\tau_{\text{ML-eval}} \tag{3}$$

$$= 0.69R_{\rm ML}C_{\rm ML}. \tag{4}$$

Since, typically, minimum-sized devices are used in the cell, the pulldown resistance is in the range of 5–10 kΩ in a 0.18-μm CMOS technology. The capacitance, $C_{\rm ML}$, depends on the number of bits on the matchline, but can be as high as a few hundred fF in a typical 0.18-μm CMOS technology.

4) Charge Sharing: There is a potential charge-sharing problem depending on whether the CAM storage bits D and $\overline{\rm D}$ are connected to the top transistor or the bottom transistor in the pulldown path. Fig. 12 shows these two possible configurations

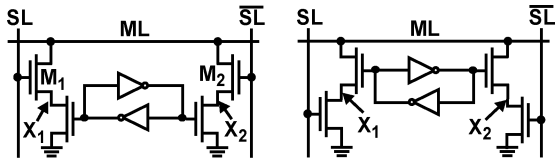

Fig. 12. Two possible configurations for the NOR cell: (a) the stored bit is connected to the bottom transistors of the pulldown pair, and (b) the stored bit is connected to the top transistors of the pulldown pair.

of the NOR cell. In the configuration of Fig. 12(a), there is a charge-sharing problem between the matchline, ML, and nodes $\rm X_1$ and $\rm X_2$. Charge sharing occurs during matchline evaluation, which occurs immediately after the matchline precharge-high phase. During matchline precharge, SL and $\overline{\rm SL}$ are both at ground. Once the precharge completes, one of the searchlines is activated, depending on the search data, causing either $\rm M_1$ or $\rm M_2$ to turn ON. This shares the charge at node $\rm X_1$ or node $\rm X_2$ with that of ML, causing the ML voltage, $V_{\rm ML}$, to drop, even in the case of match, which may lead to a sensing error. To avoid this problem, designers use the configuration shown in Fig. 12(b), where the stored bit is connected to the top transistors. Since the stored bit is constant during a search operation, charge sharing is eliminated.

5) Replica Control: A standard issue in CAM design, as in all memory design, is how to control the timing of clocked circuits. In the case of the precharge-high scheme, the signals that must be generated are slpre, $\overline{\rm mlpre}$, and a clock for MLSA. References [26], [39] use a replica matchline to generate timing signals in a similar fashion to that used in replica wordlines and bitlines in traditional memory design [40]. The replica matchline controls the length of precharge and evaluation phases during a CAM cycle minimizing the effect of process variation since the replica matchline loading tracks the variations in the rest of the array.

Fig. 13 is a simplified block diagram showing how a replica matchline may be used to control the precharge and evaluate timing. A replica word is programmed so that its matchline is in the (slowest case) one-bit miss state on every cycle regardless of the input data word. The transition on the replica matchline, $\rm ML_{REPLICA}$, is used to generate the shutoff signal which latches all the matchlines and ends the search cycle.

6) Power Consumption: The dynamic power consumed by a single matchline that misses is due to the rising edge during precharge and the falling edge during evaluation, and is given by the equation

$$P_{\rm miss} = C_{\rm ML}V_{\rm DD}^2 f \tag{5}$$

where f is the frequency of search operations. In the case of a match, the power consumption associated with a single matchline depends on the previous state of the matchline; however, since typically there is only a small number of matches we can neglect this power consumption. Accordingly, the overall matchline power consumption of a CAM block with w matchlines is

$$P_{\rm ML} = wP_{\rm miss} \tag{6}$$

$$= wC_{\rm ML}V_{\rm DD}^2 f. \tag{7}$$

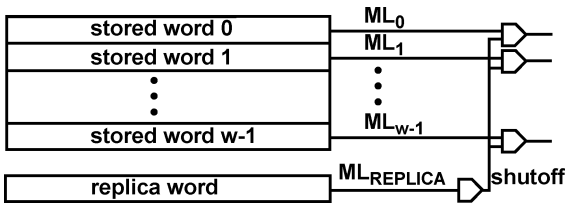

Fig. 13. Replica matchline for timing control.

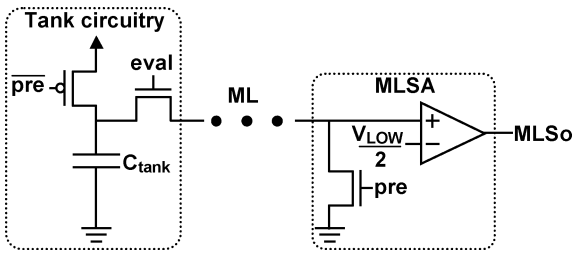

Fig. 14. Low-swing matchline sensing scheme of [21].

B. Low-Swing Schemes

One method of reducing the ML power consumption, and potentially increasing its speed, is to reduce the ML voltage swing [21], [41]. The reduction of power consumption is linearly proportional to the reduction of the voltage swing, resulting in the modified power equation

$$P_{\mathrm{ML}} = w \times C_{\mathrm{ML}} V_{\mathrm{DD}} V_{\mathrm{MLswing}} f \tag{8}$$

where V_{MLswing} is the voltage swing of the ML. The main challenge addressed by various low-swing implementations is using a low-swing voltage without resorting to an externally generated reference voltage.

Fig. 14 is a simplified schematic of the matchline sensing scheme of [21], which saves power by reducing swing. The matchline swing is reduced from the full supply swing to $V_{\mathrm{LOW}} = 300$ mV. Precharge to 300 mV is accomplished by associating a tank capacitor, C_{tank}, with every matchline. The precharge operation (assertion of pre signal) charges the tank capacitor to V_{DD} and then uses charge sharing (enabled by eval signal) to dump the charge onto the matchline. The matchline precharge voltage is given by

$$V_{\mathrm{MLpre}} = V_{\mathrm{DD}} \frac{C_{\mathrm{tank}}}{C_{\mathrm{ML}} + C_{\mathrm{tank}}} \tag{9}$$

and is set to be equal to 300 mV in the design [21]. In the case of a miss, the matchline discharges through the CAM cell(s) to ground, whereas in the case of match, the matchline remains at the precharge level. Matchline evaluation uses a sense amplifier that employs transistor ratios to generate a reference level of about $V_{\mathrm{LOW}}/2 = 150$ mV. This sense amplifier is shown in generic form in the figure. A similar charge-sharing matchline scheme was also described in [41].

C. Current-Race Scheme

Fig. 15(a) shows a simplified schematic of the current-race scheme [42]. This scheme precharges the matchline low and evaluates the matchline state by charging the matchline with a current I_{ML} supplied by a current source. The signal timing is shown in Fig. 15(b). The precharge signal, mlpre, starts the search cycle by precharging the matchline low. Since the matchline is precharged low, the scheme concurrently charges the searchlines to their search data values, eliminating the need for a separate SL precharge phase required by the precharge-high scheme of Fig. 10(b). Instead, there is a single SL/ML precharge phase, as indicated in Fig. 15(b). After the SL/ML precharge phase completes, the enable signal, $\overline{\mathrm{en}}$, connects the current source to the matchline. A matchline in the match state charges linearly to a high voltage, while a matchline in the miss state charges to to a voltage of only $I_{\mathrm{ML}} \times R_{\mathrm{ML}}/m$, where m denotes the number of misses in cells connected to the matchline. By setting the maximum voltage of a miss to be small, a simple matchline sense amplifier easily differentiates between a match state and a miss state and generates the signal MLso. As shown in Fig. 15, the amplifier is the nMOS transistor, $\mathrm{M_{sense}}$, whose output is stored by a half-latch. The nMOS sense transistor trips the latch with a threshold of V_{tn}. After some delay, matchlines in the match state will charge to slightly above V_{tn} tripping their latch, whereas matchlines in the miss state will remain at a much smaller voltage, leaving their latch in the initial state. A simple replica matchline (not shown) controls the shutoff of the current source and the latching of the match signal.

We derive the power consumption of this scheme by first noting that the same amount of current is discharged into every matchline, regardless of the state of the matchline. Looking at the match case for convenience, the power consumed to charge a matchline to slightly above V_{tn} is

$$P_{\mathrm{match}} = C_{\mathrm{ML}} V_{\mathrm{DD}} V_{tn} f. \tag{10}$$

Since the power consumption of a match and a miss are identical, the overall power consumption for all w matchlines is

$$P_{\mathrm{ML}} = w \times C_{\mathrm{ML}} V_{\mathrm{DD}} V_{tn} f. \tag{11}$$

This equation is identical to the low-swing scheme (8), with $V_{\mathrm{MLswing}} = V_{tn}$. The benefits of this scheme over the precharge-high schemes are the simplicity of the threshold circuitry and the extra savings in searchline power due to the elimination of the SL precharge phase which is discussed further in Section IV.

The current-race scheme also allows changing the CAM cell configuration due to the fact that the matchline is precharged low. With precharge low, there is no charge-sharing problem for either CAM cell configuration of Fig. 12, since the ML precharge level is the same as the level of the intermediate nodes $\mathrm{X_1}$ and $\mathrm{X_2}$. Rather than to avoid charge sharing, the criterion that determines which cell to use in this case is matching parasitic capacitances between MLs. In the configuration of Fig. 12(b), the parasitic load on a matchline depends on the ON/OFF state of $\mathrm{M_1}$ and of $\mathrm{M_2}$. Since different cells will have different stored data, there will be variations in the capacitance C_{ML} among the MLs. However, in the configuration of Fig. 12(a), the variation of parasitic capacitance on the matchline depends only on the states of SL and $\overline{\mathrm{SL}}$ which are the same for all cells in the same column. Thus, the configuration of Fig. 12(a) maintains good matching between MLs and prevents possible sensing errors due to parasitic capacitance variations.

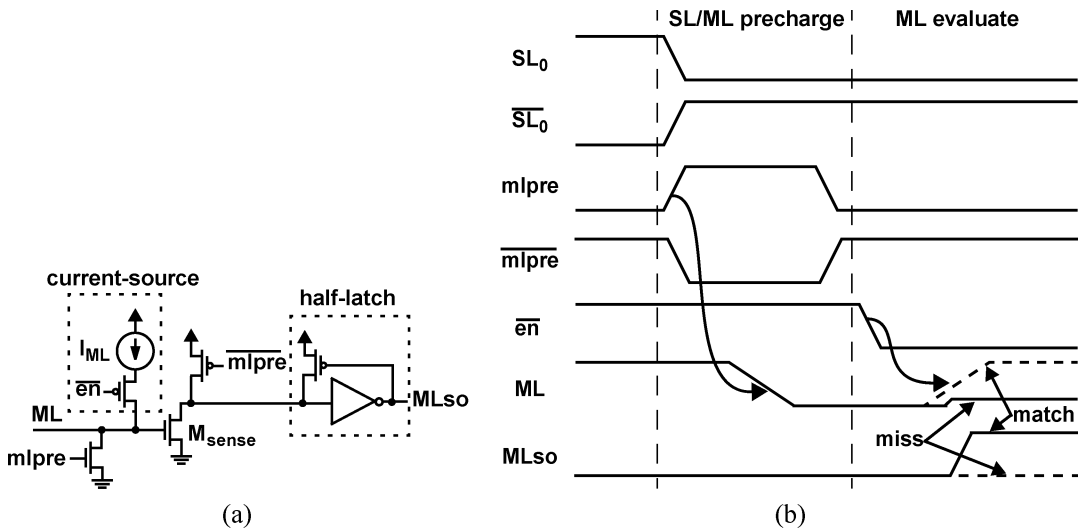

Fig. 15. For current-race matchline sensing [42]: (a) a schematic of the circuit implementation including precharge circuitry and (b) a timing diagram for a single search cycle.

D. Selective-Precharge Scheme

The matchline-sensing techniques we have seen so far, expend approximately the same amount of energy on every matchline, regardless of the specific data pattern, and whether there is a match or a miss. We now examine three schemes that allocate power to matchlines nonuniformly.

The first technique, called selective precharge, performs a match operation on the first few bits of a word before activating the search of the remaining bits [43]. For example, in a 144-bit word, selective precharge initially searches only the first 3 bits and then searches the remaining 141 bits only for words that matched in the first 3 bits. Assuming a uniform random data distribution, the initial 3-bit search should allow only $1/2^3$ words to survive to the second stage saving about 88% of the matchline power. In practice, there are two sources of overhead that limit the power saving. First, to maintain speed, the initial match implementation may draw a higher power per bit than the search operation on the remaining bits. Second, an application may have a data distribution that is not uniform, and, in the worst-case scenario, the initial match bits are identical among all words in the CAM, eliminating any power saving.

Fig. 16 is a simplified schematic of an example of selective precharge similar to that presented in the original paper [43]. The example uses the first bit for the initial search and the remaining $(n-1)$ bits for the remaining search. To maintain speed, the implementation modifies the precharge part of the precharge-high scheme [of Fig. 10(a) and (b)]. The ML is precharged through the transistor M_1, which is controlled by the NAND CAM cell and turned on only if there is a match in the first CAM bit. The remaining cells are NOR cells. Note that the ML of the NOR cells must be pre-discharged (circuitry not shown) to ground to maintain correct operation in the case that the previous search left the matchline high due to a match. Thus, one implementation of selective precharge is to use this mixed NAND/NOR matchline structure.

Selective precharge is perhaps the most common method used to save power on matchlines [22], [44]–[48] since it is both simple to implement and can reduce power by a large amount in many CAM applications.

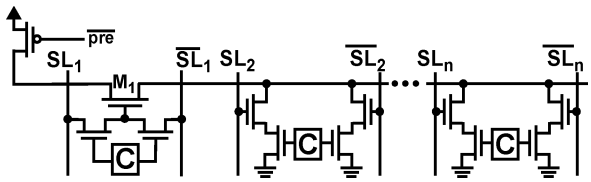

Fig. 16. Sample implementation of the selective-precharge matchline technique [43]. The first cell on the matchline is a NAND cell, while the other cells are NOR cells. Precharge occurs only in the case where there is a match in the first cell. If there is no match in the first cell, the precharge transistor is disconnected from the matchline, thus saving power.

E. Pipelining Scheme

In selective precharge, the matchline is divided into two segments. More generally, an implementation may divide the matchline into any number of segments, where a match in a given segment results in a search operation in the next segment but a miss terminates the match operation for that word. A design that uses multiple matchline segments in a pipelined fashion is the pipelined matchlines scheme [49], [50]. Fig. 17(a) shows a simplified schematic of a conventional NOR matchline structure where all cells are connected in parallel. Fig. 17(b) shows the same set of cells as in Fig. 17(a), but with the matchline broken into four matchline segments that are serially evaluated. If any stage misses, the subsequent stages are shut off, resulting in power saving. The drawbacks of this scheme are the increased latency and the area overhead due to the pipeline stages. By itself, a pipelined matchline scheme is not as compelling as basic selective precharge; however, pipelining enables the use of hierarchical searchlines, thus saving power. Section IV discusses hierarchical searchlines in detail.

Another approach is to segment the matchline so that each individual bit forms a segment [51]. Thus, selective precharge operates on a bit-by-bit basis. In this design, the CAM cell is modified so that the match evaluation ripples through each CAM cell. If at any cell there is a miss, the subsequent cells do not activate, as there is no need for a comparison operation. The drawback of this scheme is the extra circuitry required at each cell to gate the comparison with the result from the previous cell.

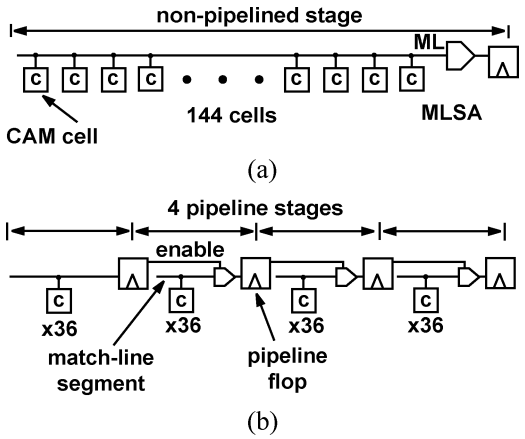

Fig. 17. Pipelined matchlines reduce power by shutting down after a miss in a stage.

F. Current-Saving Scheme

The current-saving scheme [52], [53] is another data-dependent matchline-sensing scheme which is a modified form of the current-race sensing scheme. Recall that the current-race scheme uses the same current on each matchline, regardless of whether it has a match or a miss. The key improvement of the current-saving scheme is to allocate a different amount of current for a match than for a miss. In the current-saving scheme, matches are allocated a larger current and misses are allocated a lower current. Since almost every matchline has a miss, overall the scheme saves power.

Fig. 18 shows a simplified schematic of the current-saving scheme. The main difference from the current-race scheme as depicted in Fig. 15 is the addition of the current-control block. This block is the mechanism by which a different amount of current is allocated, based on a match or a miss. The input to this current-control block is the matchline voltage, V_{ML}, and the output is a control voltage that determines the current, I_{ML}, which charges the matchline. The current-control block provides positive feedback since higher V_{ML} results in higher I_{ML}, which, in turn, results in higher V_{ML}.

Recall that matchlines in the match state are purely capacitive [see Fig. 11(a)], whereas matchlines in the miss state have both capacitance and resistance [see Fig. 11(b)]. In this scheme, the matchline is precharged low, just as in the current-race scheme. In the current-race scheme, the current source provides a constant amount of current regardless of the state of the matchline. In the current-saving scheme, the amount of current is initially the same for all matchlines, but the current control reduces the current provided to matchlines that miss (have a resistance to ground), but maintains the current to matchlines that match (with no resistance to ground). The current-control block increases the current as the voltage on the ML rises and the voltage on the ML rises faster for large ML resistance. Recall that the ML resistance depends on the number of bits that miss. Since the amount of current decreases with the number of misses, it follows that the power dissipated on the matchline also depends on the number of bits that miss. References [52] and [53] show that this scheme saves over 50% of matchline power compared to the current-race scheme.

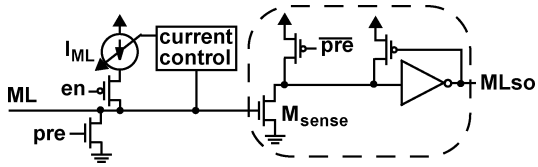

Fig. 18. Current-saving matchline-sensing scheme [52], [53].

G. Summary

Table IV summarizes the main features of the matchline sensing-schemes discussed in this section. The results in the table were simulated using HSpice for a 0.18-μm CMOS process using the baseline ternary cell of [54] in a 152 $\times$ 72 ternary CAM block. The energy is reported in the standard units of fJ/bit/search. The table includes qualitative evaluations of the simplicity of implementing the sensing scheme and the scheme's immunity to noise. Overall, we see that there is only a small difference in speed between the various schemes, so the parameters that differentiate the schemes are primarily power consumption, scheme simplicity, and noise immunity.

IV. Searchline Driving Approaches

Searchline power consumption depends partly on the matchline scheme. In this section, we look at the power consumption of the searchlines for three cases: matchline precharge high, matchline precharge low, and pipelined matchlines with hierarchical searchlines.

A. Conventional Approach

The conventional approach to driving the searchlines applies to matchline schemes that precharge the matchlines high. In this approach, during the search cycle, the searchlines are driven by a cascade of inverters first to their precharge level and then to their data value. The searchline power consumption depends on the searchline capacitance, which consists of wire capacitance and one transistor gate capacitance per row. The equation for the dynamic power consumption of the searchlines is

$$P_{\mathrm{SL}} = 2n \times \frac{1}{2} C_{\mathrm{SL}} V_{\mathrm{DD}}^2 f \tag{12}$$

$$= n C_{\mathrm{SL}} V_{\mathrm{DD}}^2 f \tag{13}$$

where C_{SL} is the total capacitance of a single searchline, n is the total number of searchline pairs (and $2n$ is the total number of searchlines), and V_{DD} is the power supply voltage. Recall that there are two searchlines per bit, which are precharged low and then charged to the appropriate (differential) search-data values. This results in two transitions per searchlines pair, or, equivalently, one transition per searchline. To this power, we must add the power consumption of the drivers. To maintain high speed, we drive the capacitance with drivers consisting of a cascade of inverters sized using exponentially increasing widths [55]. When the searchline drivers are sized to minimize delay, the drivers add an overhead of about 25% to (12).

B. Eliminating Searchline Precharge

We can save searchline power by eliminating the SL precharge phase depicted in Fig. 10(b). Eliminating the SL precharge phase reduces the toggling of the searchlines, thus reducing

TABLE IV
SUMMARY OF MATCHLINE SENSING TECHNIQUES

Scheme	ML Energy (fJ/bit/search)	Cycle Time (ns)	Scheme Simplicity	Noise Immunity	Reference(s)
Conventional	9.5	3.9	++	+	[25]
Low-swing[1]	4.2	3.1	−	−	[21], [34], [41]
Current Race	5.5	3.7	+	−	[42]
Selective Precharge[2]	5.6	3.5	+	+	[43]
Pipelining[3]	5.8	3.8	−	+	[49], [50]
Current Saving	4.3	3.7	−−	−	[52], [53]

[1] ML swing of 0.45 V. [2] Single bit used for first segment.

[3] ML divided into three ML segments.

power. As discussed in the previous section, matchline-sensing schemes that precharge the matchline low eliminate the need for SL precharge, since enabling the pulldown path in the NOR cell does not interfere with matchline precharge. These schemes directly activate the searchlines with their search data without going through an SL precharge phase. Since, in the typical case, about 50% of the search data bits toggle from cycle to cycle, there is a 50% reduction in searchline power, compared to the precharge-high matchline-sensing schemes that have an SL precharge phase. The equation for the reduced power in this case is

$$P_{\mathrm{SL}} = \frac{1}{2} n C_{\mathrm{SL}} V_{\mathrm{DD}}^2 f. \tag{14}$$

This equation shows that matchline-sensing schemes that precharge the matchlines low also save power on the searchlines. In fact, in these precharge-low schemes, the reduction in searchline power can be as large as, or even larger than, the reduction in matchline power.

C. Hierarchical Searchlines

Another method of saving searchline power is to shut off some searchlines (when possible) by using the hierarchical searchline scheme [30], [49], [50], [56]. Hierarchical searchlines are built on top of pipelined matchlines, which were discussed in Section III. The basic idea of hierarchical searchlines is to exploit the fact that few matchlines survive the first segment of the pipelined matchlines. With the conventional searchline approach, even though only a small number of matchlines survive the first segment, all searchlines are still driven. Instead of this, the hierarchical searchline scheme divides the searchlines into a two-level hierarchy of global searchlines (GSLs) and local searchlines (LSLs). Fig. 19 shows a simplified hierarchical searchline scheme, where the matchlines are pipelined into two segments, and the searchlines are divided into four LSLs per GSL. In the figure, each LSL feeds only a single matchline (for simplicity), but the number of matchlines per LSL can be 64 to 256. The GSLs are active every cycle, but the LSLs are active only when necessary. Activating LSLs is necessary when at least one of the matchlines fed by the LSL is active. In many cases, an LSL will have no active matchlines in a given cycle, hence there is no need to activate

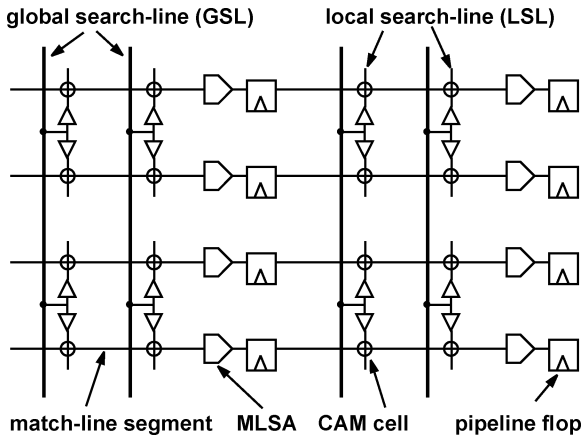

Fig. 19. Hierarchical searchline structure [49], [50].

the LSL, saving power. Thus, the overall power consumption on the searchlines is

$$P_{\mathrm{SL}} = \left(\underbrace{C_{\mathrm{GSL}} V_{\mathrm{DD}}^2}_{\mathrm{GSL}} + \underbrace{\alpha C_{\mathrm{LSL}} V_{\mathrm{DD}}^2}_{\mathrm{LSL}} \right) f \tag{15}$$

where C_{GSL} is the GSL capacitance, C_{LSL} is the LSL capacitance (of all LSLs connected to a GSL) and α is the activity rate of the LSLs. C_{GSL} primarily consists of wiring capacitance, whereas C_{LSL} consists of wiring capacitance and the gate capacitance of the SL inputs of the CAM cells. The factor α, which can be as low as 25% in some cases, is determined by the search data and the data stored in the CAM. We see from (15) that α determines how much power is saved on the LSLs, but the cost of this savings is the power dissipated by the GSLs. Thus, the power dissipated by the GSLs must be sufficiently small so that overall searchline power is lower than that using the conventional approach.

If wiring capacitance is small compared to the parasitic transistor capacitance [56], then the scheme saves power. However, as transistor dimensions scale down, it is expected that wiring capacitance will increase relative to transistor parasitic capacitance. In the situation where wiring capacitance is comparable or larger than the parasitic transistor capacitance, C_{GSL} and C_{LSL} will be similar in size, resulting in no power savings. In this case, small-swing signaling on the GSLs can reduce the

TABLE V
SUMMARY OF SEARCHLINE-DRIVING APPROACHES

Scheme	SL Energy (fJ/bit/search)	Noise Simplicity	Noise Immunity	Reference(s)
Conventional	4.6	++	+	[25]
Eliminating Precharge	2.3	++	+	[21], [42], [52], [53]
Hierarchical Searchlines[1]	1.4	−−	−	[49], [50]

[1] Simulated for a 1024×144 CAM.

power of the GSLs compared to that of the full-swing LSLs [49], [50]. This results in the modified searchline power of

$$P_{\mathrm{SL}} = 2n\left(C_{\mathrm{GSL}}V_{\mathrm{LOW}}^2 + \alpha C_{\mathrm{LSL}}V_{\mathrm{DD}}^2\right) f \qquad (16)$$

where V_{LOW} is the low-swing voltage on the GSLs (assuming an externally available power supply V_{LOW}). This scheme requires an amplifier to convert the low-swing GSL signal to the full-swing signals on the LSLs. Fortunately, there is only a small number of these amplifiers per searchline, so that the area and power overhead of this extra circuitry is small.

D. Summary

Table V summarizes the main features of the searchline-driving approaches discussed in this section. The power and speed results presented in the table were found by simulation with HSpice for a 0.18-μm CMOS process using the baseline ternary cell of [54] in a 512 × 72 ternary CAM block. The power for hierarchical searchlines is reported for the larger CAM size of 1024 × 144 as used in [49] and [50] since the scheme requires a longer word to be amortize the cost of the overhead power. The table includes qualitative evaluations of the simplicity of implementing the sensing scheme and the scheme's immunity to noise. Overall, we see that the conventional and eliminating precharge scheme are the same in all respects except the power consumption is halved in the eliminating precharge scheme, make it preferable to the conventional approach when a compatible matchline sensing scheme is available. Furthermore, the hierarchical searchlines technique saves even more power, at the cost of both increased implementation complexity and lower noise immunity.

V. POWER-SAVING CAM ARCHITECTURES

Until this point, we have seen circuit techniques to reduce power. In this section, we look at architectural techniques for saving power. In 1995, Schultz and Gulak [57] presented a survey of architectures for large-capacity CAMs. In this paper, we review three architectures that have been reported since that time.

First, a straightforward architectural technique that saves power is bank-selection, where only a subset of the CAM is active on any given cycle. The bank selection scheme was first proposed in [58] and [59] and labeled preclassification in [60] and [61]. The purpose of these older bank-selection schemes was to save area but the technique has since been modified instead to save power. The basic concept is that bank selection divides the CAM into subsets called banks. Fig. 20 provides a block diagram of a simplified bank-selection scheme. Two extra data bits, called bank-select bits, partition the CAM into four blocks. When storing data, the bank-select bits determine into which of the four blocks to store the data. When searching data, the bank-select bits determine which one of the four blocks to activate and search. The decoder accomplishes the selection by providing enable signals to each block. In the example in the figure, the bank-select bits are 10 which selects bank 2.

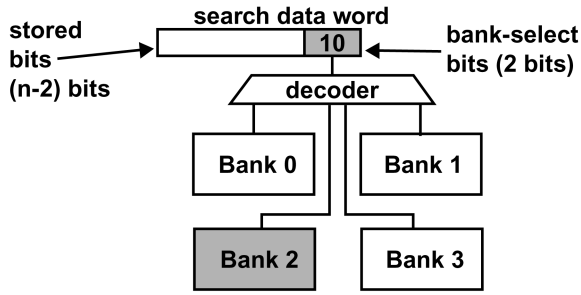

Fig. 20. Simplified diagram of a bank-selection scheme. The bank-selection scheme partitions the CAM and shuts off unneeded banks.

In the original preclassification schemes, this architecture was used to reduce area by sharing the comparison circuitry between blocks. Although the blocks in Fig. 20 are shown as physically separate, they can be arranged such that words from different blocks are adjacent. Since only one of four blocks is active at any time, only 1/4 of the comparison circuitry is necessary compared to the case with no bank selection, thus saving area.

Instead of saving area, recent bank selection schemes aim to reduce power [21]. Bank selection reduces overall power consumption in proportion to the number of blocks. Thus, using four blocks ideally reduces the power consumption by 75% compared to a CAM without bank selection.

The major drawback of bank selection is the problem of bank overflow. Since, in a CAM, there are many more input combinations than storage locations, the storage of a bank can quickly overflow. Take, for example, a CAM with 72-bit words (and an additional bank-select bit) and 32K entries divided into two banks with 16K entries. While each bank has 16K locations, there are actually 2^{72} possible entries per bank. Thus, it can often occur that there are more entries than can fit in the assigned bank. This overflow condition requires extra circuitry and forces multiple banks to be activated at once, decreasing the savings in power. To avoid overflow, an external mechanism can balance the data in the banks by periodically re-partitioning the banks. Algorithms for partitioning the input data space in packet-forwarding applications are an active area of research [62], [63].

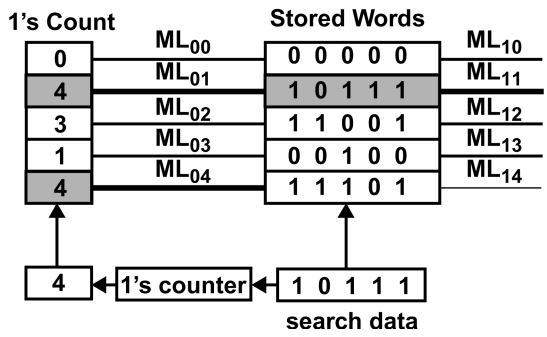

Fig. 21. Conceptual view of pre-computation-based CAM.

The second architectural technique for saving power, which applies only to binary CAM, is pre-computation. Pre-computation stores some extra information along with each word that is used in the search operation to save power [64], [65]. These extra bits are derived from the stored word, and used in an initial search before searching the main word. If this initial search fails, then the CAM aborts the subsequent search, thus saving power. This is the same mechanism that we have already seen used in the selective-precharge technique; however, pre-computation has a second purpose, which is to save CAM cell area, a fact which we will see shortly. Fig. 21 shows a conceptual view of the operation of a pre-computation-based CAM (PB-CAM). The extra information holds the number of ones in the stored word. For example, in Fig. 21, when searching for the data word, 10111, the pre-computation circuit counts the number of ones (which is four in this case). The number four is compared on the left-hand side to the stored 1's counts. Only matchlines ML_{01} and ML_{04} match, since only they have a 1's count of four. In the stored-words stage in Fig. 21, only two comparisons actively consume power and only matchline ML_{11} results in a match.

The compelling part of PB-CAM results from exploiting the 1's count to simplify the comparison circuitry in the second stage. Although the 1's count comparison circuitry uses conventional CAM cells, the regular storage uses simplified CAM cells that have only one pulldown path (a total of two pulldown transistors instead of four), as shown in Fig. 22. Even with the simplified cell, the match result is always correct since a mismatched word in this second stage always has the same number of 1 s as the search data word. A mismatch will always cause at least one path to ground. For the match case, there are no paths to ground, guaranteeing correct operation.

The third architectural concept for saving power is to change the encoding of stored data in the CAM cell [66], [67]. First, notice that, by storing the entry 101XX in a 5-bit ternary CAM, the stored word will match the four search data words 10100, 10101, 10110, and 10111. Correspondingly, we can view a stored word in a ternary CAM as actually storing multiple words. We can extend this view to multiple entries. Thus, for example, the three entries 00011, 001XX, 01000 will match any search word in the range 00011 to 01000. Efficiently mapping a set of ranges to stored ternary CAM words allows for reduction of the size of the CAM required for an application [68]–[70]. Reducing the number of entries also has the effect of reducing power consumption, since power consumption in CAM is proportional to the array size.

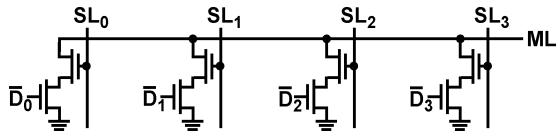

Fig. 22. Schematic view of matchline structure of pre-computation-based CAM.

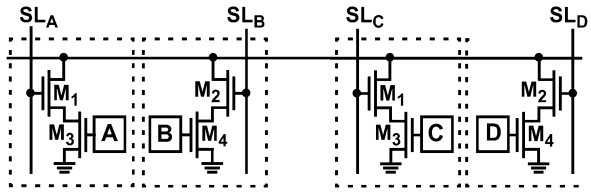

Fig. 23. Two ternary CAM cells are viewed as four independent half cells in the dense encoding scheme of [66] and [67].

To allow for a more dense encoding than conventional ternary CAM, Hanzawa *et al.* [66], [67] propose changing the encoding of the CAM resulting in a smaller average number of entries. For the application of network routing, their scheme in comparison to conventional ternary encoding reduces the number of entries by almost 50% in some cases.

We describe this new dense encoding by comparing it to conventional CAM encoding. Fig. 23 shows two ternary CAM cells on a matchline, with each half ternary cell identified by a dotted box. A group of two ternary cells is called a local matchline in this scheme. In conventional ternary CAM encoding, there are three possible states (0, 1, X) for each ternary CAM cell and, correspondingly, nine states in total for two ternary cells. Table VI lists all the possible encodings for a pair of ternary CAM cells. The first column of Table VI lists these nine states and the second column lists the value of the four storage cells (two per ternary cells) denoted A, B, C, and D with respect to Fig. 23. The third column lists the words that cells represent. For the last four rows, the X entries result in multiple stored words per cell. Note that only some of the possible combinations are available. For example, the combinations of 0, 2 and 1, 3 are available, but the combinations of 0, 3 and 0, 2, 3 are not available.

Since there are four storage cells for two ternary cells, there are actually $2^4 = 16$ possible states. The dense encoding scheme makes use of these previously unused states to make possible the storage of any combination. Table VII lists the encoding for four ternary half-CAM cells (forming two ternary CAM cells) for the dense encoding scheme. The first column of Table VII lists the possible entry data for the dense-encoding and the second column lists the values of the storage cells A, B, C, and D, corresponding to those in Fig. 23. In this scheme, every possible combination of 0, 1, 2, and 3 can be stored.

The matchline architecture is modified to accommodate the dense encoding as shown in Fig. 24. The matchline is divided into local matchlines (LMLs) and global matchlines (GMLs). Only two LMLs for one GML are shown in the figure for simplicity. The GMLs are divided into sets of LMLs that are made up of four ternary half-cells (Fig. 23). A GML operates the same way as the conventional NOR matchline, where a path to ground indicates a miss, and no path to ground indicates a match. The logic levels on the LML, however, are inverted so that a path to

TABLE VI
ENCODINGS FOR TWO TERNARY CELLS

Ternary Word	Stored Values (AB CD)	Entry Data
00	01 01	0
01	01 10	1
10	10 01	2
11	10 10	3
0X	01 00	0, 1
1X	10 00	2, 3
X0	00 01	0, 2
X1	00 10	1, 3
XX	00 00	0, 1, 2, 3

ground indicates a match, while no path to ground indicates a miss. To see how this works, we assume that an LML stores the numbers 0, 2, and 3, which according to Table VII, means that $\mathrm{ABCD} = 1101$. In the case where the search word is 0, 2, or 3 (corresponding to searchlines set to 0001, 0100, and 1000, respectively), there is a path to ground indicating a match. In the case of search for a 1 (corresponding to searchlines set to 0010) there is no path to ground indicating a miss. The GMLs are used to construct longer words that are made up of multiple LMLs.

VI. DISCUSSION AND FUTURE TRENDS

The main thread of recent CAM research is the reduction of power consumption. Power dissipation in CAM is dominated by dynamic power as summarized by the relation

$$P \propto aCV_{\mathrm{swing}}V_{\mathrm{DD}} \tag{17}$$

where α represents the switching activity. Based on this relation, there is only a small number of parameters that can be varied to reduce power. First, wherever possible, we organize the CAM so as to minimize transistor and wire capacitance to keep C small. Second, as the matchline-sensing schemes show, we reduce V_{swing} on the matchlines to save power. Third, we reduce the transition activity factor, α, by reducing the transition rate as the in selective-precharge, hierarchical searchlines, and bank-selection schemes. Finally, we also improve power savings with reductions of the supply voltage, V_{DD}, dictated by the CMOS process used.

One critical area of CAM performance that is largely unexplored is that of peak power consumption with worst-case data patterns. Many of the techniques surveyed in this paper are effective in reducing average power consumption but actually increase peak power consumption. Peak power consumption is a concern because many design decisions, especially at the board level, are determined by peak power considerations. For example, a system with 6 CAM chips that each consume 5 W of peak power must dedicate 30 W of the supply's power to the CAM chips. With aggressive power-saving techniques, the chips may each draw as little as 1 W average power, leaving unused 24 W of the potential power. This extra power capability could have been used for other components, or a cheaper supply with lower output capability could have been used.

TABLE VII
POSSIBLE ENCODINGS FOR TWO TERNARY CELLS IN DENSE ENCODING

Entry Data	Stored Values (ABCD)
–	0000
0	0001
1	0010
2	0100
3	1000
0, 1	0011
2, 3	1100
0, 2	0101
1, 3	1010
1, 2	0110
0, 3	1001
0, 1, 2	0111
1, 2, 3	1110
0, 2, 3	1101
0, 1, 3	1011
0, 1, 2, 3	1111

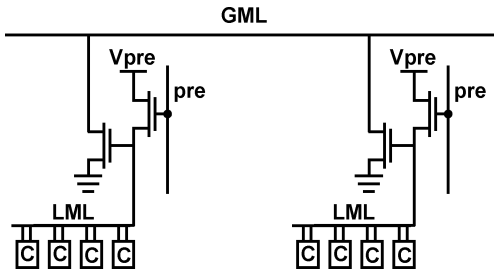

Fig. 24. Each local matchline (LML) is encoded as in Table VII using active low logic. The logic level on the global matchline operates with the typical matchline logic levels with a logic high indicating a match and a logic low indication a miss [66], [67].

We also expect the promise of emerging nonvolatile memories to spur innovation in CAM design. These memories, such as ferroelectric RAM (FeRAM), magnetoresistive RAM (MRAM), and phase-change RAM (PRAM) [71]–[73], present several new challenges. The primary one comes from the fact that unlike SRAM cells (or DRAM cells), these memories do not simply store data in the form of low and high voltage levels. Instead, for example, MRAM and PRAM store data as low and high resistance values This means that the comparison operation and the matchline-sensing scheme in nonvolatile CAM will likely be implemented in a different fashion from that now used.

VII. CONCLUSION

In this paper, we have surveyed CAM circuits and architectures, with an emphasis on high-capacity CAM. First, we motivated our discussion by showing how CAMs can be applied to packet forwarding in network routers. At the circuit level, we have reviewed the two basic CMOS cells, namely the NOR cell and the NAND cell. We have also shown how the cells are combined in a matchline structure to form a CAM word. We have

explored the conventional precharge-high scheme for sensing the matchline, as well as several variations that save matchline power including low-swing sensing, the current-race scheme, selective precharge, pipelining, and current saving scheme. We have also reviewed the conventional approach for driving searchlines, and the power-saving approaches which eliminate the searchline precharge or employ hierarchical searchlines. At the architectural level, we have reviewed three architectures for reducing CAM power, namely bank-selection, pre-computation, and dense encoding. Finally, we have presented our views of future development in the CAM field.

ACKNOWLEDGMENT

The authors thank for their insightful feedback on drafts of this paper O. Tyshchenko, Prof. K. C. Smith, and N. Azizi from the University of Toronto, Toronto, Canada; K. J. Schultz from Chariot Venture Acceleration, Ottawa, Canada; I. Arsovski from IBM Microelectronics, Essex Junction, VT, USA; and H. Tamura from Fujitsu, Kawasaki, Japan. The authors also thank the anonymous reviewers for their valuable comments and suggestions.

REFERENCES

[1] T. Kohonen, *Content-Addressable Memories*, 2nd ed. New York: Springer-Verlag, 1987.
[2] L. Chisvin and R. J. Duckworth, "Content-addressable and associative memory: alternatives to the ubiquitous RAM," *IEEE Computer*, vol. 22, no. 7, pp. 51–64, Jul. 1989.
[3] K. E. Grosspietsch, "Associative processors and memories: a survey," *IEEE Micro*, vol. 12, no. 3, pp. 12–19, Jun. 1992.
[4] I. N. Robinson, "Pattern-addressable memory," *IEEE Micro*, vol. 12, no. 3, pp. 20–30, Jun. 1992.
[5] S. Stas, "Associative processing with CAMs," in *Northcon/93 Conf. Record*, 1993, pp. 161–167.
[6] M. Meribout, T. Ogura, and M. Nakanishi, "On using the CAM concept for parametric curve extraction," *IEEE Trans. Image Process.*, vol. 9, no. 12, pp. 2126–2130, Dec. 2000.
[7] M. Nakanishi and T. Ogura, "Real-time CAM-based Hough transform and its performance evaluation," *Machine Vision Appl.*, vol. 12, no. 2, pp. 59–68, Aug. 2000.
[8] E. Komoto, T. Homma, and T. Nakamura, "A high-speed and compact-size JPEG Huffman decoder using CAM," in *Symp. VLSI Circuits Dig. Tech. Papers*, 1993, pp. 37–38.
[9] L.-Y. Liu, J.-F. Wang, R.-J. Wang, and J.-Y. Lee, "CAM-based VLSI architectures for dynamic Huffman coding," *IEEE Trans. Consumer Electron.*, vol. 40, no. 3, pp. 282–289, Aug. 1994.
[10] B. W. Wei, R. Tarver, J.-S. Kim, and K. Ng, "A single chip Lempel-Ziv data compressor," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, vol. 3, 1993, pp. 1953–1955.
[11] R.-Y. Yang and C.-Y. Lee, "High-throughput data compressor designs using content addressable memory," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, vol. 4, 1994, pp. 147–150.
[12] C.-Y. Lee and R.-Y. Yang, "High-throughput data compressor designs using content addressable memory," *IEE Proc.—Circuits, Devices and Syst.*, vol. 142, no. 1, pp. 69–73, Feb. 1995.
[13] D. J. Craft, "A fast hardware data compression algorithm and some algorithmic extansions," *IBM J. Res. Devel.*, vol. 42, no. 6, pp. 733–745, Nov. 1998.
[14] S. Panchanathan and M. Goldberg, "A content-addressable memory architecture for image coding using vector quantization," *IEEE Trans. Signal Process.*, vol. 39, no. 9, pp. 2066–2078, Sep. 1991.
[15] T.-B. Pei and C. Zukowski, "VLSI implementation of routing tables: tries and CAMs," in *Proc. IEEE INFOCOM*, vol. 2, 1991, pp. 515–524.
[16] ——, "Putting routing tables in silicon," *IEEE Network Mag.*, vol. 6, no. 1, pp. 42–50, Jan. 1992.
[17] A. J. McAuley and P. Francis, "Fast routing table lookup using CAMs," in *Proc. IEEE INFOCOM*, vol. 3, 1993, pp. 1282–1391.
[18] N.-F. Huang, W.-E. Chen, J.-Y. Luo, and J.-M. Chen, "Design of multi-field IPv6 packet classifiers using ternary CAMs," in *Proc. IEEE GLOBECOM*, vol. 3, 2001, pp. 1877–1881.
[19] G. Qin, S. Ata, I. Oka, and C. Fujiwara, "Effective bit selection methods for improving performance of packet classifications on IP routers," in *Proc. IEEE GLOBECOM*, vol. 2, 2002, pp. 2350–2354.
[20] H. J. Chao, "Next generation routers," *Proc. IEEE*, vol. 90, no. 9, pp. 1518–1558, Sep. 2002.
[21] G. Kasai, Y. Takarabe, K. Furumi, and M. Yoneda, "200 MHz/200 MSPS 3.2 W at 1.5 V Vdd, 9.4 Mbits ternary CAM with new charge injection match detect circuits and bank selection scheme," in *Proc. IEEE Custom Integrated Circuits Conf. (CICC)*, 2003, pp. 387–390.
[22] A. Roth, D. Foss, R. McKenzie, and D. Perry, "Advanced ternary CAM circuits on 0.13 μm logic process technology," in *Proc. IEEE Custom Integrated Circuits Conf. (CICC)*, 2004, pp. 465–468.
[23] T. Yamagato, M. Mihara, T. Hamamoto, Y. Murai, T. Kobayashi, M. Yamada, and H. Ozaki, "A 288-kbit fully parallel content addressable memory using a stacked-capacitor cell structure," *IEEE J. Solid-State Circuits*, vol. 27, no. 12, pp. 1927–1933, Dec. 1992.
[24] T. Ogura, M. Nakanishi, T. Baba, Y. Nakabayashi, and R. Kasai, "A 336-kbit content addressable memory for highly parallel image processing," in *Proc. IEEE Custom Integrated Circuits Conf. (CICC)*, 1996, pp. 273–276.
[25] H. Kadota, J. Miyake, Y. Nishimichi, H. Kudoh, and K. Kagawa, "An 8-kbit content-addressable and reentrant memory," *IEEE J. Solid-State Circuits*, vol. SC-20, no. 5, pp. 951–957, Oct. 1985.
[26] K. J. Schultz, F. Shafai, G. F. R. Gibson, A. G. Bluschke, and D. E. Somppi, "Fully parallel 25 MHz, 2.5-Mb CAM," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, 1998, pp. 332–333.
[27] V. Lines, A. Ahmed, P. Ma, and S. Ma, "66 MHz 2.3 M ternary dynamic content addressable memory," in *Record IEEE Int. Workshop on Memory Technology, Design and Testing*, 2000, pp. 101–105.
[28] R. Panigrahy and S. Sharma, "Sorting and searching using ternary CAMs," in *Proc. Symp. High Performance Interconnects*, 2002, pp. 101–106.
[29] ——, "Sorting and searching using ternary CAMs," *IEEE Micro*, vol. 23, no. 1, pp. 44–53, Jan.–Feb. 2003.
[30] H. Noda, K. Inoue, M. Kuroiwa, F. Igaue, K. Yamamoto, H. J. Mattausch, T. Koide, A. Amo, A. Hachisuka, S. Soeda, I. Hayashi, F. Morishita, K. Dosaka, K. Arimoto, K. Fujishima, K. Anami, and T. Yoshihara, "A cost-efficient high-performance dynamic TCAM with pipelined hierarchical search and shift redudancy architecture," *IEEE J. Solid-State Circuits*, vol. 40, no. 1, pp. 245–253, Jan. 2005.
[31] S. R. Ramírez-Chávez, "Encoding don't cares in static and dynamic content-addressable memories," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 39, no. 8, pp. 575–578, Aug. 1992.
[32] S. Choi, K. Sohn, M.-W. Lee, S. Kim, H.-M. Choi, D. Kim, U.-R. Cho, H.-G. Byun, Y.-S. Shin, and H.-J. Yoo, "A 0.7 fJ/bit/search, 2.2 ns search time hybrid type TCAM architecture," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, 2004, pp. 498–499.
[33] S. Choi, K. Sohn, and H.-J. Yoo, "A 0.7 fJ/bit/search, 2.2-ns search time hybrid-type TCAM architecture," *IEEE J. Solid-State Circuits*, vol. 40, no. 1, pp. 254–260, Jan. 2005.
[34] H. Miyatake, M. Tanaka, and Y. Mori, "A design for high-speed low-power CMOS fully parallel content-addressable memory macros," *IEEE J. Solid-State Circuits*, vol. 36, no. 6, pp. 956–968, Jun. 2001.
[35] S. Liu, F. Wu, and J. B. Kuo, "A novel low-voltage content-addressable memory (CAM) cell with a fast tag-compare capability using partially depleted (PD) SOI CMOS dynamic-threshold (DTMOS) techniques," *IEEE J. Solid-State Circuits*, vol. 36, no. 4, pp. 712–716, Apr. 2001.
[36] G. Thirugnanam, N. Vijaykrishnan, and M. J. Irwin, "A novel low power CAM design," in *Proc. 14th Annu. IEEE ASIC/SOC Conf.*, 2001, pp. 198–202.
[37] K. J. Schultz, "Content-addressable memory core cells: a survey," *Integration, VLSI J.*, vol. 23, no. 2, pp. 171–188, Nov. 1997.
[38] J.-S. Wang, H.-Y. Li, C.-C. Chen, and C. Yeh, "An AND-type matchline scheme for energy-efficient content addressable memories," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, 2005, pp. 464–465.
[39] F. Shafai, K. J. Schultz, G. F. R. Gibson, A. G. Bluschke, and D. E. Somppi, "Fully parallel 30-MHz, 2.5-Mb CAM," *IEEE J. Solid-State Circuits*, vol. 33, no. 11, pp. 1690–1696, Nov. 1998.
[40] B. S. Amrutur and M. A. Horowitz, "A replica technique for wordline and sense control in low-power SRAMs," *IEEE J. Solid-State Circuits*, vol. 33, no. 8, pp. 1208–1219, Aug. 1998.
[41] M. M. Khellah and M. Elmasry, "Use of charge sharing to reduce energy consumption in wide fan-in gates," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, vol. 2, 1998, pp. 9–12.
[42] I. Arsovski, T. Chandler, and A. Sheikholeslami, "A ternary content-addressable memory (TCAM) based on 4T static storage and including a current-race sensing scheme," *IEEE J. Solid-State Circuits*, vol. 38, no. 1, pp. 155–158, Jan. 2003.

[43] C. A. Zukowski and S.-Y. Wang, "Use of selective precharge for low-power content-addressable memories," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, vol. 3, 1997, pp. 1788–1791.
[44] I. Y.-L. Hsiao, D.-H. Wang, and C.-W. Jen, "Power modeling and low-power design of content addressable memories," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, vol. 4, 2001, pp. 926–929.
[45] A. Efthymiou and J. D. Garside, "An adaptive serial-parallel CAM architecture for low-power cache blocks," in *Proc. IEEE Int. Symp. Low Power Electronics and Design (ISLPED)*, 2002, pp. 136–141.
[46] ——, "A CAM with mixed serial-parallel comparison for use in low energy caches," *IEEE Trans. VLSI Syst.*, vol. 12, no. 3, pp. 325–329, Mar. 2004.
[47] N. Mohan and M. Sachdev, "Low power dual matchline content addressable memory," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, vol. 2, 2004, pp. 633–636.
[48] K.-H. Cheng, C.-H. Wei, and S.-Y. Jiang, "Static divided word matchline line for low-power content addressable memory design," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, vol. 2, 2004, pp. 629–632.
[49] K. Pagiamtzis and A. Sheikholeslami, "Pipelined match-lines and hierarchical search-lines for low-power content-addressable memories," in *Proc. IEEE Custom Integrated Circuits Conf. (CICC)*, 2003, pp. 383–386.
[50] ——, "A low-power content-addressable memory (CAM) using pipelined hierarchical search scheme," *IEEE J. Solid-State Circuits*, vol. 39, no. 9, pp. 1512–1519, Sep. 2004.
[51] J. M. Hyjazie and C. Wang, "An approach for improving the speed of content addressable memories," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, vol. 5, 2003, pp. 177–180.
[52] I. Arsovski and A. Sheikholeslami, "A current-saving match-line sensing scheme for content-addressable memories," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, 2003, pp. 304–305.
[53] ——, "A mismatch-dependent power allocation technique for match-line sensing in content-addressable memories," *IEEE J. Solid-State Circuits*, vol. 38, no. 11, pp. 1958–1966, Nov. 2003.
[54] P.-F. Lin and J. B. Kuo, "A 0.8-V 128-kb four-way set-associative two-level CMOS cache memory using two-stage wordline/bitline-oriented tag-compare (WLOTC/BLOTC) scheme," *IEEE J. Solid-State Circuits*, vol. 37, no. 10, pp. 1307–1317, Oct. 2002.
[55] J. M. Rabaey, A. Chandrakasan, and B. Nikolić, *Digital Integrated Circuits: A Design Perspective*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 2003.
[56] H. Noda, K. Inoue, M. Kuroiwa, A. Amo, A. Hachisuka, H. J. Mattausch, T. Koide, S. Soeda, K. Dosaka, and K. Arimoto, "A 143 MHz 1.1 W 4.5 Mb dynamic TCAM with hierarchical searching and shift redundancy architecture," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, 2004, pp. 208–209.
[57] K. J. Schultz and P. G. Gulak, "Architectures for large-capacity CAMs," *Integration, VLSI J.*, vol. 18, no. 2–3, pp. 151–171, Jun. 1995.
[58] M. Motomura, J. Toyoura, K. Hirata, H. Ooka, H. Yamada, and T. Enomoto, "A 1.2-million transistor, 33-MHz, 20-bit dictionary search processor with a 160 kb CAM," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, 1990, pp. 90–91.
[59] ——, "A 1.2-million transistor, 33-MHz, 20-b dictionary search processor (DISP) with a 160-kb CAM," *IEEE J. Solid-State Circuits*, vol. 25, no. 5, pp. 1158–1165, Oct. 1990.
[60] K. J. Schultz and P. G. Gulak, "Fully parallel multi-megabit integrated CAM/RAM design," in *Records IEEE Int. Workshop on Memory Technology, Design and Testing*, 1994, pp. 46–51.
[61] ——, "Fully parallel integrated CAM/RAM using preclassification to enable large capacities," *IEEE J. Solid-State Circuits*, vol. 31, no. 5, pp. 689–699, May 1996.
[62] R. Panigrahy and S. Sharma, "Reducing TCAM power consumption and increasing throughput," in *Proc. Symp. High Performance Interconnects*, 2002, pp. 107–112.
[63] F. Zane, G. Narlikar, and A. Basu, "CoolCAMs: power-efficient TCAMs for forwarding engines," in *Proc. IEEE INFOCOM*, vol. 1, 2003, pp. 42–52.
[64] C.-S. Lin, J.-C. Chang, and B.-D. Liu, "Design for low-power, low-cost, and high-reliability precomputation-based content-addressable memory," in *Proc. Asia-Pacific Conf. Circuits Syst.*, vol. 2, 2002, pp. 319–324.
[65] ——, "A low-power precomputation-based fully parallel content-addressable memory," *IEEE J. Solid-State Circuits*, vol. 38, no. 4, pp. 654–662, Apr. 2003.
[66] S. Hanzawa, T. Sakata, K. Kajigaya, R. Takemura, and T. Kawahara, "A dynamic CAM—based on a one-hot-spot block code-for millions-entry lookup," in *Symp. VLSI Circuits Dig. Tech. Papers*, 2004, pp. 382–385.
[67] ——, "A large-scale and low-power CAM architecture featuring a one-hot-spot block code for IP-address lookup in a network router," *IEEE J. Solid-State Circuits*, vol. 40, no. 4, pp. 853–861, Apr. 2005.
[68] H. Liu, "Efficient mapping of range classifier into ternary-CAM," in *Proc. Symp. High Performance Interconnects*, 2001, pp. 95–100.
[69] ——, "Routing table compaction in ternary cam," *IEEE Micro*, vol. 22, no. 1, pp. 58–64, Jan.–Feb. 2002.
[70] V. Ravikumar and R. N. Mahapatra, "TCAM architecture for IP lookup using prefix properties," *IEEE Micro*, vol. 24, no. 2, pp. 60–69, Mar.–Apr. 2004.
[71] A. Sheikholeslami and P. G. Gulak, "A survey of circuit innovations in ferroelectric random-access memories," *Proc. IEEE*, vol. 88, no. 5, pp. 677–689, May 2000.
[72] S. Tehrani, J. M. Slaughter, M. DeHerrera, B. N. Engel, N. D. Rizzo, J. Salter, M. Durlam, R. W. Dave, J. Janesky, B. Butcher, K. Smith, and G. Grynkewich, "Magnetoresistive random access memory using magnetic tunnel junctions," *Proc. IEEE*, vol. 91, no. 5, pp. 703–714, May 2003.
[73] K. Kim and G. Jeong, "Memory technologies in the nano-era: challenges and opportunities," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, 2005, pp. 576–577.

Kostas Pagiamtzis (S'98) received the B.A.Sc. degree (with honors) in computer engineering from the Division of Engineering Science in 1999 and the M.A.Sc. degree in electrical and computer engineering in 2001, both from the University of Toronto, Toronto, ON, Canada. He is currently working toward the Ph.D. degree in the Department of Electrical and Computer Engineering, University of Toronto.

He spent 16 months as a student intern with the Memory Development Group, Nortel Networks, in 1998. He spent the summer of 1999 at SiberCore Technologies, Ottawa, Canada working on content-addressable memories. His primary research interest is in architecture and circuit design for content-addressable memories. He is also interested in the design of emerging nonvolatile memories including ferroelectric and magnetoresistive memories, and circuits and architectures for field-programmable gate arrays (FPGAs).

Mr. Pagiamtzis has held the Natural Sciences and Engineering Research Council of Canada (NSERC) PGS A and NSERC PGS B postgraduate scholarships and the Ontario Graduate Scholarship in Science and Technology (OGSST). He was the co-creator and Symposium Chair of *Connections 2005*, the inaugural Symposium for graduate students in the Department of Electrical and Computer Engineering, University of Toronto. He received the Teaching Assistant Award in 2002, 2003, and 2005 by the popular vote of the undergraduate students in the Department of Electrical and Computer Engineering, University of Toronto.

Ali Sheikholeslami (S'98–M'99–SM'02) received the B.Sc. degree from Shiraz University, Shiraz, Iran, in 1990 and the M.A.Sc. and Ph.D. degrees from the University of Toronto, Toronto, ON, Canada, in 1994 and 1999, respectively, all in electrical and computer engineering.

In 1999, he joined the Department of Electrical and Computer Engineering, University of Toronto, where he is currently an Associate Professor. His research interests are in the areas of analog and digital integrated circuits, high-speed signaling, VLSI memory design (including SRAM, DRAM, and CAM), and ferroelectric memories. He has collaborated with industry on various VLSI design projects in the past few years, including work with Nortel, Canada, in 1994, with Mosaid, Canada, since 1996, and with Fujitsu Labs, Japan, since 1998. He is currently spending the first half of his sabbatical year with Fujitsu Labs in Japan. He presently supervises three active research groups in the areas of ferroelectric memory, CAM, and high-speed signaling. He has coauthored several journal and conference papers (in all three areas), and holds two U.S. patents on CAM and one U.S. patent on ferroelectric memories.

Dr. Sheikholeslami served on the Memory Subcommittee of the IEEE International Solid-State Circuits Conference (ISSCC) from 2001 to 2004, and on the Technology Directions Subcommittee of the same conference from 2002 to 2005. He presented a tutorial on ferroelectric memory design at the ISSCC 2002. He was the program chair for the 34th IEEE International Symposium on Multiple-Valued Logic (ISMVL 2004) held in Toronto, Canada. He received the Best Professor of the Year Award in 2000, 2002, and 2005 by the popular vote of the undergraduate students in the Department of Electrical and Computer Engineering, University of Toronto. He is a registered Professional Engineer in the province of Ontario, Canada.