Integrated Device Technology, Inc.

PROTECTING YOUR DATA WITH THE IDT49C465 32-BIT Flow-thruEDC™ UNIT

APPLICATION NOTE AN-64

By Tao Lin, Gerard Lyons and Frank Schapfel

INTRODUCTION

A Time for Error-Free Memories

With the advent of high-performance 32-bit RISC and CISC microprocessors, general purpose computing across a wide spectrum of applications software is now easily accessible on a desktop. We can now draw on computer resources which are very sophisticated, multi-tasking systems with distributed processing power, and we no longer must rely on the centralized mini-computers and mainframes for processing horsepower. Both the technical and the commercial computing environments demand the insatiable hunger for processing power.

This increasing demand for sophisticated applications software requires more system memory on a local level. Tightly coupled microprocessors and cache memory are designed for optimized processing throughput, but the cache memory is no substitute for system memory. Cache memory is typically composed of very high-speed static RAMs, with access times of sub 20 nanoseconds or less. System or main memory is almost always comprised of slower but very high-density dynamic RAMs, typically with access times of sub 70 nanoseconds or more, but with four times the density of static RAMs. So, when the state-of-the-art static RAMs are 1 Megabit large, the newest density dynamic RAM is 4 Megabits. Therefore, dynamic RAMs will always provide the most cost-effective implementation for system memory.

Dynamic RAMs, though, are very prone to externally induced errors. These externally induced errors are called soft errors, since they do not cause permanent damage to the memory cell. Soft errors can be induced by system noise, alpha particles and power supply surges. Any of these can cause random data bits to be flipped from “1” to “0”, or vice versa. Although these soft error occurrences may be rare and inconsequential when using small amounts of DRAMs, large DRAM arrays are much more error prone. Also, as seen in Figure 1, larger DRAM components are much more susceptible to soft errors by virtue of their smaller memory cell size. Hardware errors may also occur on system memory boards. These hard errors occur if one RAM component or RAM cell fails and is stuck at “0” or stuck at “1”. Although less frequent, hard errors may cause a complete system shut down.

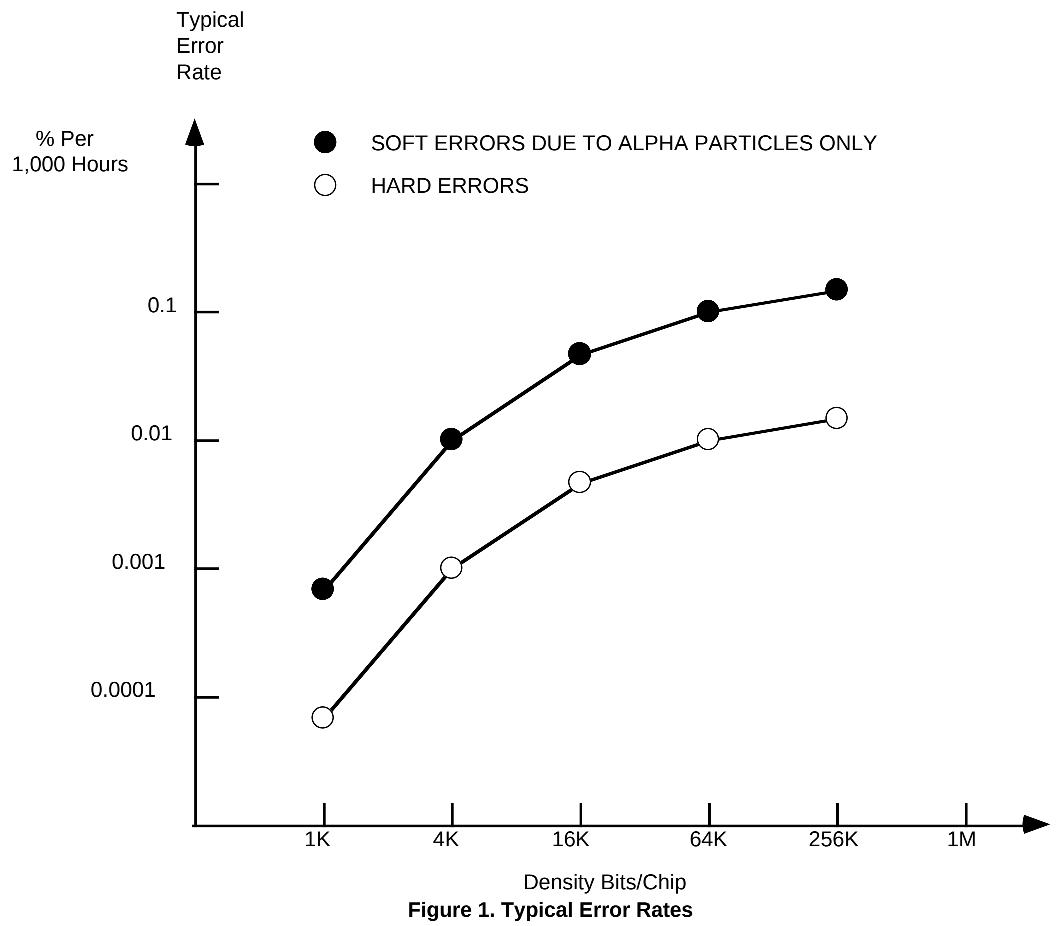


Figure 1. Typical Error Rates

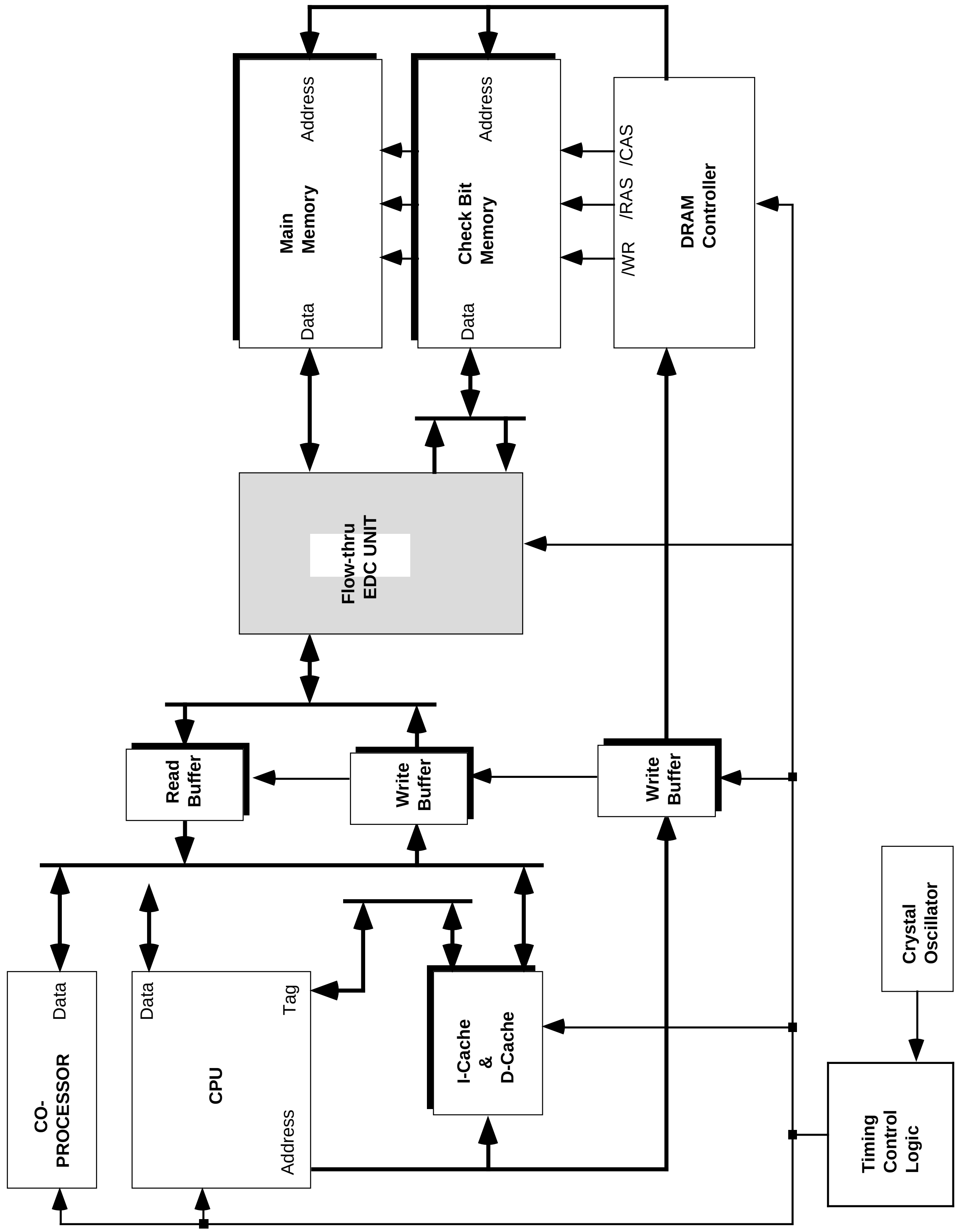


Figure 2. A Typical Architecture of High-performance RISC or CISC Systems

Error Correction to the Rescue

A scheme exists that not only is able to detect soft and hard errors, but is capable of correcting an erroneous bit. This scheme is implemented by a family of error detection and correction chips from Integrated Device Technology. Using a modified Hamming code, developed at AT&T Bell Labs, all single-bit errors may be detected and corrected, while all two-bit and most three-bit errors can be detected. IDT pioneered EDC chips, using CMOS technology in 1986, after recognizing the importance of large DRAM memory arrays in distributed computing.

TYPICAL ARCHITECTURE OF HIGH-PERFORMANCE RISC/CISC SYSTEMS

Figure 2 shows a typical architecture of high-performance RISC or CISC systems which have the following features: (1) high-speed cache memory (separate or common, Instruction-cache and Data-cache) for fast access to frequently used instructions and data, (2) write and read buffers to handle the mismatch between the high-speed CPU and the slow-speed main memory and (3) high-speed flow-thru EDC unit to insure data integrity.

While most high-performance computer systems in current market have the first and second features, the third feature is becoming more attractive and important when the main memory space grows and the memory word-length increases. Certainly, using an EDC unit is an effective way to improve the system reliability.

GENERAL EDC OPERATION

The basic function of an EDC device is to check the integrity of data being read from a memory system, flag an error if one has been detected and if possible correct that error. The IDT family of EDC devices implements this function using the same general principles, with some variations from device to device.

The operation of an EDC device can be generally split into:(1) generation of a coded word based on the data-word being written to memory. This coded word is called **The Check-Bit Word.** This operation is called **Generate;** (2) detection of errors in a data-word read from memory by comparing the corresponding check-bit word read from memory and a newly generated check-bit word (based on the data-word read from memory) and if possible correcting this error. The comparison of these two check-bit words (an exclusive-or (XOR) function) produces a **Syndrome Word**. This operation is called **Detect/ Correct.**

The coding scheme employed in IDT's EDC devices is a modified **Hamming Code.** For each data-word written to memory, a coded pattern, or check-bit word, is appended to the date-word. The new word (the data-word plus the check-bit word) can be termed a **valid code**. The modified Hamming Code establishes a Distance-of-4 between one valid code and another. This means that to go from one valid code to another, 4-bits have to change. It can be shown that a **Distance-of-4** code enables you to **detect all Single and Double-Bit** errors and **correct all Single-Bit** errors**.**

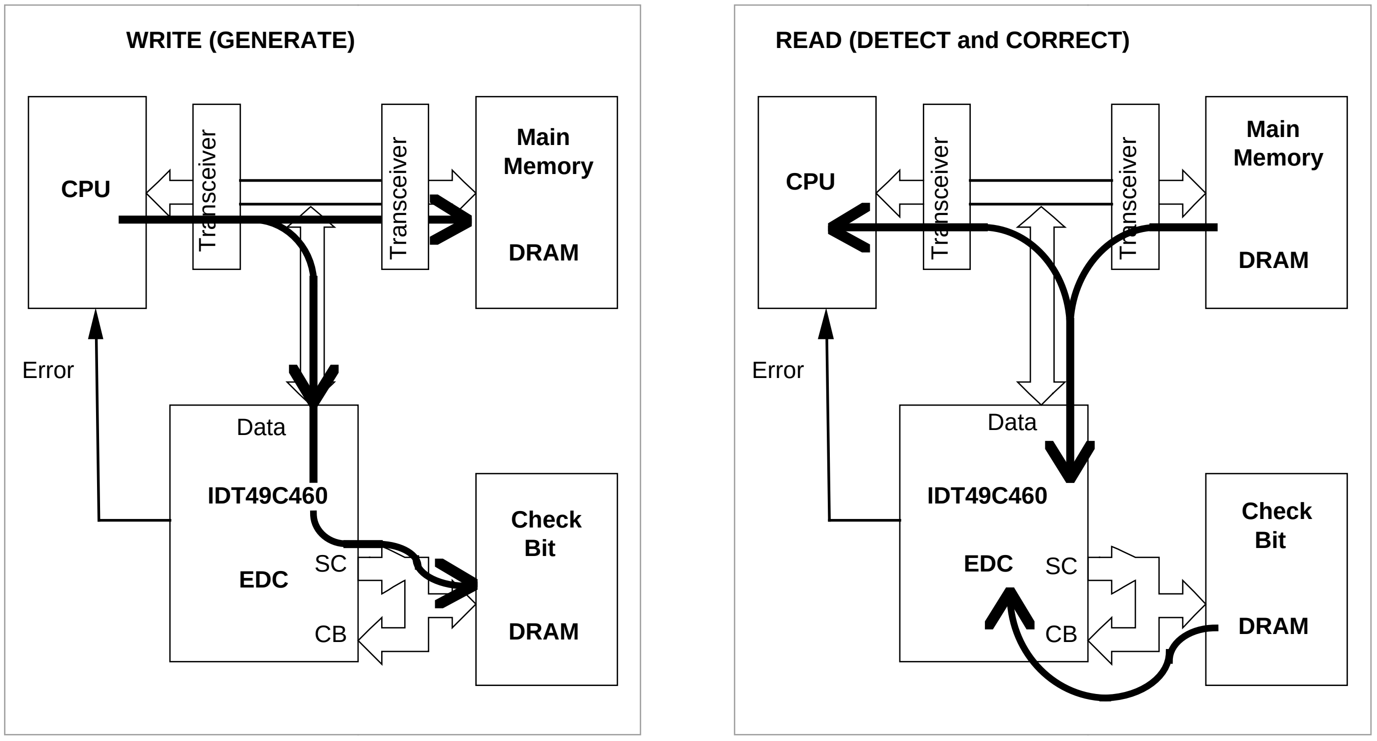
To implement a Distance-of-4 code on a 32-bit data-word, a 7-bit check-bit word must be appended. For a 64-bit word, a 8-bit check-bit word must be appended. The Hamming Code algorithm to generate a check-bit word from a 32-bit data-word or a 64-bit data-word can be found in either IDT49C460 data sheet or IDT49C465 data sheet.

EDC ARCHITECTURES AND WORD-LENGTH

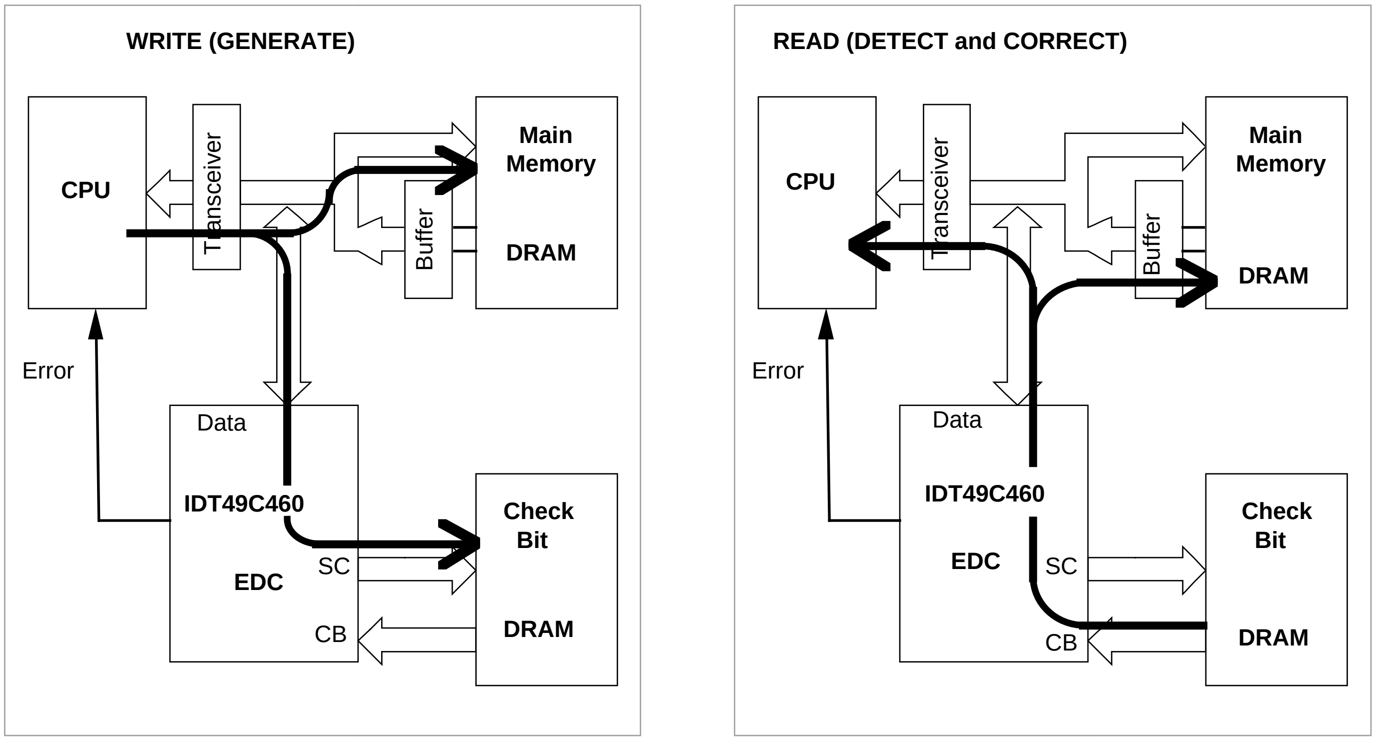
There are two basic architectures for EDC operation: flow-thru and bus-watch. IDT provides a full line of EDC devices to support 16-bit and 32-bit bus-watch architectures and 32-bit and 64-bit flow-thru architectures, as shown in Table 1.

Part Number	Architecture	Word-length	Comment
IDT39C60	Bus-watch	16-bit	Cascadable up to 64-bit using 4 devices
IDT49C460	Bus-watch	32-bit	Cascadable up to 64-bit using 2 devices
IDT49C465	Flow-thru	32-bit	Cascadable up to 64-bit using 2 devices
IDT49C466	Flow-thru	64-bit	

Table 1. IDT EDC Product Line

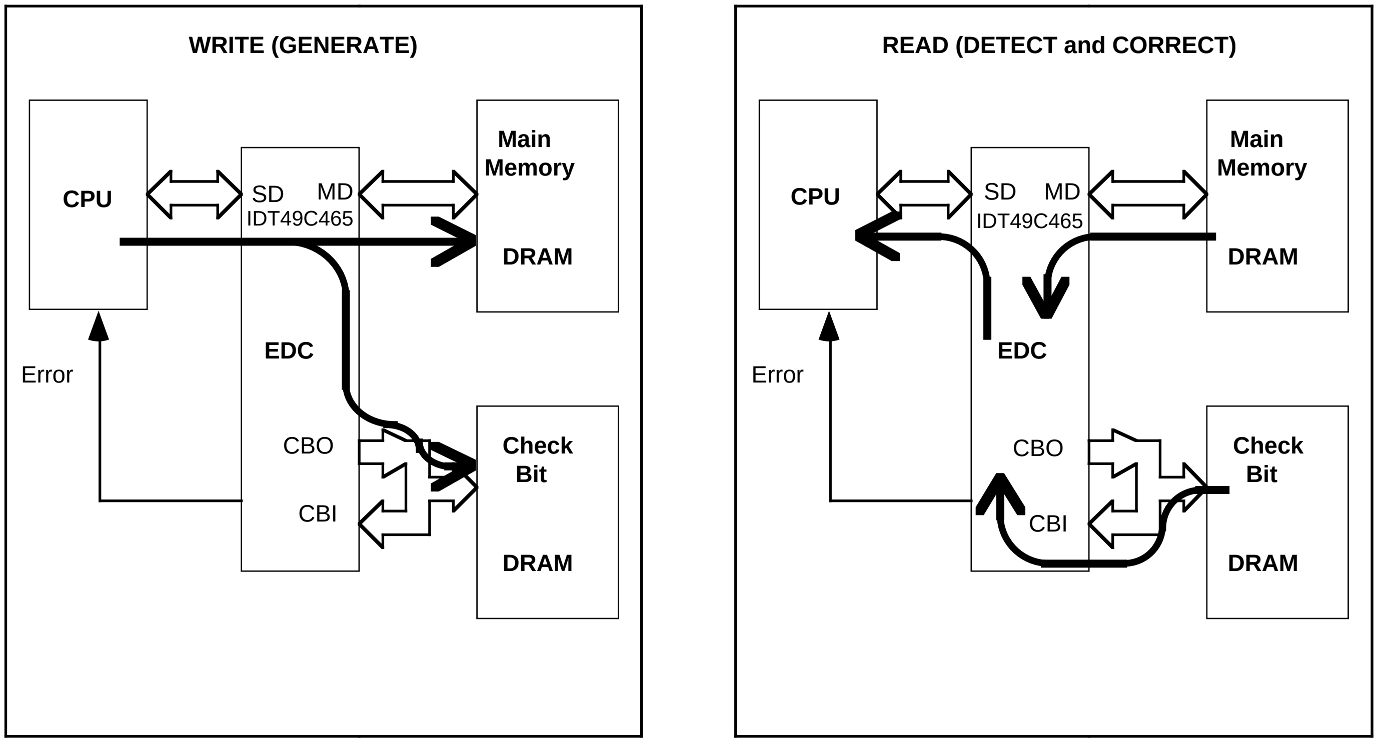


(a) Common I/O Memory System

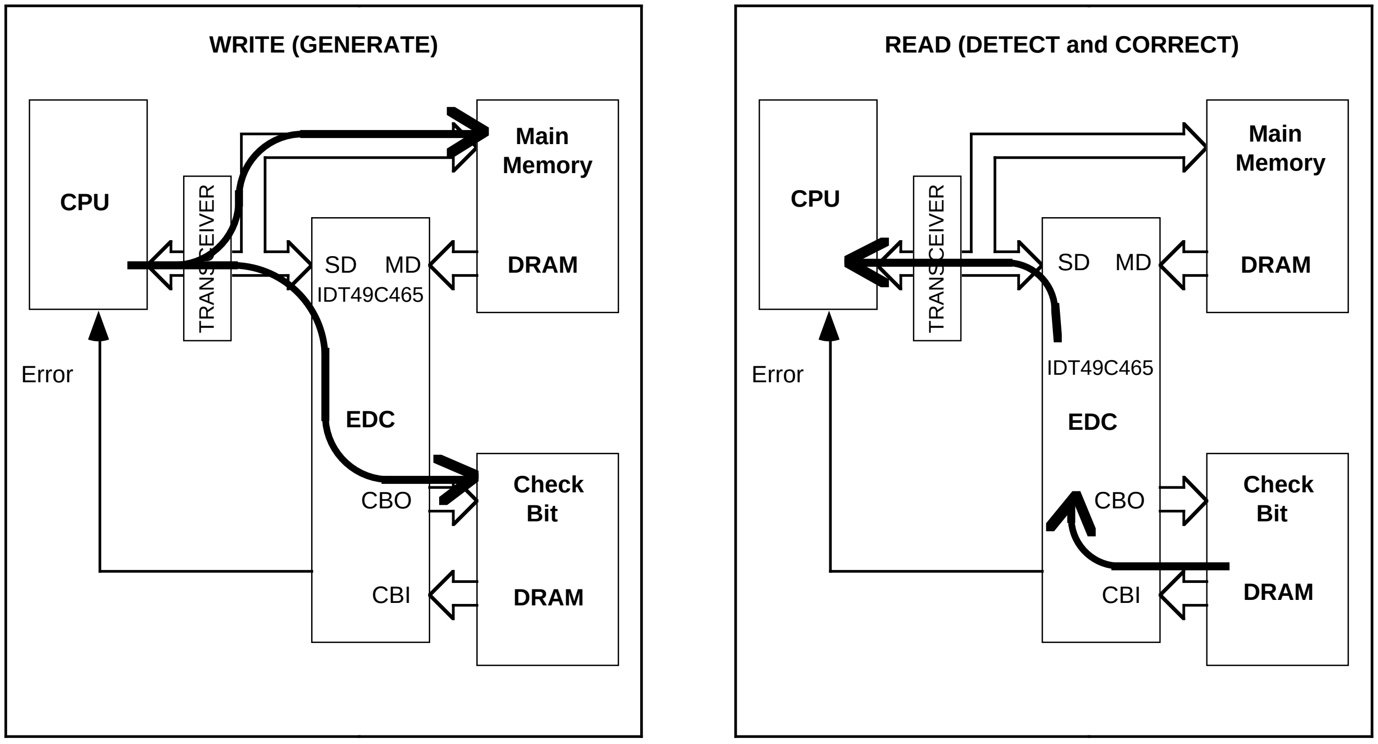


(b) Separate I/O Memory System

Figure 3. Basic Configurations Using a Bus-watch EDC Architecture

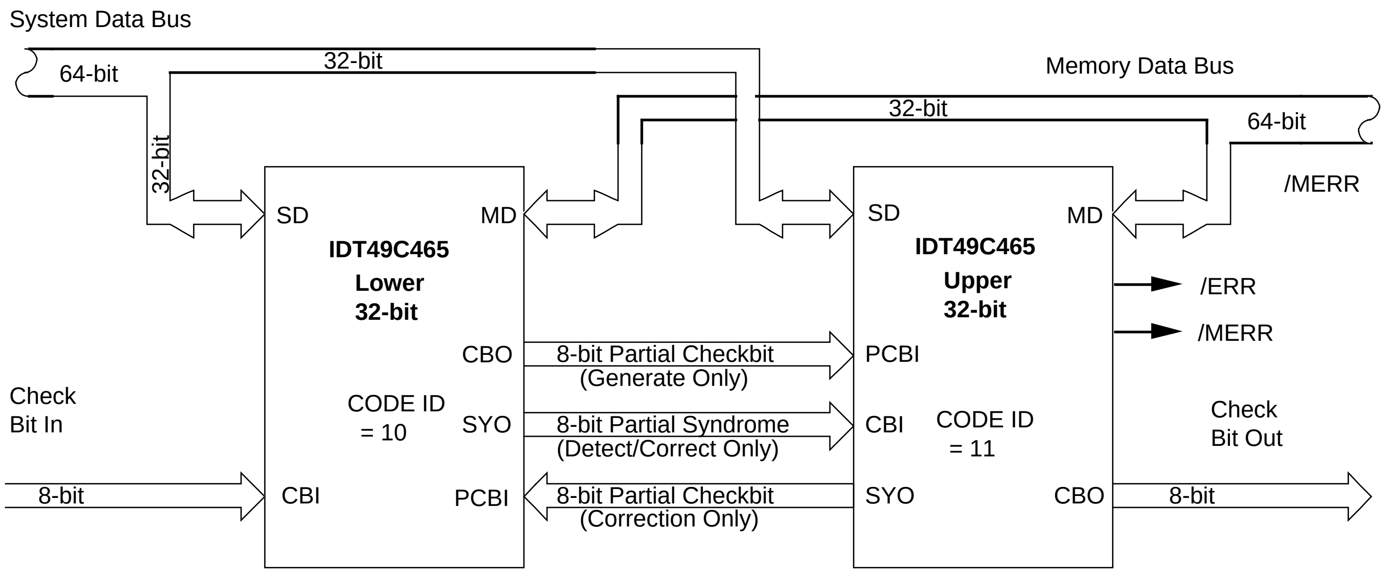


(a) Common I/O Memory System

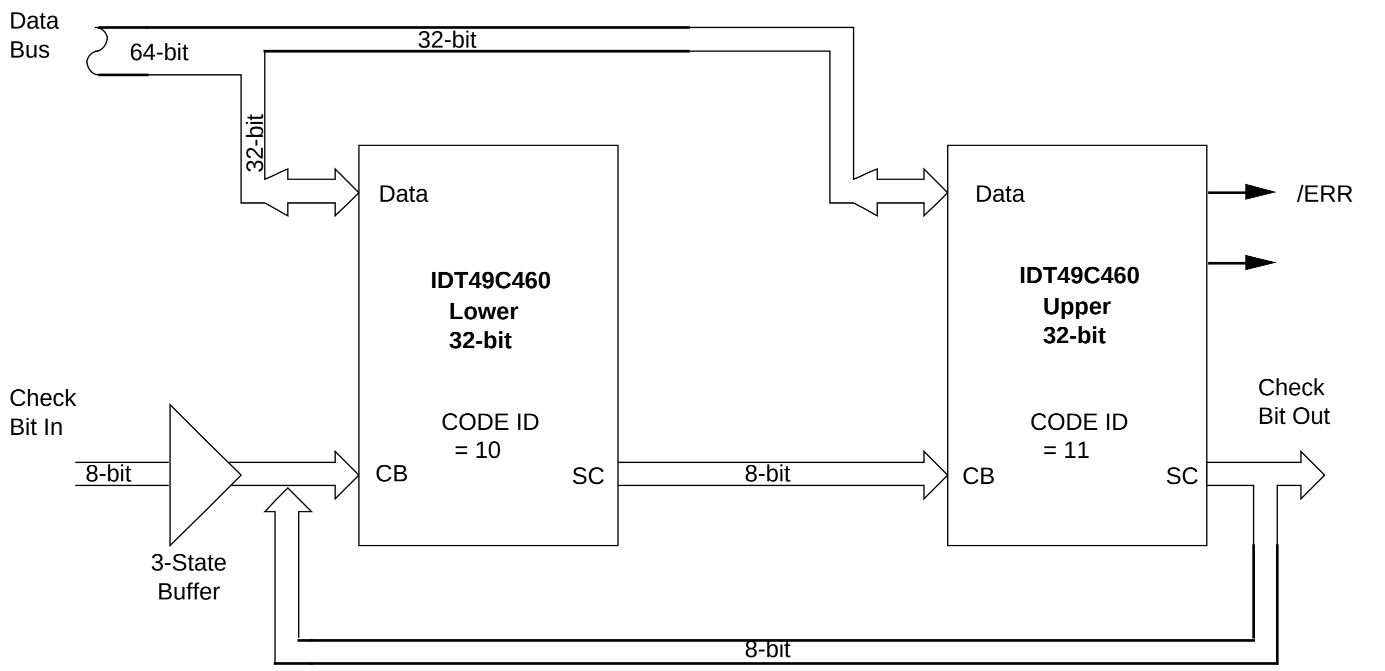


(b) Separate I/O Memory System

Figure 4. Basic Configurations Using a Flow-thru EDC Architecture



(a) Cascading Flow-thru EDC IDT49C465



(b) Cascading Bus-watch EDC IDT49C460

Figure 5. 64-bit Configurations by Cascading Two 32-bit EDC Units

Bus-watch Architecture

A bus-watch EDC such as the IDT49C460 has a single data bus. The basic configurations, using the IDT49C460 for common I/O memory and separate I/O memory, are illustrated in Figure 3.

During a write (store) operation, the CPU sends data to the main memory. At the same time the data goes to the EDC unit, which then generates the check bits and stores them in the check-bit memory.

On the other hand, during a read (load) operation, the data from the main memory and the check bits from the check-bit memory first go to the EDC unit. Based on the information carried by the check bits, the EDC unit can detect all single-bit and some multiple-bit errors, and correct all single-bit errors. On either the main memory data or the check-bit data, the corrected main memory data is then sent to the CPU.

Flow-thru Architecture

In contrast to a bus-watch EDC, a flow-thru EDC such as the IDT49C465 provides two data buses: a system data (SD) bus and a memory data (MD) bus. The dual-bus architecture improves the throughput of the EDC operation and simplifies the interface between the CPU system bus and the memory bus. The basic configurations using the IDT49C465 for common I/O memory and separate I/O memory are illustrated in Figure 4.

In the common I/O configuration, during a write (store) operation, the data from CPU flows through the EDC unit and is written to the main memory. When the data flows through the EDC, the check bits are generated and stored into the check-bit memory. During a read (load) operation, the data from the main memory enters the EDC unit through the MD bus while the check bits enter the EDC unit through the CBI bus. The EDC unit then detects any errors and loads the corrected data to the CPU through the SD bus.

In the separate I/O configuration, during a write (store) operation, the data from CPU are directly sent to the main memory. At the same time, the data is sent to the EDC unit through the SD bus. The EDC unit then generates the check bits and stores them into the check-bit memory. During a read (load) operation, the data from the main memory enters the EDC unit through the MD bus while the check bits enter the EDC unit through the CBI bus. The EDC unit then detects any errors and loads the corrected data to the CPU through the SD bus.

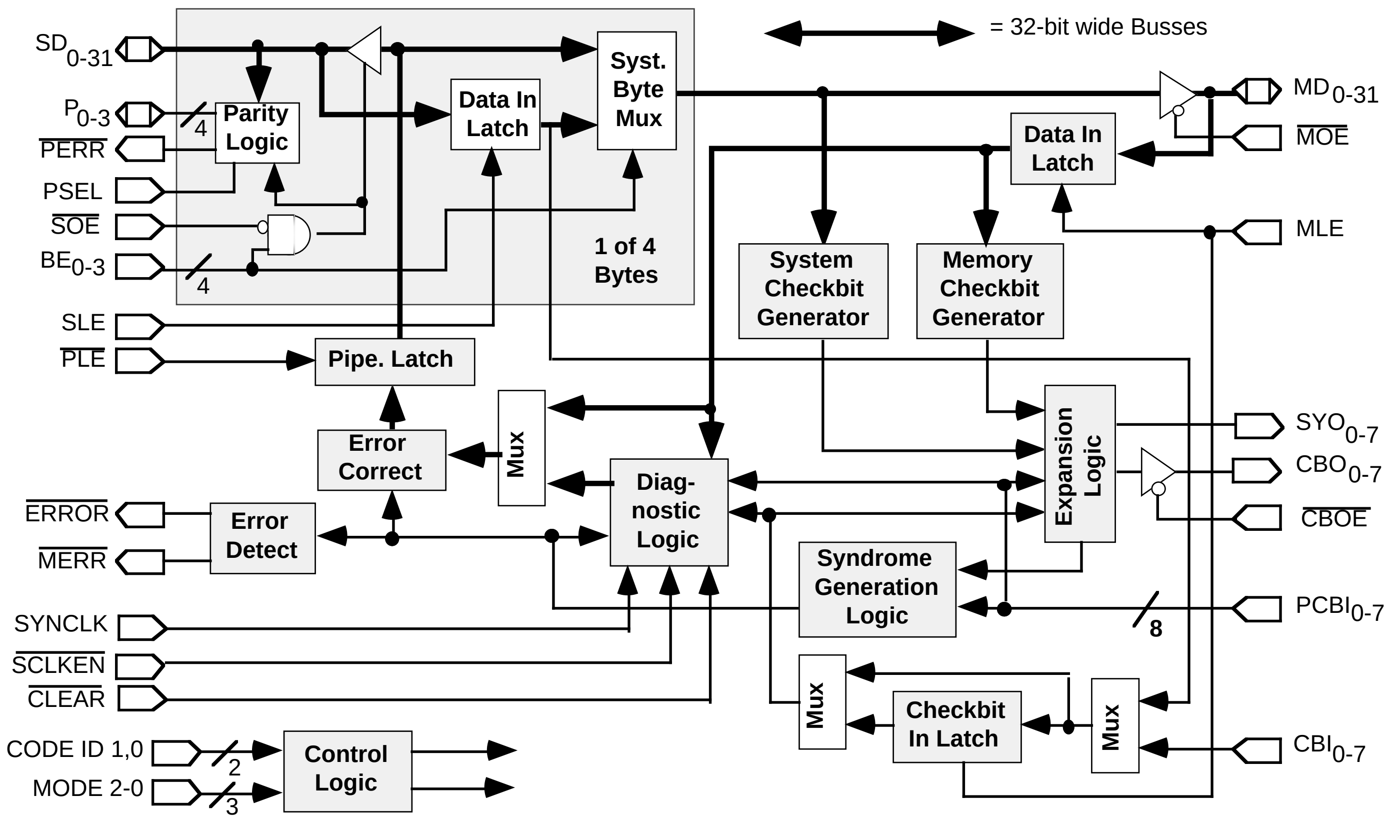


Figure 6. Block Diagram of IDT49C465

Cascading 32-bit EDC Devices for 64-bit Memory Systems

As mentioned in the previous section, for a 32-bit data word, 7 check bits are necessary, while for a 64-bit data word, 8 check bits are needed. Although the IDT49C460 and the IDT49C465 are both 32-bit EDC units, they have an **8-bit output-bus** for output of generated check-bits to memory and an **8-bit input bus** to read back check-bit from memory. In this way, they can be cascaded to support 64-bit applications. In the 32-bit mode, only 7 bits of the check-bit input and output buses are used, while in the 64-bit mode, all 8 bits are used. In 32-bit mode the eighth bit of the CB and SD busses should be tied through a resistor to Ycc or ground.

Figure 5 shows how two IDT49C465s (or two IDT49C460s) can be cascaded to build a complete 64-bit EDC unit. In the cascaded 64-bit mode, the EDC operation can be broken into two stages; a **lower 32-bit** stage and an **upper 32-bit** stage.

For the IDT49C465 (see Figure 5a), a general description of the EDC operation is discussed below.

1. Generation starts by generating a **Partially Generated Check-bit Word** in the lower slice, based on the lower 32-bit of the 64-bit data-word, and sending this to the upper slice. The upper slice combines the Partially Generated Check-bit word from the lower slice, with its generated check-bit word (based on the upper 32-bit of the 64-bit data-word), to form a final check-bit word. Thus, the source of check-bit in a cascaded system is the upper-slice device.
2. Detection/Correction starts in the **lower-slice** where the check-bits from memory are input, as well as the lower 32-bit of the 64-bit data-word. Here the inputted check-bits are compared with the newly generated check-bits (based on the lower 32-bit of DATA-word) using an XOR function to produce a **Partial Syndrome Word,** which is passed onto the upper-slice device. At the same time, in the upper slice the upper 32-bits of 64-bit data-word is used to generated a so-called **Partial Check-Bit Word** which is sent to the lower slice. So now we have in both the upper and lower slice devices almost simultaneously two pieces of data; **the Partial Syndrome Word** (generated in the lower-slice) and the **Partial Check-Bit Word** (generated in the upper-slice). In each slice these two pieces of data are XOR'd to produce a **Final Syndrome Word** which is used to detect and correct errors on the 64-bit data word.

The IDT49C460 carries out Detect/Correct slightly differently (see Figure 5b), namely, the Partial Syndrome generated in the lower-slice is sent to the upper slice, then the Final Syndrome is generated in the upper-slice and this Final-Syndrome is now fed back to the lower-slice. Thus Detect/Correct in the IDT49C460 employs a serial approach, whereas the IDT49C465 uses a faster paralleled approach. Moreover, in the IDT49C460 case, an additional tri-state buffer such as the IDT74FCT244 is needed while in the IDT49C465 case, no additional external logic is needed.

OVERVIEW OF THE IDT49C465 ARCHITECTURE

The IDT49C465 architecture is an evolutionary development on the IDT49C460 EDC device. The IDT49C460 is a single-bus 32-bit EDC cascadable to 64-bits. The IDT49C465 draws on this basic architecture to provide a dual-bus or flow-Thru 32-bit EDC cascadable to 64-bits. Figure 6 shows a block diagram of the IDT49C465; the key difference between the IDT49C460 and the IDT49C465 is the presence of a second 32-bit DATA bus to provide the flow-thru path for data through the device.

Data Buses

The System Data Bus, or **SD Bus,** is a 32-bit bi-directional bus. Data is written to the EDC using this bus for **Check-Bit Generation,** so that when a data-word is written to memory the corresponding check-bits are written simultaneously. Also when a data-word read from memory is corrected, the corrected data-word is read from the SD Bus by the system processor. The SD Bus has associated with parity-checking and generation and also separate byte enables on the SD Bus' output buffers so that Partial Byte operations can be supported.

The Memory Data Bus, or **MD Bus**, is a 32-bit bi-directional bus. Data written from the system processor through the SD Bus can be written to memory using this bus. When the processor is reading a word from memory, the data word is read in through the MD Bus and the corrected data word (depending on the status of the data) is sent to the processor through the SD Bus.

Expansion Buses

The IDT49C465 has four 8-bit buses that are an integral part in the Detect/Correct path for both a 32-bit EDC system and a 64-bit EDC system.

CBI(7:0)

1. When the IDT49C465 is operating as a 32-bit EDC system or is the lower 32-bit slice in a 64-bit EDC system, this 8-bit bus is the input port for Check-Bits read from Memory.
2. When the IDT49C465 is operating as the upper 32-bit slice in a 64-bit EDC system, this bus is the input port for partial Syndromes from the lower slice.

PCBI(7:0)

1. When the IDT49C465 is operating as a 32-bit EDC system, this bus is unused .
2. When the IDT49C465 is operating as the lower 32-bit slice in a 64-bit EDC system, this 8-bit bus is the input port for Partial Check-Bits read from the upper slice. (Reference Figure 5.)
3. When the IDT49C465 is operating as the upper 32-bit slice in a 64-bit EDC system, this bus is the input port for Partially Generated Check-Bits from the lower slice.

CBO(7:0)

1. When the IDT49C465 is operating as a 32-bit EDC system or is the upper 32-bit slice in a 64-bit EDC system, this 8-bit bus is the output port for Check-Bits being written to Memory. (Reference Figure 5.)
2. When the IDT49C465 is operating as the lower 32-bit slice in a 64-bit EDC system, this bus is the output port for Partially Generated Check-bits being sent to the upper slice. (Reference Figure 5.)

SYO(7:0)

1. When the IDT49C465 is operating as a 32-bit EDC system, this 8-bit bus outputs the Final Syndrome word associated with the Detect/Correct logic.
2. When the IDT49C465 is operating as the lower 32-bit slice in a 64-bit EDC system, this bus is the output port for Partial Syndrome word being sent to the upper slice.
3. When the IDT49C465 is operating as the upper 32-bit slice in a 64-bit EDC system, this bus is the output port for Partial Check-bit word being sent to the lower slice.

Operating Modes

The IDT49C465 has 3 mode control pins, MODEID(2:0), which enable the user to select which mode the part is operating in. These modes are summarized in Table 2.

	MODE DESCRIPTION
000	ERROR DATE MODE
X01	DIAGNOSTIC OUTPUT MODE
X10	GENERATE-DETECT MODE
100	CHECK-BIT INJECTION MODE
X11	NORMAL OPERATING MODE

Table 2. IDT49C465 Operating Modes

Error Data Mode (000) :

In this mode the contents of the Error-Data Register are output uncorrected on the SD Bus. The Error-Data Register is a 32-bit register which gets latched under the following conditions: 1. an error condition has been detected by the EDC-ERROR) and is asserted low, 2. the on-chip 4-bit Error-Counter reads zero 0000 (i.e. no error has occurred since the last clear operation), 3. the input signal, SCLKEN), is held low so that the diagnostic clock, SYNCLK, is enabled, 4. the diagnostic clock, SYNCLK, undergoes a LOW-to-HIGH transition.

Data is latched into the Error-Data Register from the output of the Memory Data Latch when and only when these conditions are met. Thus, the Error Data Register contains the Memory Data word corresponding to the first error since a clear operation (assuming SYNCLK has been run continuously). If the Error Data Register has just been cleared, then output of the contents of this register will provide a source of zero-data if that is required.

Diagnostic Output Mode (X01) :

In this mode a 32-bit Diagnostic Word is output on the SD Bus. The structure of this word is outlined in Figure 7.

BITS 0:7	The output of the check-bit multiplexer is output directly on the SD Bus at these positions (LSB at bit 0 and MSB at bit 7).
BIT 8:15	Whatever is being forced on the PCBI(7:0) input pins is output on the SD Bus at these positions (LSB at bit 8 MSB at bit 15).
BIT16:23	The contents of the Syndrome Register, which is an 8-bit register within the Diagnostic Unit, is output on the SD Bus at these positions (LSB at bit 16, MSB at bit 23). The Syndrome Register gets latched at the same time as the Error Data Register and contains the Final Syndrome corresponding to the first error to occur since a clear operation.
BIT 24:27	The contents of the on-chip 4-bit Error-Counter are output on the SD Bus at these positions (LSB at bit 23 and MSB at bit 27). The Error-Counter which gets clocked under the following conditions: 1. An error condition has been detected by the EDC, i.e. xto(ERROR) is asserted low, 2. the on-chip 4-bit Error Counter does not read 1111 (or F HEX), therefore, not more than 16 errors have occurred since the last clear operation, 3. the input signal, xto(SCLKEN), is held low so that the diagnostic clock, SYNCLK is enabled and 4. the diagnostic clock, SYNCLK, undergoes a LOW-to-HIGH transition. The Error Counter will tell the number of errors that have occurred since the last clear operation.
BIT28:29	Reserved
BIT30:31	The contents of the Error Signal Register, which is a 2-bit register is output on the SD Bus at these positions (SB at bit 30 and MSB at bit 31). Bit 0 and bit 1 of the register are set if a Multiple Error has been flagged and bit 1 only is set if a Single Error has been flagged at the same time and under the same conditions as the Error Data and Syndrome Registers are latched.

Error Type		Test		Error Counter				Syndrome Bits Register Contents								Partial Checkbits from Input Pins								Checkbits							
Byte 3								Byte 2								Byte 1								Byte 0							
M	S	—	—	2^3	2^2	2^1	2^0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
31			28	27			24	23							16	15							8	7							

Figure 7. Output Syndrome/Diagnostic Word

Generate-Detect Mode (X10) :

In this mode, detection and generation take place but no correction. Data whether correct or not, passes thru the device from the MD Bus to the SD Bus.

Normal Operating Mode (X11) :

This is the mode where normal detection/correction and generation takes place for a single-slice device (32-bit EDC system) or for the upper and lower slices in a cascaded 64-bit EDC system.

Check-Bit Injection Mode (100) :

In this mode the check-bit multiplexer enables bits 0:7 from the output of the System Data Latch to be fed into the EDC as a check-bit input, normal correction is activated. This is a very useful capability for carrying out a diagnostic check on the detect/correct path of the EDC.

PARITY FOR THE SYSTEM BUS

The IDT49C465 supports byte parity on the SD Bus, with the polarity of the parity (even or odd) selectable using the input pin PSEL. If PSEL is low, then parity (both checking and generation) will be even. If SPSEL is high, then parity will be odd. The part has 4 parity I/O lines one for each byte of the SD Bus and a parity error signal, PERR), which flags a parity error on in-coming data by being asserted low.

Partial Byte Write and Read-Modify-Write Capability

The IDT49C465 supports, through a number of features, **Partial Byte Writes** and **Read-Modify-Writes** cycles. Firstly the SD Bus has 4 Byte Enable signals associated with it, **BE(3:0),** these input lines provide, in conjunction with **SOE)**, separate output enable control on each byte of SD bus data. The BE bus is also the control input to the Sys-Byte-Mux, this mux enables mixing on a byte-by-byte basis of data from the SD latch (A input to mux) and from the Pipe-Line latch (B input to mux). So, for example, if the processor wanted to do a Partial Write or Partial Store of a byte (byte position 3) to a memory location byte position 3 the following sequence would occur: (1) read the memory location in question through the MD Bus and correct if possible or necessary. The corrected data-word will be latched into the Pipe-Line latch, (2) the byte to be written is latched into the SD Latch at byte position 3, all other byte are undetermined, (3) Now we have both pieces of data necessary to construct the 32-bit word to be written to memory and (4) BE(3) is held low and all other BEs are held high. Thus the output of the Sys-byte-Mux is the correctly constructed 32-bit word which is then written to memory through the MD Bus with it's corresponding check-bits.

64-Bit Generate

A very useful and ultimately cost-saving measure associated with the IDT49C465, is its 64-bit generate mode. If the CODE ID of the IDT49C465 is set at 01, the part is configured as a single-slice 64-bit generate EDC. While operating in this mode, the lower 32-bit of the 64-bit data word is input on the MD bus pins and the upper 32-bit of the 64-bit data word is input on the SD bus. The 8-bit generated check bits are output on the CBO bus. In 64-bit generate mode, the EDC is dedicated to check-bit generation, all other features are disabled.

Because the 64-bit generate is executed in a single slice, very fast generate speed can be achieved (15ns as opposed to 30ns in a two-slice 64-bit cascaded system). This feature can also help reduce part count. In 64-bit memory systems, it is common to use 4 32-bit EDC devices; 2 for detect/correct and 2 for generate. With the 64-bit generate capability, this part count is reduced from four to three.

WHY FLOW-THRU EDC

To fully understand the advantages of the IDT49C465 flow-thru EDC over the IDT49C460 bus-watch EDC, it is necessary to first know the architectural differences between the IDT49C465 and the IDT49C460. Figure 8 compares the simplified internal architectures of the two chips. As compared with the IDT49C460, the IDT49C465 has the following unique features:

- Dual data buses
- Dual check-bit generators: one for SD Bus and the other for MD bus
- Independent check-bit generation path
- Independent error detection/correction path
- Dedicated syndrome output
- Dedicated check-bit output
- Output pipeline latch
- Parity check/generation

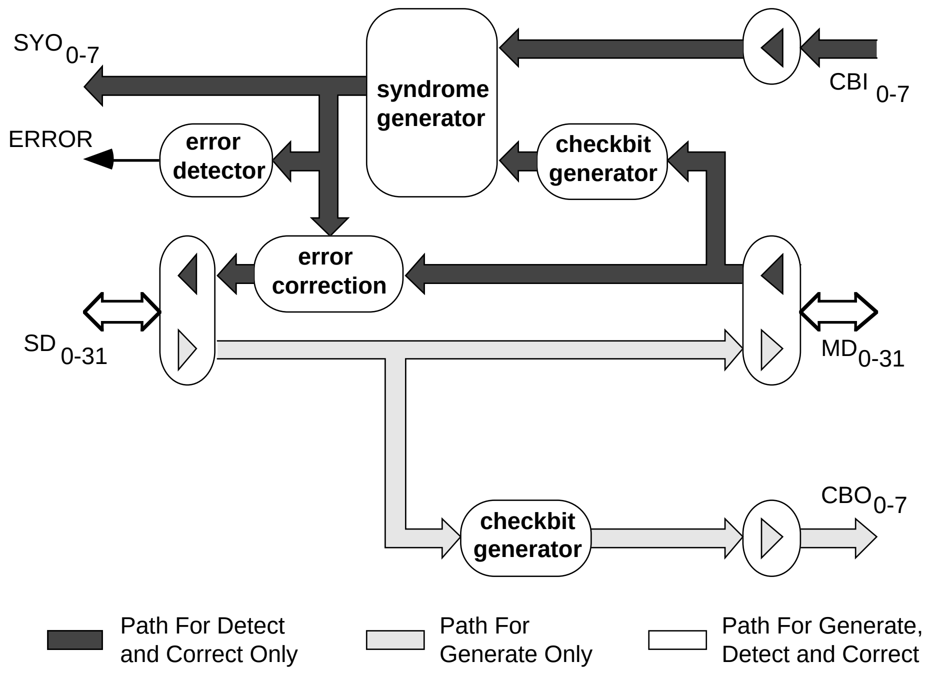
These features greatly simplify the interface of the EDC unit with the system data bus and the memory data bus, and thus can considerably improve the system performance. Generally speaking, in a single bus EDC architecture like the IDT49C460, the data bus connects to both the processor and the memory system. Thus, in a normal correction cycle, data is read into the EDC from memory through the data bus, and the data is corrected. Then, the data bus is enabled as an output and the corrected data is sent to the processor. Therefore, during a correction cycle, the data bus must be turned around from being an input to being an output. Consequently, a single bus architecture has inherent delays associated with the enable/disable times of the data bus output buffer. On the other hand, separate data buses, as in the IDT49C465, allow us to dedicate buses to a specific direction of data flow and, as such, is a superior architecture.

In a 32-bit system using common I/O memory, the dual bus architecture of the IDT49C465 allows direct interface of the flow-thru EDC unit with the system data bus and the memory data bus, as shown in Figure 4a. On the other hand, if the IDT49C460 is used, then two sets of transceivers are needed to buffer both system data bus and memory data bus to the single data bus of the IDT49C460, as shown in Figure 3a.

Similarly, in a 32-bit system using separate I/O memory, the dual bus architecture of the IDT49C465 allows direct interface of the flow-thru EDC unit with the memory data bus. Only a single set of transceivers is used to connect the CPU system data bus to the EDC unit, as shown in Figure 4b. On the other hand, if the IDT49C460 is used, then a set of transceivers and a set of buffers are needed to hook up both system data bus and memory data bus with the single data bus of the IDT49C460.

In particular, the multi-bus architecture and the independent error generation and detection/correction paths of the IDT49C465 provide significant performance improvement in a 64-bit system using two cascaded EDC units. Figure 9 shows the internal data paths of cascaded IDT49C465s and cascaded IDT49C460s during a read (error detect/correct) operation. In the IDT49C465 case, the entire error detect/correct path can be divided into two steps. In the first step, the lower 32-bit unit generates the partial check bits from the lower 32-bit data, and then compares the partial check bits with the original check bits to generate the partial syndrome bits. At the same time, the upper 32-bit unit generates the partial check bits from the upper 32-bit data. Then, the partial syndrome bits from the lower unit and the partial check bits from the upper unit are exchanged between the two units. In the second step, both lower and upper units generate the final syndrome bits independently and then correct errors in the lower 32-bit data and the upper 32-bit data, respectively, in parallel. Therefore, the total delay time is the sum of MD-to-SYO plus CBI-to-SD. On the other hand, in the IDT49C460 case, the entire error detect/correct path can be divided into three steps. In the first step, like in the IDT49C465 case, the lower 32-bit unit generates the partial check bits from the lower 32-bit data, and then compares the partial check bits with the original check bits to generate the partial syndrome bits. At the same time, the upper 32-bit unit generates the partial check bits from the upper 32-bit data. However, in contrast to the IDT49C465 case, only the partial syndrome bits from the lower unit are sent to the upper unit. In the second step, the upper unit compares the partial check bits from the upper 32-bit data with the partial syndrome bits from the lower unit to generate the final syndrome bits. Then, the final syndrome bits are sent back to the lower unit. Finally, in the third step, the lower unit and the upper unit correct the errors in the lower 32-bit data and the upper 32-bit data, respectively. Consequently, the total delay time is the sum of DATA-to-SC plus BC-to-SC plus CB-to-DATA, which is much longer than the delay in the IDT49C465 case. Moreover, since the IDT49C460 has only one check bit input bus, an external octal tri-state buffer is needed to multiplex the original check bits and the partial check bits.

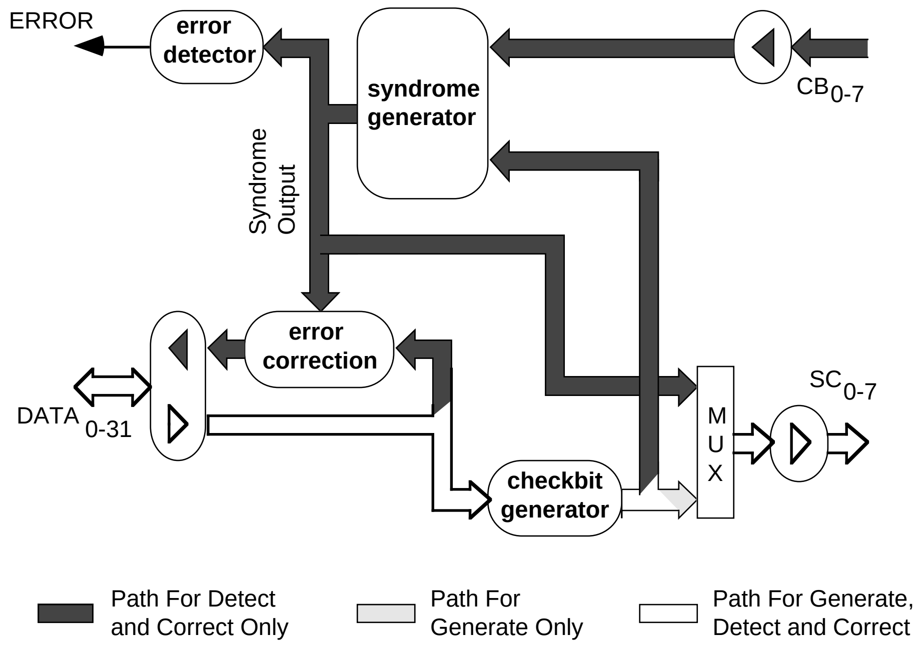
Based on the above discussion, Table 3 summarizes the performance comparison between the IDT49C465 and the IDT49C460D, the fastest version of the IDT49C460. It can be seen that in most situations, the IDT49C465 has significant speed advantage over the IDT49C460.



Features Of IDT49C465

- •Dual Data Bus Architecture
- •Dual CheckBit(CB) Generators
- •Independent CB Generate Path
- •Independent Error Detect/Correct Path
- •Dedicated Syndrome Output
- •Dedicated CB Output
- •Output Pipeline Latch
- •Parity Check/Generate
- •1 Off-chip Feedback for 64-bit Error Correct
- •144-pin PGA

(a) Simplified Block Diagram of IDT49C465 EDC Unit

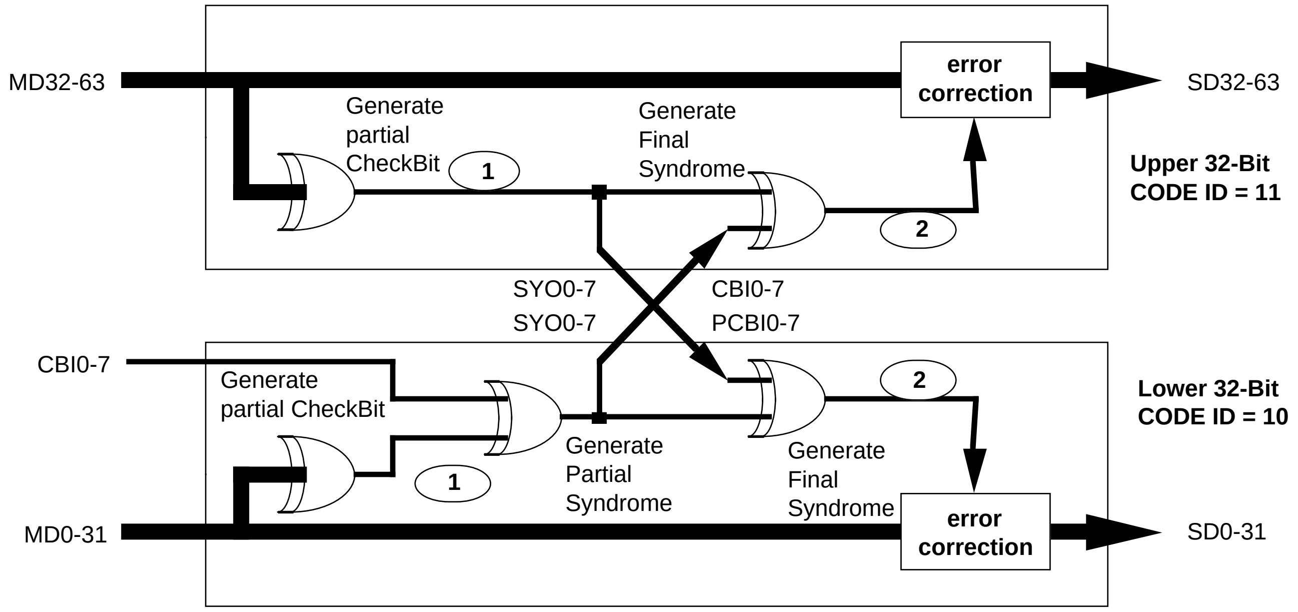


Features Of IDT49C460

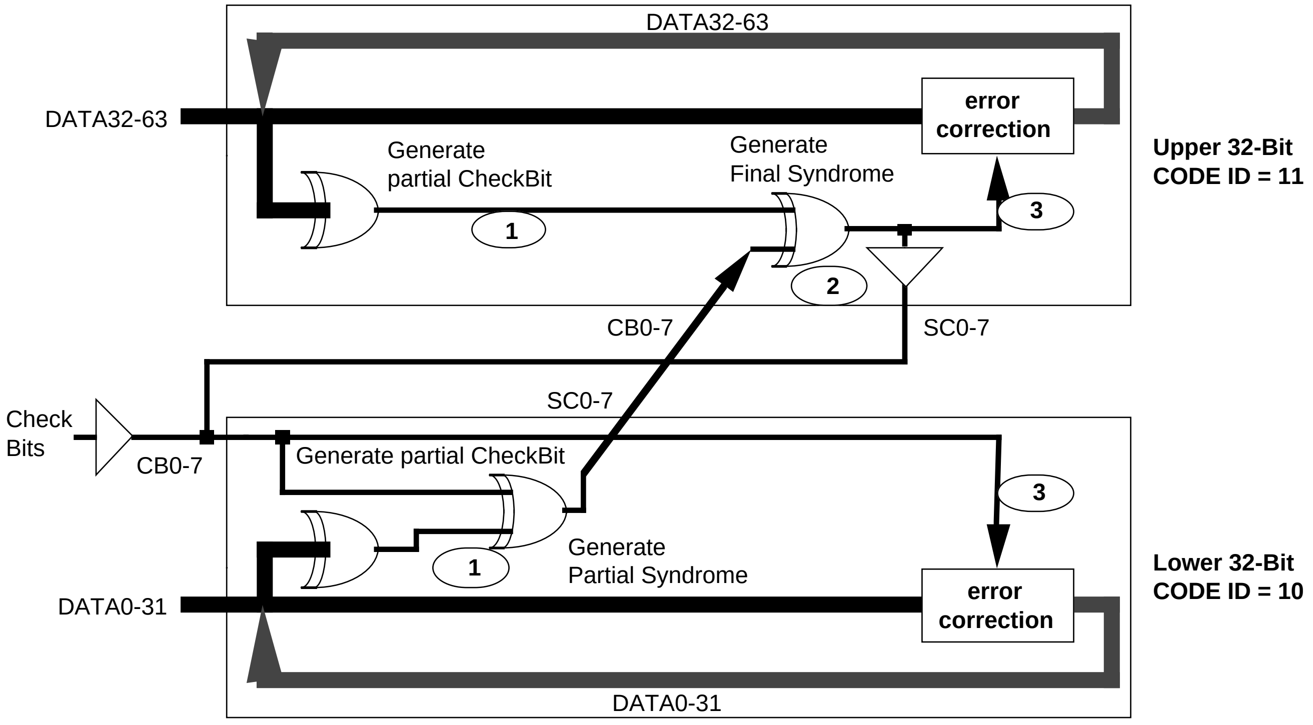
- •Single Data Bus Architecture
- •Single Checkbit Generator
- •Shared Syndrome and CB Output
- •2 off-chip Feedback for 64-bit Error Correct
- •68-pin PGA

(b) Simplified Block Diagram of IDT49C460 EDC Unit

Figure 8. Internal Architecture Differences Between IDT49C465 and IDT49C460



(a) 64-bit Error Detect/Correct Path of Cascading IDT49C465


DATA32-63
DATA32-63
error correction
Generate partial CheckBit
1
Generate Final Syndrome
3
2
Upper 32-Bit CODE ID = 11
CB0-7
SC0-7
Check Bits
CB0-7
SC0-7
Generate partial CheckBit
3
1
Generate Partial Syndrome
DATA0-31
error correction
DATA0-31
Lower 32-Bit CODE ID = 10

(b) 64-bit Error Detect/Correct Path of Cascading IDT49C460

Figure 9. Comparison of 64-bit Error Detect/Correct Path Between IDT49C465 and IDT49C460

	Common I/O				Separate I/O			
	32-bit		64-bit Cascade		32-bit		64-bit Cascade	
	Read [1]	Write	Read [1]	Write	Read[1]	Write	Read[1]	Write
IDT49C 465	MD->SD 20ns **20ns** ***40% Faster***	SD->CBO 15ns **15ns** ***26% Faster***	MD->SYO 15ns CBI->SD 20ns **35ns** ***34% Faster***	SD->CBO 15ns PCBI->CBO 15ns **30ns**	MD->SD 20ns FCT245[2] 5ns **25ns** ***12% Faster***	FCT245[2] 5ns SD->CBO 15ns **20ns**	MD->SYO 15ns CBI->SD 20ns FCT245[2] 5ns **40ns** ***18% Faster***	FCT245[2] 5ns SD->CBO 15ns PCBI->CBO 15ns **35ns**
IDT49C 460D	**28ns** FCT245[2] 5ns D->D 18ns FCT245[2] 5ns	**19ns** FCT245[2] 5ns D->SC 14ns	**47ns** FCT245[2] 5ns D->SC 14ns CB->SC 11ns CB->D 12ns FCT245[2] 5ns	**30ns** FCT245[2] 5ns D->SC 14ns CB->SC 11ns	**28ns** FCT245[2] 5ns D->D 18ns FCT245[2] 5ns	**19ns** FCT245[2] 5ns D->SC 14ns	**47ns** FCT245[2] 5ns D->SC 14ns CB->SC 11ns CB->D 12ns FCT245[2] 5ns	**30ns** FCT245[2] 5ns D->SC 14ns CB->SC 11ns

NOTES:
1. The EDC units perform correction always.
2. FCT245 is high-speed bidirectional transceiver.

Table 3. Performance Comparison

CONCLUSIONS

Whether designing a correct always (flow-thru) EDC or bus-watch EDC memory systems, IDT offers a high performance solution for keeping memory systems error free. The key system benefit for using EDC is the continuous system operation, even with hard or soft errors occur. The key benefit for using a flow-thru EDC is the reduced memory design time when performing the correct always function, and improved performance for 64-bit memory systems.