A Winbond Company

ChipCorder®
TECHNOLOGY BY ISD

Applications Note 7

Message Management in Large Array ISD Devices

ISD Application Note Number 2, **Message Management in the ISD33000** demonstrated a method of recording, playing back and erasing messages in first generation 3-Volt ISD product.. Only 100 bytes of nonvolatile memory were necessary for a reasonable message management scheme. Since each row of memory represented only a maximum of 300 milliseconds, a message block of 8 rows still gave a resolution of 2.4 seconds. The 800 total rows could be addressed by 100 bytes of memory.

ISD devices have continued to grow in size however. Today's ISD4004 has 2400 rows of data and each row has a maximum of 400 milliseconds of storage. It requires a 12-bit address to access all the rows of memory in this device. If the same message management scheme and a comparable row resolution are used, we are pushed into using 2 bytes of memory per pointer. This technique would now require over 1 Kbytes of nonvolatile memory to accomplish this function.

Today's more sophisticated applications also require more features. For example, many messaging systems need multiple mailboxes, messages priorities, etc. Each of these added requirements increases the complexity of the message pointer bytes. This paper will demonstrate another, more memory economical way of achieving these application goals. A comparison of these two approaches will give the software designer insight into different methods of message management.

The technique described below could be used with any of the ISD33000 or ISD4000 Single-Chip Record/Playback device families. This discussion, however, will focus on the ISD4004-16, which is a member of the ISD4004 series. The ISD4004 generic device has 3,840,000 cells in its Flash cell analog memory array. As with all curren t ISD ChipCORDER devices, the resultant record and playback time depends on the speed of the internal sample rate clock. Each analog sample taken is stored away in an analog memory cell. Thus the ISD4004-8 achieves 8 minutes of storage by sampling the array at 8 KHz and the ISD4004-16 achieves 16 minutes by sampling at 4 KHz. See the ISD4004 Series Family data sheet for complete information on this device.

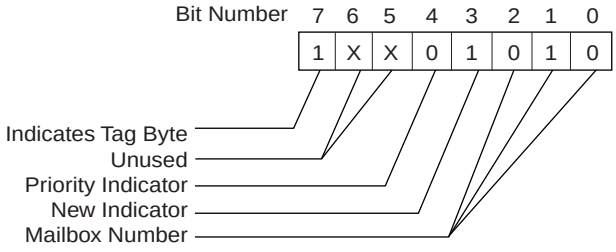
MESSAGE MANAGEMENT PLANNING

The ISD4004-16 has 2400 addressable rows, each 400 milliseconds long. This proposed message management method would group a number of rows into a message block to minimize the size of the memory necessary for pointers. Once a message is recorded into a block or a series of blocks, it cannot be moved to other address locations in the chip. The blocks of memory used may not be contiguous because of previous record and erase operations. Pointers are used to keep track of the beginning address locations of each block of memory in a message. We will call this list of pointers the MAT or Message Address Table.

This type of message management is made possible by the ability of the ISD33000 and ISD4000 series to seamlessly jump from the end of one row to the beginning of any other row in the device's memory during record or playback. This allows us to find unused memory during the record operation and efficiently reuse it, regardless of its location. The information in the MAT allows us to rebuild and playback the recorded message.

Our message management scheme will use two types of data bytes. The first byte in the MAT for any message will be a Tag Byte. . The sign bit, i.e. bit 7 of the Tag Byte will always be a 1. Except for bit 7, this byte may be formatted as required by the needs of the messaging system. The bits in it can be used for any purpose. In our example, we will use the Tag Byte to indicate mailbox number, an unread message and priority of message. Bits 0, 1 and 2 indicate a mail box number, leaving room for 8 different mailboxes. Bit 3 will indicate that the message is new and therefore unread. Bit 4 will be used to indicate a priority message. We will always keep the priority messages ahead of non-priority messages in playback order. Bits 5 and 6 are unused for now but may be used for something else in a "future product." Figure 1 charts the bit positions of a representative Tag Byte.

Figure 1: Example Tag Byte



The Tag Byte above would indicate that our message is non-priority, has not yet been read, and is in mailbox number 2. Note that it is not necessary to keep track of a "number" for the message in this example. We will derive a number for a message by keeping them in order. For example, the first message in the MAT in Mailbox 3 will be message #1 of Mailbox #3. The second message in the MAT in Mailbox 3 will be message #2 of Mailbox #3. If we wish to change the number of a message in a mailbox, we simply reorder the mailbox entries in the MAT.

The second type of byte is the Address Pointer Byte. Bit 7 is always a zero in an Address Pointer Byte. The other 7 bits contain the 7 most significant bits of the beginning address of the memory block this byte points to.

Figure 2:

Example Binary Multiply by 18

The process of multiplying by 18 requires a multiply by 16 of the original number added to a multiply by 2 of the original number. A multiply by 2 is a single left shift operation of the original 12-bit address. A multiply by 16 is 4 bit shift left of the original number. We can then combine the two operations by performing one shift left, storing off the result, then performing 3 additional shift left operations. Then add the 4 bit shifted result to the 1 bit shifted result. We now have the original number multiplied by 18.

Example: 18 x 24 = 432 (= 0001 1011 0000 binary)

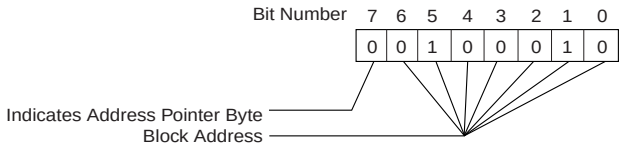
24 = 0000 0001 1000

Shift left 1 = 0000 0011 0000 (mult x 2)
Shift left 4 = 0001 1000 0000 (mult x 16)
Add together = 0001 1011 0000 (mult x 18)

We can determine the minimum possible block size by dividing the total number of addressable rows by the 128 possible combinations in 7 active bits of the Address Pointer Byte. The result of dividing 2400 by 128 is 18.75 rows. This is of course not an integer number so 19 rows per block is the theoretical minimum. In our example message management scheme we are going to assume a block size of 18 rows or 7.2 seconds. When we multiple 128 x 18 we get 2304. This is the number of rows we can address in the ISD4004. That leaves 94 rows or a total of 37.6 seconds we can use for prompts or other non-message storage. The only difficulty in this approach is the generation of the real address for the ISD4004. This requires a multiply by 18 function which may require some effort in a low-end micro. But four 16 bit shifts and one 16 bit add is not a difficult task for most processors. See Figure 2.

An example Address Pointer Byte is shown in Figure 3. The seven bit address 010 0010 (34 decimal) is contained in the Pointer Byte. The real address of the first byte of the message block to be addresses is 0010 0110 0100 (612 decimal) after a multiply by 18.

Figure 3: Example of an Address Pointer Byte



SIZE OF THE MAT

We can now define the size of the MAT. We need 128 bytes to address the message blocks in the analog array. It additionally takes one Tag Byte for each message. If we define the system such that we can have a maximum of 50 messages stored in our ISD4004-16, then it will take a maximum of 178 bytes in the MAT. Our minimum message resolution is 7.2 seconds, which defines the smallest possible message, and the maximum amount of space that might be wasted in a multiple block message.

Table 1: Example MAT

Message Addr.Table (MAT) entry	Tag Byte?	Priority Message?	New Message?	Mail Box #	Block Address		Real Beginning message Block Address (X18)		Msg Nr.
					Binary	Dec	Binary	Dec	
1000 0101	Yes	No	No	3					
0000 0011					000 0011	3	0000 0011 0110	54	#1/3 (20
0000 0110					000 0110	6	0000 0110 1100	108	seconds
0000 1001					000 1001	9	0000 1010 0010	162	long)
1001 0010	Yes	Yes	No	2					
0000 1000					000 1000	8	0000 1001 0000	144	#1/2
0000 1010					000 1010	10	0000 1011 0100	180	(28.2
0000 1011					000 1011	11	0000 1100 0110	198	seconds
0000 1111					000 1111	15	0001 0000 1110	270	long)
1000 0111	Yes	No	No	7					
0000 1000					000 1000	4	0000 0100 1000	72	#1/7 (21
0000 0111					000 0111	7	0000 0111 1110	126	seconds
0000 1110					000 1110	14	0000 1111 1100	252	long)
1000 1010	Yes	No	Yes	2					
0000 0000					000 0000	0	0000 0000 0000	0	#2/2
0000 0001					000 0001	1	0000 0001 0010	18	(50.4
0000 0010					000 0010	2	0000 0010 0100	36	seconds
0000 0101					000 0101	5	0000 0101 1010	90	long)
0000 1100					000 1100	12	0000 1101 1000	216	
0000 1101					000 1101	13	0000 1110 1010	234	
0001 0000					001 0000	16	0001 0010 0000	288	
0000 0000 155 bytes, all zeros									

An example MAT is shown in Table 1. It indicates that 4 messages are currently stored in the chip. They are of different lengths and fragmented over the first 17 message blocks of the memory array. Three of the 4 messages have been read; only Message #2 of Mailbox #2 is unread. There are 21 bytes currently used in the MAP leaving 155 bytes for future message storage. Note that the MAP will always fill from the top down. Unused memory will always be at the bottom of the MAT table.

OTHER POSSIBILITIES

It is certainly possible to set up the message management system with larger or smaller blocks and for a larger or smaller number of messages. There may be more Message Address Pointer memory available enabling a smaller message block and more efficient message storage. Conversely, there may be less memory available requiring a larger block size. The determination of the exact system used is left up to the system designer.

PLAYBACK EXAMPLE

We chose to playback any unread message in Mailbox #2. The software will search the Tag Bytes, only looking for those from Mailbox #2 with a set bit 3 (New Message bit). In this case, it will find the 4th message in the MAT, which is also the second message in Mailbox #2. Next, the software will clear bit 3 of the current Tag Byte to indicate the message has been read, and set up to playback the message. The first message block is at address zero, and message playback will begin with the execution of a single 3 byte command[1].

As soon as playback is started, the microcontroller software should start polling the RAC pin of the ISD4004. The RAC pin will go LOW 50 milliseconds before the end of each playback row and go back HIGH exactly at the end of each memory row[2] . Since we know how long the RAC pin is LOW, a poll repetition rate of approximately 25 milliseconds is a good choice. We now count 17 RAC pin cycles. At the end of the 17th cycle, after the RAC pin has gone back HIGH, we can look in the MAT for the address of the next block in this message. In this example, the second block of the message is also the second block of memory in the chip. We could detect this in the software and just allow the playback to continue on into the next block without further action. Our example however will use the more general case and will always load the next address regardless of where it is located.

After the completion of the 17th RAC cycle, we are playing back the 18th (and last) row of this block. To avoid confusion, please note that in this specific example, the 18 row has the row address of 17, since we started playback with address zero. Our MAT tells us that the address of the next block for playback is "1". After our 18X multiply, we know the real starting address of this block is 18 decimal. We can now load a new play at an address command into the device to cause it to next playback at row 18. When the end of the current row (row 17) is reached, the ISD4004 "jumps" to address 18 and continues playback.

The more interesting jump occurs at the end of block 2 when the chip jumps to the beginning of block 5, or address 90 decimal.

The last block in this message is Block 16 (which is also the last block in the current MAT). After the jump to address 288 decimal, playback will proceed through this block until a set EOM bit is encountered. At that point, playback will end and an EOM Interrupt will occur.

The only change to the MAT after this playback operation is that the second message in Mailbox 2 will now be marked as "read". That is, the Tag Byte will now be 1000 0010 (82 hex). Table 2 shows the new Tab Byte entry for this message.

1. Note that the other devices in the ISD4000 series require only a 2-byte command because of their smaller size and addressing range. Also, the ISD4000 series automatically sets the IAB bit after the execution of a Play or Record command at an address. The older ISD33000 series required a second Record or Play command just to set the IAB bit.

2. See the ISD4004 series data sheet for exact timings.

Table 2: New Tab Byte Entry

Message Addr. Table (MAT) entry	Tag Byte?	Priority Message?	New Message?	Mail Box #	Block Address		Real Beginning message Block Address (X18)		MSG Nr.
					Binary	Dec	Binary	Dec	
≈									
1000 0010	Yes	No	No	2					
≈									

ERASE EXAMPLE

We will now show how a message is erased and its memory space reclaimed. To demonstrate this, we will erase the message in Mailbox #7 (See Table 1). The erase routine must first determine how many bytes are in this message's MAT table entry. The MAT entry for this message contains 4 bytes, i.e. one Tag Byte and 3 Address Pointer Bytes. After this determination, the erase routine simply shifts all the following bytes up the determined number of places, in this case 4 bytes. This "erases" the message by removing the MAT entry for this message. After the message is erased, the MAT will look as indicated in Table 3 [3]. Blocks 4, 7 and 14 are no longer used and may be used in a future message.

3. The second message in Mailbox 2 is being shown as 'new' for the purposes of the following Record Example.

Table 3: Erase Example

Message Addr. Table (MAT) entry	Tag Byte?	Priority Message?	New Message?	Mail Box #	Block Address		Real Beginning message Block Address (X18)		Msg Nr.
					Binary	Dec	Binary	Dec	
1000 0101	Yes	No	No	3					
0000 0011					000 0011	3	0000 0011 0110	54	#1/3 (20
0000 0110					000 0110	6	0000 0110 1100	108	seconds
0000 1001					000 1001	9	0000 1010 0010	162	long)
1001 0010	Yes	Yes	No	2					
0000 1000					000 1000	8	0000 1001 0000	144	#1/2
0000 1010					000 1010	10	0000 1011 0100	180	(28.2
0000 1011					000 1011	11	0000 1100 0110	198	seconds
0000 1111					000 1111	15	0001 0000 1110	270	long)
1000 1010	Yes	No	Yes	2					
0000 0000					000 0000	0	0000 0000 0000	0	#2/2
0000 0001					000 0001	1	0000 0001 0010	18	(50.4
0000 0010					000 0010	2	0000 0010 0100	36	seconds

Table 3: Erase Example (Continued)

Message Addr. Table (MAT) entry	Tag Byte?	Priority Message?	New Message?	Mail Box #	Block Address		Real Beginning message Block Address (X18)		Msg Nr.
					Binary	Dec	Binary	Dec	
0000 0101					000 0101	5	0000 0101 1010	90	long)
0000 1100					000 1100	12	0000 1101 1000	216	
0000 1101					000 1101	13	0000 1110 1010	234	
0001 0000					001 0000	16	0001 0010 0000	288	
0000 0000 155 bytes, all zeros									

RECORD EXAMPLE

We will now show how a new message is recorded. The MAT will initially be configured as shown in Table 3. A new non-priority message in Mailbox 2 will be recorded. The TAG Byte would therefore be 1000 1010 (8A hex). It will be loaded into the MAT immediately after the last currently valid Address Pointer Byte. Note that this Tag Byte is identical to the Tag Byte in the second message already in Mailbox #2. This message is also is non-priority and unread.

Our software must next determine the location of the first free block of memory residing in the ISD4004-16M. There are numerous clever methods of accomplishing this type of search. The simplest is to start searching the Address Pointer Bytes for an unused block 0, then an unused block 1, then 2 etc. From Table 2, we can see that blocks 0 through 3 are already in use. The first free block is block 4. The first Address Pointer Byte in this message will then be 0000 0100. The address of the beginning of block 4 is (18x4) decimal 72. This is a real beginning message block address of 00 0100 1000 (hex 048). A 3-byte command at that address begins recording of the message.

Once again we begin counting RAC cycles. At the end of the 17 RAC cycle, and after the RAC pin goes back HIGH signaling that we are now recording on the last row of block 4. We can now calculate the next block address jump. Our search reveals that the next available block address is address 7. We again go through the address calculation and determine that this block has a decimal address of 126 and real beginning message block address of 00 0111 1110 (hex 07E). A 3-byte command at that address is immediately issued (and must be issued before the end of the current row). The chip will continue to record on the current row until it is complete. When the RAC cycle completes and RAC goes HIGH, recording will continue in the new block 7.

This sequence continues as long as the user wishes to record with blocks 14, 17, 18, 19 etc. being used. When the record process is stopped, an EOM bit is recorded into the chip's digital EOM memory. For the purposes of this example, we are assuming that recording stopped somewhere in block 19. That means the recorded message is between 36 and 43.2 seconds in length. The new MAT is shown in Table 4.

Table 4: Record Example

Message Addr. Table (MAT) entry	Tag Byte?	Priority Message?	New Message?	Mail Box #	Block Address		Real Beginning message Block Address (X18)		Msg Nr.
					Binary	Dec	Binary	Dec	
1000 0101	Yes	No	No	3					
0000 0011					000 0011	3	0000 0011 0110	54	#1/3 (20
0000 0110					000 0110	6	0000 0110 1100	108	seconds
0000 1001					000 1001	9	0000 1010 0010	162	long)
1001 0010	Yes	Yes	No	2					
0000 1000					000 1000	8	0000 1001 0000	144	#1/2
0000 1010					000 1010	10	0000 1011 0100	180	(28.2
0000 1011					000 1011	11	0000 1100 0110	198	seconds
0000 1111					000 1111	15	0001 0000 1110	270	long)
1000 1010	Yes	No	Yes	2					
0000 0000					000 0000	0	0000 0000 0000	0	#2/2
0000 0001					000 0001	1	0000 0001 0010	18	(50.4
0000 0010					000 0010	2	0000 0010 0100	36	seconds
0000 0101					000 0101	5	0000 0101 1010	90	long)
0000 1100					000 1100	12	0000 1101 1000	216	
0000 1101					000 1101	13	0000 1110 1010	234	
0001 0000					001 0000	16	0001 0010 0000	288	
1000 1010	Yes	No	Yes	2					
0000 1000					000 1000	4	0000 0100 1000	72	#3/2 (36
0000 0111					000 0111	7	0000 0111 1110	126	To 43.2
0000 1110					000 1110	14	0000 1111 1100	252	Seconds
0000 0000					001 0001	17	0001 0011 0010	306	Long)
0000 0001					001 0010	18	0001 0100 0100	324	
0000 0010					001 0011	19	0001 0101 0110	342	
0000 0000 152 bytes, all zeros									

CONCLUSION

The above explanation has shown a method of message management that will work on large array ISD devices such as the ISD4004-16M. This technique allows a user maximum flexibility in handling recording, playback and erasure of messages while requiring a minimum of processing bandwidth from the host microcontroller. Other methods of message management are possible. Our customers are invited to try their hand at other clever methods of using this device.

欢迎索取免费详细资料、设计选型指南和光盘、样品；产品繁多未能尽录，欢迎来电查询。

中国传感器科技信息网：HTTP：//WWW.SENSOR-IC.COM/
工控安防网：HTTP：//WWW.PC-PS.NET/
消费电子专用电路网：HTTP://WWW.SUNSTARE.COM/
E-MAIL： xjr5@163.com szss20@163.com
MSN: suns8888@hotmail.com
QQ: 195847376
地址： 深圳市福田区福华路福庆街鸿图大厦 1602 **室**
电话：0755-83376549 83376489 83387030 83387016
传真：0755-83376182 83338339 **邮编：**518033 **手机：**(0)13902971329
深圳展销部：深圳华强北路赛格电子市场 2583 **号** TEL/FAX：0755-83665529 25059422
北京分公司：北京海淀区知春路 132 **号中发电子大厦** 3097 **号**
TEL：010-81159046 82615020 13501189838 FAX：010-82613476
上海分公司：上海市北京东路 668 **号上海赛格电子市场** 2B35 **号**
TEL：021-28311762 56703037 13701955389 FAX：021-56703037
西安分公司：西安高新开发区 20 **所(中国电子科技集团导航技术研究所)**
西安劳动南路 88 **号电子商城二楼** D23 **号**
TEL：029-81022619 13072977981 FAX:029-88789382
成都：TEL:(0)13717066236
技术支持：0755-83394033 13501568376