January 26, 1996

MC68302 Errata

Revision B (Masks 2B14M, 3B14M, 4B14M)

ERRATA

1. SCCE Registers

Summary:
A previously undocumented characteristic of the MC68302 is as follows. It is possible for an SCCE bit to remain set, even though it was cleared in software. This can cause a spurious interrupt request from the SCC. This spurious interrupt is not serious, and can be safely ignored in most interrupt handlers.

Details:
If the M68000 core is moving a one to a bit in one of the three SCCE registers in order to clear the event bit, and, at the same time, the RISC is setting that exact same bit in one of the other two SCCE registers, that event bit will remain set. Example: If the M68000 core executes a "MOVE.B #$01,SCCE1" to clear bit 0 in the SCCE1, and at the same time the RISC is setting bit 0 in either the SCCE2 or SCCE3, then the SCCE1 bit 0 will remain set. Once the interrupt handler completes, a spurious interrupt will occur. The probability of a spurious interrupt goes up as the number of frames/second increases, or as the number of "buffers used"/second increases, or as the frequency of CTS* and CD* line changes increase. In applications with short frames and high speed (100s of Kbps) links, a spurious interrupt could occur as often as once every 10 minutes.

What to do with this information:

Option 1) Do Nothing
Spurious interrupts can occur for many legitimate reasons such as the same event occurring twice during the execution of the interrupt handler. Furthermore, events reported in the SCCE register are also indicated in the Tx BDs, the Rx BDs, or the SCCS register. Therefore, most interrupt handlers should already contain a provision for handling spurious interrupts, and should be well able to verify the validity (or lack of validity) of the interrupt through the Tx BDs, Rx BDs, or the SCCS.

Option 2) Check the SCCE bit After Clearing It
The M68000 core can check the bit after clearing it, to make sure that it has been cleared. If not, it can clear it again. This will eliminate any spurious interrupt of this type. This could be implemented as:

```
BACK:       MOVE.B      #$01,SCCE
            BTST        #0,SCCE
            BNE         BACK
```

Option 3) Write the Bit Twice
It can be possible to fully eliminate this type of spurious interrupt by simply writing the bit twice-in-a-row during the interrupt handler.

```
            MOVE.B      #$01,SCCE
            MOVE.B      #$01,SCCE
```

In order to guarantee this, the user must assure that, (1) no interrupt will occur between the two instructions shown above, and (2) that neither the IDMA nor an external bus master will take the bus between these instructions. The possibility of one SDMA access between the above instructions is assumed in the above solution. Even if these conditions cannot be met, this may be a good compromise, since it will tremendously reduce the probability of this type of spurious interrupt. Fixed in REV C. No impact on user software/hardware will occur. The workaround software may still be used on REV C devices with no adverse effects.

2. UART Echo Mode

The automatic echo mode configuration in the SCM of the UART mode register does not work properly. Only about 95% of the UART character bits are re-transmitted correctly, due to a synchronization problem.

Workaround: echo the characters in software.

Fixed in REV C. No impact on user software/hardware will occur, except that the feature will echo characters properly 100% of the time, if this feature is invoked.

3. Chip Selects Read-Only and System Configuration Registers

When one of the chip selects overlaps the addresses of the system configuration registers ($F0-FF), and is programmed to be read-only, a write operation to a system configuration register will cause a bus error (BERF asserted by the MC68302).

The workaround is simply to program the chip select to be read-write before writing one of the system configura registers, and changing it back to read-only after the operation is complete.

Fixed in REV C. No impact on user software/hardware will occur. The workaround software may be used on RE devices with no adverse effects.

4. D-Channel Collision

When the MC68302 is in HDLC mode with the automatic retransmit enable (RTE) bit set in the HDLC mode regi and this SCC is supporting the D-channel collision mechanism of the IDL mode, the re-transmission cannot place immediately, when the collision is smaller than 36 bits. The MC68302 waits 2 IDL frames (4 D channel bits) DGRANT goes high before it retransmits.

Improved in REV C. No impact on user's software/hardware will occur.

5. Interrupt Vectors in Slave Mode

Summary:

If the MC68302 is used in slave mode, AND the external master internal asynchronous read/write interface tim are used (SAM bit = 0), AND the MC68302 is instructed to issue vectors during interrupt acknowledge cycles (bit = 1), AND the external master is operating at a frequency that is truly asynchronous from the MC68302 external 68020 at 25 MHz, and 68302 at 16.67 MHz), it is possible after some amount of time (minutes, hours days) for the wrong vector or the error vector to be issued to the external master during an interrupt acknowle cycle.

Workarounds:

Option 1) Eliminate One of the Possible Causes
The user may implement a workaround by eliminating one of the conditions listed above, that can ca the bug.

Option 2) Hardware Workaround
It is also possible to fix the problem in hardware. Simply synchronize the falling edge of the AS* i signal generated to the MC68302 during an interrupt acknowledge cycle to the MC68302 CLKO sig In effect, the AS* input of the MC68302 is not longer asynchronous during the interrupt acknowle cycle. The problem will be eliminated if the falling edge of the MC68302 AS* input signal occurs w the MC68302 CLKO signal is in the high state. AS* should be stable while CLKO is low.

6. Very Short Frames in Synchronous Protocols

Summary:

If a MC68302 SCC is used in a synchronous protocol mode (HDLC, BISYNC, Transparent, or DDCMP), **AND** transmit frame is stored in just one data buffer, **AND** the transmit frame is only 1 or 2 bytes in total length (i.e. 1 bytes stored in the data buffer) **AND** more than one MC68302 SCC is operating simultaneously, it is possible some amount of time (minutes, hours, or days) for a frame to be reported as being transmitted successfully (i.e. Tx BD shows that it has been transmitted without errors), but in reality, the frame was not transmitted over the pin.

Workarounds:

Option 1) In the case of transmitting a two byte frame, simply split the frame into 2 one-byte buffers. This eliminate the problem, and will not affect serial performance.

Option 2) Do nothing. Since the problem occurs very rarely, wait for the missing frame to be detected by the hi layers of the software, and a retransmission requested.

January 26, 1996

7. BISYNC

Summary:

When external microcode is downloaded into the RAM area of the 68302 (not necessarily operating), **AND** the BISYNC protocol is operating on any channel **AND** { the Hunt Mode bit (H) is set in the BISYNC Co Character Table **OR** a receiver overrun status bit (OV) is set in the Receive Buffer Descriptor **OR** a carrier de signal status bit (CD) is set in the Receive Buffer Descriptor}, there may be incorrect operation of the BIS protocol.

Workarounds:

Option 1) Eliminate one of the first two essential causes :

The user may implement a workaround by eliminating one of the first two essential conditions li above, that can cause the bug.

Option 2) Eliminate all of the conditions in the third cause :

(i) Do not set the Hunt mode (H) bit in the BISYNC Control Character Table, instead put the channel Hunt mode by host command (Enter Hunt);

(ii) ensure the margins of operation are such that a receiver overrun can never occur; and

(iii) permanently connect the SCC's CD* input line to a logic LO to prevent loss of Carrier Detect. The status of CD may be detected by software using a port connected to the peripheral's CD line.