Mask Set Errata 1
MC68HC912BC32 Microcontroller Unit

INTRODUCTION

This errata provides mask-set specific information applicable to the following MC68HC912BC32 MCU mask set devices:

- 3J15G

MCU DEVICE MASK SET IDENTIFICATION

The mask set is identified by a four-character code consisting of a letter, two numerical digits, and a letter, for example F74B. Slight variations to the mask set identification code may result in an optional numerical digit preceding the standard four-character code, for example 0F74B.

MCU DEVICE DATE CODES

Device markings indicate the week of manufacture and the mask set used. The data is coded as four numerical digits where the first two digits indicate the year and the last two digits indicate the work week. The date code “9115” would indicate the 15th week of the year 1991.

MCU DEVICE PART NUMBER PREFIXES

Some MCU samples and devices are marked with an SC, PC, ZC or XC prefix. An SC, PC or ZC prefix denotes special/custom device. An XC prefix denotes device is tested but is not fully characterized or qualified over the full range of normal manufacturing process variations. After full characterization and qualification, devices will be marked with the MC prefix.

When contacting a Motorola representative for assistance, please have the MCU device mask set and date code information available.

MSE Published Date: March 1, 2001

FLASH: FASTER WAIT MODE WAKEUP RECOVERY TIME FOR FLASH AR457

Flash requires 250nsec delay for wakeup from wait mode with FEESWAI=1. If the operating bus frequency is greater than 4MHz, the Flash cannot be used when recovering from WAIT mode when the FEESWAI bit is equal to ‘1’.

Work around If your interrupt vectors are located in the Flash array, do not set the FEESWAI bit in Wait mode.

ATD: CONVERSION OF THE (V_{RH}–V_{RL})/2 INTERNAL REF VOLTAGE RETURNS $7F, $80 OR $81 AR311

The (V_{RH}–V_{RL})/2 internal reference conversion result may be $7F, $80 or $81.

Work-around If the (V_{RH}–V_{RL})/2 internal reference is used (perhaps for system diagnostics), expected pass result may be $7F, $80 or $81.

BKP: ADDRESS AND DATA REGISTERS RESET ONLY ON POR AR463

Breakpoint Address and Data registers are properly reset only upon Power on Reset.

Work around To ensure BRKAH, BRKAL, BRKDH and BRKDL registers have the correct default values, always clear each immediately after reset.

INT: EDGE SENSITIVE IRQ DOES NOT WORK CORRECTLY DURING STOP AR528

When using an edge sensitive IRQ signal to trigger an interrupt service routine and the IRQ is not released during servicing, the part will not enter stop mode if the following executed instruction is STOP.

Work-around When IRQ is set to be edge sensitive, release pin before executing STOP instruction.

INT: WAIT CANNOT BE EXITED IF XIRQ/IRQ LEVEL DEASSERTION OCCURS WITHIN PARTICULAR WINDOW OF TIME AR600

The device can get trapped in WAIT mode if, on exiting the WAIT instruction, the deassertion timing of the XIRQ or level-sensitive IRQ occurs within a particular timeframe. Only reset will allow recovery. Noise/bounce on the pins could also cause this problem.

Work-around

1. Use edge-triggered IRQ (IRQE=1) instead of XIRQ or level-triggered IRQ.
2. Use RTI , timer interrupts, KWU or other interrupts (except level-sensitive IRQ or XIRQ) to exit WAIT. If using RTI, it must be enabled in WAIT (RSWAI=0) and the COP must be disabled (CME=0).
3. Assert XIRQ or level-sensitive IRQ until the interrupt subroutine is entered.
4. Add de-bouncing logic to prevent inadvertent highs when exiting WAIT.

INTERRUPTS: ARE NOT MASKED DURING 1ST CYCLE OF ANY BACKGROUND INSTRUCTION AR441

There is a cycle at the beginning of executing a BDM instruction (tag, trace, instruction) that is susceptible to being interrupted directly after the background has executed. Since the interrupt source is masked, the interrupt vector that is requested is $FFF6. The BDM firmware at $FFF6 points to the routine at $FF24. The interrupt was stacked when it was taken, but the BDM routines do not un-stack (no RTI). When you return from BDM, the stack pointer is pointing to the wrong place.

Work around

This problem may occur when using the BDM. Avoid using TRACE command during cycles where interrupts become unmasked (i.e. TRACE of a CLI if interrupts are pending). Do not use the BGND instruction directly following an instruction that clears the X or I mask bits. Do not try to TAG the opcode directly following an instruction that clears the X or I mask bits if interrupts are pending. There is still the scenario where an interrupt occurs exactly during the execution of the 1st cycle of any BDM execution, which will still cause the problem.

INT: DISABLING INTERRUPT WITH I MASK BIT CLEAR CAN CAUSE SWI AR527

If the source of an interrupt is taken away by disabling the interrupt without setting the I mask bit in the CCR, an SWI interrupt may be fetched instead of the vector for the interrupt source that was disabled.

Work-around Before disabling an interrupt using a local interrupt control bit, set the I mask bit in the CCR.

IOB: AR510

Expanded mode operation causes an increase in bus driver shoot through current pin leakage which can cause inaccurate A/D conversions.

Work around Enabling reduced drive on Ports A, B and E while the device is in expanded mode reduces the amoun t of VDD/VSS noise caused by bus driver shoot through current. Therefore, the A/D accuracy meets specified limits.

MSCAN12: ADDITIONAL TX BIT WHEN GOING BUS-OFF AR435

When a CAN transmission takes place and an error occurs, one extra dominant bit will be sent when going off bus. The problem will be corrected in a future revision.

Work around None.

MSCAN12: INCORRECT RX MESSAGE IN OVERRUN AR434

There is a chance that the Rx buffer may contain a corrupted message in Overrun. The last received message which triggered the Overrun condition will be a shifted image in the Rx buffer, if the RXF was cleared during reception.When starting to transmit a message in an almost Overrun condition (FG and BG buffers are filled) and the SW clears RXF during the arbitration field and arbitration is lost, the Rx buffer will also contain a corrupted (shifted) message.

Work around To prevent this condition, the receive drivers must process the incoming stream of messages at a fast enough rate to never have both buffers filled at the same time. The msCAN receive interrupt should be elevated to the highest priority.

MSCAN12: RECOVERY FROM SLEEP MODE AR450

When entering sleep mode, RXF is not changed, however the foreground/background pointer is reset, resulting in buffer Rx1 in the foreground readable by the CPU. If there was one message in Rx0 (and none in Rx1) the pointer will now point to the wrong buffer with undefined contents.

Work around The receive buffers must be cleared by the CPU before entering sleep mode.

MSCAN12: TERRIF IS SET AT END OF BUS OFF AR451

When the msCAN is in bus-off state and has counted 128 sequences of 11 recessive bits, the TERRIF flags is set.

Work around Always clear TERRIF together with BOFFIF.

ROC: CPU STOPS GOING INTO WAIT BEFORE STRETCHED CYCLE IS FINISHED AR472

When the device exits WAIT mode, it does not return to the correct location within the routine if the stack is positioned in external memory and if the stretch bits have been enabled to lengthen the clock.

Work around To overcome this problem, locate the stack in internal RAM resources and/or clear the stretch bits to prevent clock stretching.

CGM: CRYSTAL START-UP WITHOUT DELAY AT CLOCK MONITOR RESET AR504

When a clock monitor reset occurs, the crystal may start-up with no delay period. As a result, the MCU attempts to fetch reset vectors before the crystal has fully stabilized.

Work-around

When clock monitor failure has occurred, hold the reset signal until the crystal has reached stable oscillation.

In some applications the clock monitor reset is used to detect an accidental crystal clock loss. An alternative solution is to engage the PLL and enable the limp-home mode. As the limp-home mode is enabled, the clock monitor will not reset the MCU (NOLHM =0 in PLLCR). In case of clock loss, the limp-home mode will be engaged.

If the crystal clock is restored, the PLL will be automatically re-engaged. As the transition from PLL to limp-home and vice versa is smooth, the CPU will continue to run the code even if the crystal oscillation amplitude is not fully stabilized.

The LHIF flag in PLLFLG is set when MCU enters or exits the limp-home condition.

CGM: CANNOT INTERRUPT OUT OF STOP WITH DLY=1 AR562

STOP mode cannot be exited using interrupts when DLY=1 depending on where the Real-Time-Interrupt (RTI) counter is when the STOP instruction is executed. The RTI counter is free-running during normal operation and is only reset at the beginning of Reset, during Power-on-Reset, and after entry into STOP. The free-running counter will generate a one cycle pulse every 4096 cycles. If that pulse occurs at the exact same time that the stop signal from the CPU is asserted then the OSC is stopped but the internal stop signal will remain low. In this state the OSC is shut off until RESET.

Work-around

1. If you are not using the Real Time Interrupt function you can wait for a RTI flag before entering into STOP to guarantee the counter is in a safe state. When executing the following code all interrupt sources except for those used to exit STOP mode must be masked to prevent a loss of synchronization.
   A loss of synchronization can occur if an interrupt is processed between the setting of the RTIF and the execution of the STOP instruction. Also, you must enable the RTI counter in the initialization code, set to the fastest RTI time-out period, and the RTIE bit should NOT be set.

```
        BRCLRRTIFLG,#RTIF,RTIFClr; RTIF flag is already clear
        LDAB#RTIF; if it's set, clear the flag.
        STABRTIFLG
RTIFClr:BRCLRRTIFLG,#RTIF,*; wait until the RTIFLG is set.
        NOP
        NOP
        NOP
        STOP   ; enter stop mode
```

2. If you are driving a clock in (not using the OSC) then you could set DLY=0.
3. Pseudo-STOP and DLY=0 could be used. The oscillator will be kept alive during STOP at the expense of power consumption but no recovery delay is needed. Set the PSTP bit and clear DLY bit prior to going to STOP.
4. Limp Home and DLY=0 could be used. The part comes out of STOP in limp home mode while the crystal recovers. If a known frequency is not a requirement, this workaround avoids having the crystal alive in STOP mode. Clear the NOLHM and DLY bits prior to going to STOP.

ROC: STOP DOES NOT TAKE TIMEOUT WHEN RESET IS THE RELEASE AR400

When coming out of STOP by using reset, the crystal start-up delay time- out does not occur. This means the MCU may be attempting to run before the crystal has become stable.

Work around

When coming out of STOP with reset, hold the reset signal until the crystal has reached stable oscillation.

March 1, 2001

Additional mask set errata can be found on the World Wide Web at http://www.mcu.motsps.com/documentation/index.html