# AN1263

# Designing for Electromagnetic Compatibility with Single-Chip Microcontrollers

**By Dugald Campbell**
**CSIC Applications Engineering**
**Aylesbury, England**

## INTRODUCTION

This application note discusses how to design a single-chip microcontroller application considering electromagnetic compatibility (EMC). Today almost every consumer, automotive, and industrial application has a microcontroller (MCU) inside. More often than not, it will be a low-cost, single-chip MCU. Single-chip MCUs are ideal because of the flexibility and functionality incorporated on one piece of silicon. Typical MCUs have their own CPU, RAM, ROM, and input/output (I/O) ports and can have customized functions such as analog/digital modules, LCD drivers, on-screen display for television applications, dual-tone multifrequency (DTMF) generators for telephones, AC motor drive circuits, and EEPROM for non-volatile data storage.

As MCU functionality increases becoming more complex and with market costs being driven lower, MCU producers must reduce their manufacturing costs continually. Reducing the geometries of the on-chip transistors and gates achieves this, and also helps produce MCUs capable of functioning at higher operating frequencies.

As a transistor's gate size is reduced, the transition time decreases, and, according to Fourier Analysis, fast edges on signals produce harmonic signals. These signals, if amplified, can cause emission problems. In a similar vein, if the devices have faster transition times, they can react to faster incoming signals, which can result in a gate being switched because of a high frequency noise spike and a false signal. Most modern MCUs operate with speeds ranging from 2 MHz to 40 MHz, with internal devices having switching speeds from a few nanoseconds to below a nanosecond, making them potential EMC problems.

## EMC — ELECTROMAGNETIC COMPATIBILITY

Because many electronic circuits are in proximity to each other, it is essential that their design is not affected by external noise sources and that the circuit itself is not a noise source affecting other circuits. This relationship is known as electromagnetic compatibility (EMC).

An EMC problem will occur in one of two categories: emissions or susceptibility. Both emissions and susceptibility will consist of a conductive (via electrical connection) or radiated nature. For example, Figure 1 shows an office photocopier emitting radio frequency (RF) signals to the outside world that cause the monitor's display to fail. This is a radiated EMC problem. Similarly, the photocopier could radiate noise down the power mains which would in turn conduct onto the computer monitor power mains and thereby cause the fault in a conductive manner. The monitor could be considered a victim of the photocopier because of poor design and high RF noise radiation. Arguably, the monitor could be designed poorly and be unable to handle background environment noise existing in electronic office equipment.

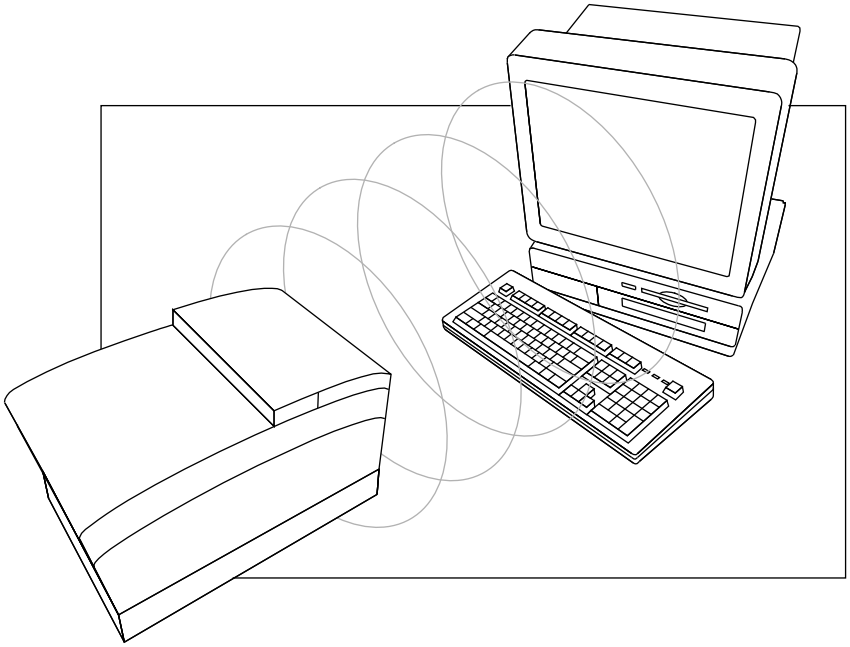

**Figure 1. Electromagnetic Compatibility Scenario**

# IDENTIFYING AN EMC PROBLEM

Finding an EMC problem when a product is in early or full production can be a very costly experience and could possibly result in a complete redesign of the printed circuit board (PCB). Consequently, EMC should be considered during the initial stages of the design. However, regardless at what stage an EMC problem is discovered, the first question that should be asked is:

**“Is the EMC problem due to emissions or susceptibility?”**

The questions that logically follow are: “Is the MCU radiating noise that is upsetting other circuits, or is noise being generated externally causing the MCU to act abnormally?”

When solving susceptibility problems, the changes made to improve performance frequently also will help improve the product’s emissions performance. Conversely, for changes made to improve emissions will help improve susceptibility as well.

## EMISSIONS

Emissions from any digital source are normally high frequency, coming from the harmonics produced by the high-frequency clock rates that make up a square wave. These emissions generally are radiated and created by the currents switching between the source and return paths of the digital signals and on the main power supply to the digital source. These fast switching currents flow in loops acting as small antennas, which radiate magnetic fields known as differential-mode radiation. Also, any resistive or inductive paths can cause voltage drops in the circuit, which will put some parts of the circuit at a common-mode potential relative to true ground.

If an emissions problem exists with single-chip MCU applications, the PCB design most likely will be the major contributing factor, as loop areas within the silicon are several magnitudes smaller than those created on a PCB. Emissions are likely to come from the power supply (GND and $V_{DD}$), since the sum of all currents in the digital chip pass through here. Port pins are designed with higher drive capability than internal gates and, therefore, generally are larger in size, resulting in increased capacitance ensuring that the transitions of the digital signal are slower than the transition times of internal gates. Since port pins are controlled normally by software, they will switch at lower frequencies than those of internal MCU circuits, meaning that port pins are not usually problematic areas in emissions. The exception is when the port pins drive or source high currents, where the magnetic field strength is proportional to the switching current and also proportional to the loop antenna area. The solution is either to reduce the current or reduce the loop area. In most cases, the loop area is altered.

The fastest fundamental external frequency to the MCU is the oscillator circuit. This is normally a signal which is nearly sinusoidal, if a crystal-based circuit is used. Because the oscillator circuit normally is very small and sinusoidal, the higher order harmonics are attenuated greatly and, consequently, should not upset other circuits. Again, if the clock comes from an external source, take care that the path for this signal is kept as close as possible to the return ground path/plane to reduce the loop area to a minimum.

For MCUs with external memory peripherals and even microprocessors, emissions can be more problematic. For instance, the main noise source is the clock/strobe that enables the external memory, which is switching very fast, has been routed without thought, and has a very long ground return path. This ensures that the clock signal acts as a strong loop antenna which radiates the clock and its higher harmonic signals. The lower address lines and the data bus also will be problematic if not laid out correctly, but reducing the current loop as described here will help significantly.

## SUSCEPTIBILITY

Most MCUs are designed using CMOS technology and work on the principle of latches and flip-flops to generate more complex functions. Because of their synchronous nature, clocking a bad level can easily cause malfunction. Any CMOS device will have a noise threshold that if exceeded will cause failure.

The failure usually will occur in these four categories:

1. MCU fails momentarily, then corrects itself.
2. MCU fails and interrupt or reset recovers the MCU.
3. MCU fails. Powering OFF then ON recovers the failure.
4. MCU fails and latch-up occurs, resulting in permanent damage.

If the failure occurs in categories 1 and 2, then it may go unnoticed and the end user may either never see the failure or accept it due to its irregular occurrence.

However, if the failure is in categories 3 or 4, then it definitely will be seen as an immunity problem and will be unacceptable by any manufacturer. A category 4 failure will need to happen in the field for the first time, since the product release would have been halted had the failure happened during the design phase. This failure would have gone undetected only if no EMC testing had been carried out in the product's design phase.

# SOLVING THE EMC PROBLEM

EMC performance can be improved on all MCU applications by giving extra attention to three design areas:

1. PCB layout
2. Watchdog circuit
3. Defensive programming

## PCB Layout

PCB layout is the single most common cause of an immunity failure in an MCU. With care and attention to the three most critical areas of an MCU, EMC immunity/emission performance can be improved. The critical areas are:

1. Power supply routing
2. Oscillator circuit
3. Input pin termination

### *Power Supply*

Any noise induced on the power supply undoubtedly can cause malfunction. If transients are imposed on the $V_{DD}$ or $V_{SS}$ line, then internal circuits such as flip-flops and inverters can change their state and result in malfunction. Equally, if large, fast-switching currents are present, then higher emissions will occur. As a general rule, all unwanted high-frequency signals must be attenuated by ensuring low impedance paths to ground. Since all signals have currents that flow through $V_{SS}$ and $V_{DD}$, signal lines should be kept as close as possible to the $V_{DD}$ and $V_{SS}$ to reduce the current loop area.

Using a multi-layer PCB with power planes connected to $V_{SS}$ and/or $V_{DD}$ will always give better results, since this always reduces the loop area of all current paths to the minimum and decreases mutual inductance between signal lines. In low-cost applications, normal multi-layer PCB is the first cost reduction, with a two- or one-sided PCB generally used. For this reason, to help combat EMC, more attention is needed on other parts of the PCB design.

Ensuring that the $V_{SS}$ and $V_{DD}$ tracks are as wide as possible and that decoupling is as close to the MCU as possible will reduce susceptibility. Thin tracks are high impedance with high-frequency currents, meaning potential differences will develop across them.

For fast-switching currents – depending on how the DC supply to the device is generated – improvements can be made to increase performance by ensuring that either the $V_{SS}$ or $V_{DD}$ is the low-impedance path to earth.

The most popular way to generate DC power from the AC power mains is illustrated in Figure 2. This shows that the AC power mains are stepped down to a smaller AC signal and fed through a bridge rectifier to produce a DC voltage, then either filtered or smoothed, and followed by a voltage regulator. In this case, any metal enclosures are connected to the ground line.

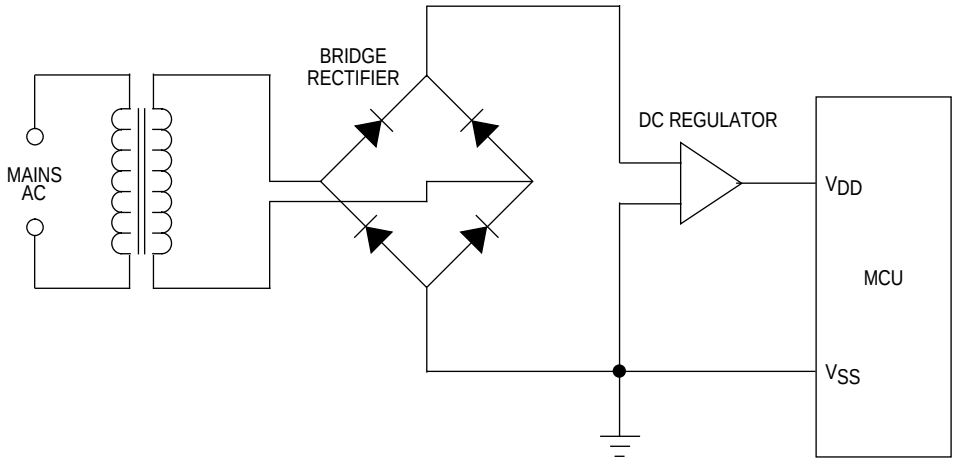


**Figure 2. Typical Method of Providing DC Power from AC Power Mains**

High-frequency signals will regard $V_{DD}$ and $V_{SS}$ as the same potential because of decoupling capacitance. In Figure 2, the low-impedance path to earth for switching currents is generally the $V_{SS}$ path.

## NOTE

If poor connections are made between enclosures, then higher impedance ground may result and cause higher differential voltages. This in turn will give poorer results.

For this circuit, any high-frequency signal lines should be decoupled to the $V_{SS}$ to avoid crosstalk and emissions.

In some low-cost applications, the DC power is generated slightly differently. An alternative circuit is shown in Figure 3 where a high-voltage diode and capacitor are connected across the power mains, giving a half wave rectifier. The ripple voltage is then smoothed out by a smoothing capacitor, and a zener diode connected to the power mains, where the anode becomes a floating ground.

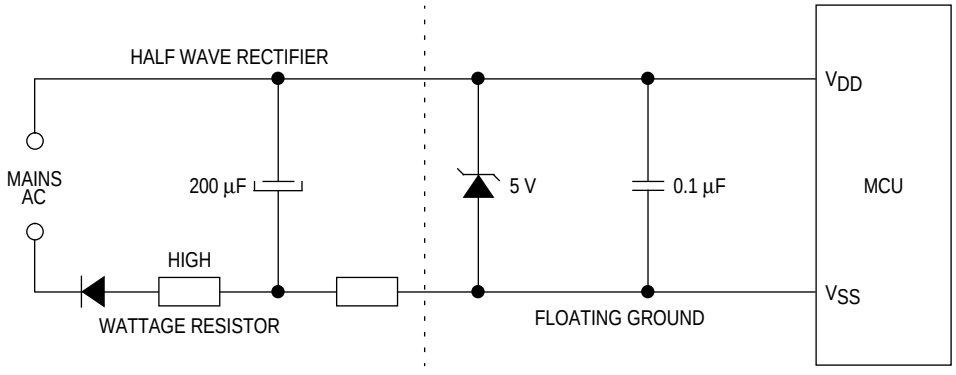


**Figure 3. Low-Cost Method of Providing DC power from AC Power Mains**

The low-impedance path to earth is now the $V_{DD}$ line via the live power mains. In this case, decoupling high-frequency signal lines to $V_{SS}$ may not help an MCU's EMC performance. Rather, for this type of circuit, decoupling high-frequency signal lines to $V_{DD}$ is more desirable.

A common failure is inadequate decoupling on the power supplies. Connecting the standard 0.1 µF capacitor between the power supplies is normal. This capacitor is good for low-frequency noise and ensures a good steady current for the MCU for the high transient currents (reservoir capacitor) generated from CMOS technology. But if the MCU is having emissions/immunity problems at frequencies of 40 MHz and beyond (as with electrostatic discharge (ESD) and transients on power lines), then the 0.1 µF probably will have no great effect in reducing the attenuation. Because all capacitors have an internal inductance and lead inductance even with SOIC-type packaging, a typical decoupling capacitor will have an inductance of around 5 nH. This means that the parallel resonance will occur at approximately 7 MHz, where it will become the major component at higher frequencies after the inductive reactance. This is where implementing a value of 1 to 10 nF as close as possible to the power supply lines can give good results (resonance will be greater than 20 MHz). When designing single-layer PCB, choosing the correct decoupling capacitor can have significant effects on its EMC performance.

### *Oscillator*

The oscillator circuit (see Figure 4) consists of an inverter inside the MCU that is connected to a quartz-crystal or ceramic resonator. Also attached are a feedback resistor and some small decoupling capacitors to reduce the harmonics produced by the piezo-electric device.

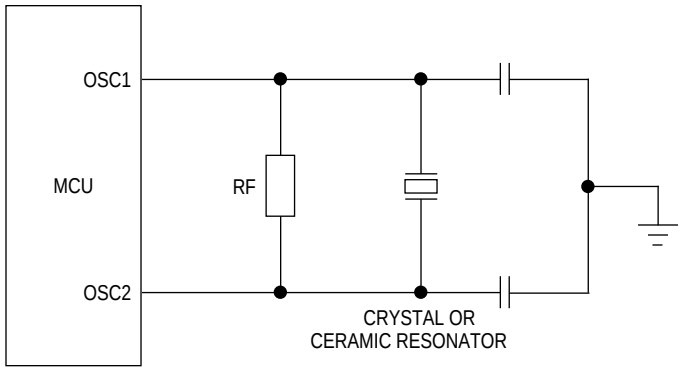


**Figure 4. Typical Oscillator Circuit**

The inverter within the MCU has a high-input (OSC1) impedance as well as a high-output (OSC2) impedance. The output has a low drive capability so as not to overdrive and damage the crystal. The oscillator output also goes to another buffer then to a prescaler. The prescaler divides the oscillator clock by two, which ensures a stable 50% duty clock for the internal CPU clock.

The CPU uses the high transition to enable address set-up and the falling edge to capture data. The CPU designed to normally work at 2 MHz can be processed to execute properly at 4 MHz. The speed restriction of the MCU is governed by the CPU functionality, where internal propagation delays must be met.

If any noise is induced on the oscillator circuit, then CPU failure can occur. The reason for this is diagrammatically shown in Figure 5. Figure 5 (A) shows the expected oscillator clock where the divider ensures 50% duty cycle for the CPU, from any outside jitter. Figure 5 (B) shows the effects of having some noise spikes applied to the oscillator circuit, which results in the CPU clock not being 50% duty. This means that the CPU may not correctly set up the address for the program counter or not fetch the correct data. As a result, the CPU runs away. If the noise spike is a once-only event, then this may be recovered normally with a watchdog circuit that performs a reset (see **WATCHDOG CIRCUITS**).

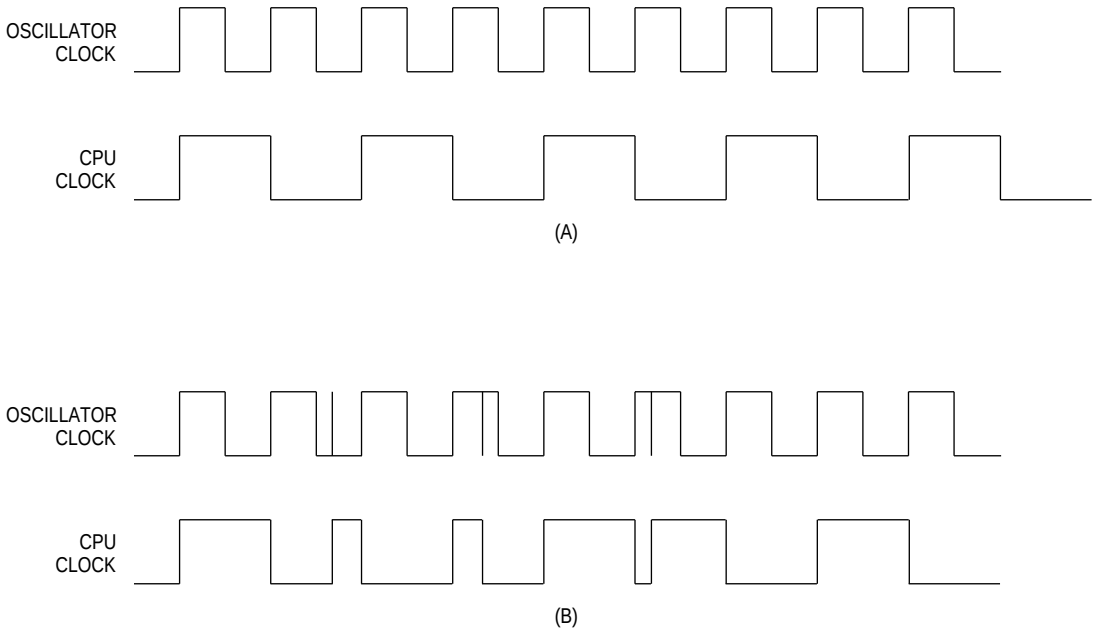


**Figure 5. Effects of Spikes on an Oscillator Circuit**

Ensure that the oscillator is designed with the cleanest possible environment. This can be achieved by:

1. Ensuring the crystal and other oscillator components are connected as close as possible to the OSC1 and OSC2 pins. Avoid having long PCB lines which act as antennas to high frequency signals that may be present in the application's surroundings.
2. If possible, enclosing the oscillator circuit with a $V_{SS}$ guard ring which will reduce the loop area.

**NOTE**

Even though the guard ring track is thin due to mechanical restriction, it is still better than having no guard ring at all.

3. Decoupling the crystal circuit to the lowest impedance path to earth, normally $V_{SS}$ (see points given under ***Power Supply***).
4. Keeping any fast switching signals away from the oscillator lines.

Refer to Figure 6 for a typical oscillator PCB layout.

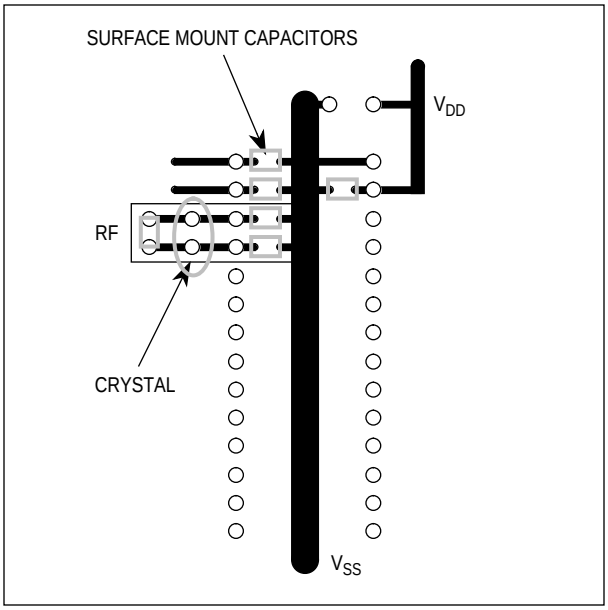


**Figure 6. Typical Oscillator PCB Layout for MC68HC05P3**

***Input Pins***

Protecting input pins is also critical, especially the RESET and IRQ pins. A higher $I_{DD}$ will normally be the tell-tale sign that shows an input pin has been left unterminated. This is caused by the effect of n-channel and p-channel transistor gates, if unterminated, floating to half way between the $V_{SS}$ and $V_{DD}$ value. At this stage, both negative and positive devices are ON and represent a resistance between $V_{SS}$ and $V_{DD}$. Therefore, current flows through both transistors. After the input buffer, it is expected that one level will take precedence and ensure all other consequent gates are in a static state (one or zero). But any noise that is radiating in the vicinity can create small fluctuations that can cause the buffer to switch its output between both rails, resulting in the consequent gates being switched and increasing the power consumption even more.

If the switching is of a high enough frequency, then crosstalk between other circuits in the MCU can cause malfunction. As a general rule, all input pins that may be located near high-frequency signals should have a hardware low-pass filter attached or some small decoupling at a minimum.

It is unlikely that the RESET pin would be left unterminated, since the MCU would not function correctly in that case. What is most likely is that the PCB track would come from another device or circuit that is not close to the pin. And although schematically terminated by a pullup resistor, the line is seen as an antenna at high frequencies and may pick up enough energy to give a small pulse which will cause an unknown or partial reset to the MCU. Specifications state that the RESET pulse must be longer than two CPU clock cycles to ensure that a correct reset has occurred.

Decoupling these pins to the low-impedance return path, again as close as possible to the pins, will attenuate any high-frequency noise and give an increase in immunity performance (typically 1 to 10 nF).

Another plus point is to try to reduce the amount of edge-triggered interrupts. If interrupts can be triggered only with a level, then this will ensure that noise on an interrupt pin will not cause an undesired operation. If this is not possible, then on immediate entry of an edge-triggered interrupt, a software check on the pin to determine if the level is correct should suffice.

## WATCHDOG CIRCUITS

A watchdog circuit generally is described as a piece of hardware that must be updated at a regular interval by the MCU or the circuit will cause the MCU to reset. An external watchdog circuit is the best, as it is independent of the MCU, and, if the MCU fails to update the watchdog, then the circuit will give a reset to the MCU. A watchdog normally consists of a hardware timer which will time out if no update or report is made from the MCU.

A simple watchdog circuit is shown in Figure 8.

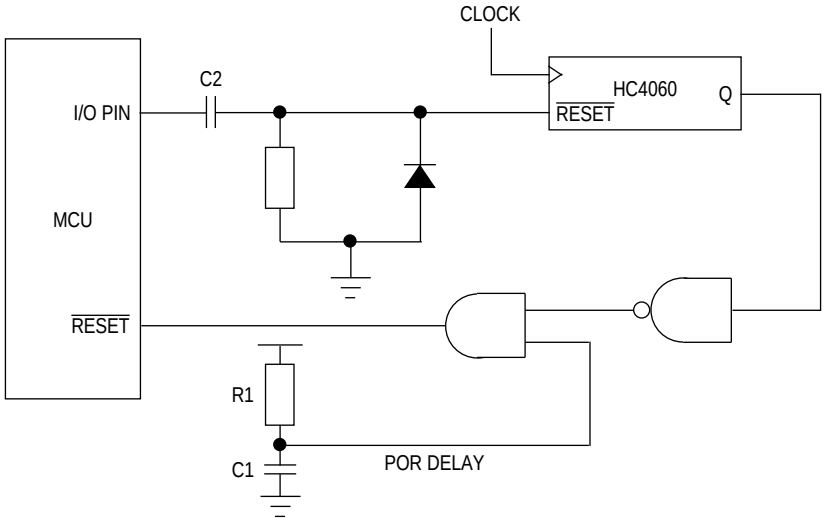


**Figure 7. External Watchdog Circuit**

The MCU is held in its reset condition with the power-on reset (POR) delay on power-up. The clock may come from an external source or be a tap from the crystal oscillator used on the MCU. While the MCU is in reset condition, the I/O pin is an input and the counter will be running. If Q is zero and C1 is charged up, the MCU will come out of reset.

The MCU's first task is to give a small pulse which will reset the counter to a known time-out value. The I/O port from the MCU is AC coupled to ensure that only an edge will re-trigger (kick) the counter. This ensures a constant DC level will not disable the MCU or watchdog. The MCU must send a small pulse before the Q output goes to a one to ensure that the MCU is not reset. Normally, at least two software instructions are needed from the MCU.

The next best alternative is for the MCU to have its own watchdog circuit on-chip (see Figure 8). For example, Motorola’s MC68HC705C8 has its own watchdog consisting of a programmable timer that must be refreshed approximately every 32,768 clock cycles or it will generate an internal reset.

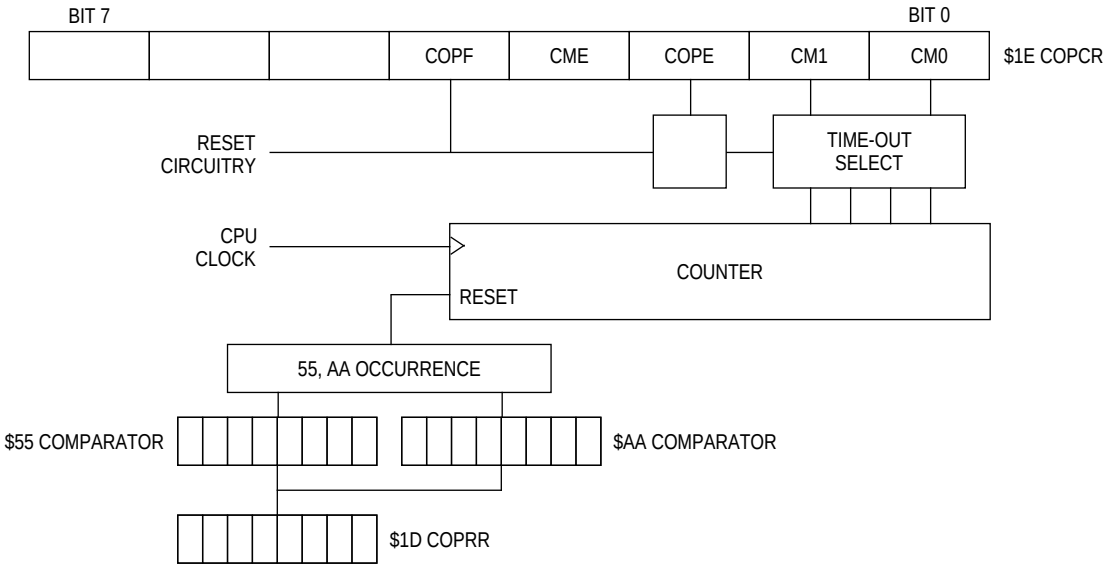


**Figure 8. On-Chip Watchdog Circuit**

The CM1/CM0 bits are used to select a time-out period (for example, CPU clock divide by $2^{15}$,$2^{17}$,$2^{19}$,$2^{21}$). The COPE bit is write-once only, and after the watchdog has been enabled, it cannot be disabled except by a reset. The COPF bit is a flag to inform the user that the reset that occurred was caused by the watchdog timing out or otherwise. The sequence to refresh the watchdog is two writes on the COP reset register (COPRR). The sequence is a write of $55 followed by a write of $AA to the COPRR.

Not all MCUs have the luxury of an on-board watchdog circuit, but in some cases the software designer can implement his own watchdog with an unused interrupt. For example, an MC68HC05P1 has no watchdog but has a 16-bit free-running timer. This timer can generate an interrupt in three ways:

1. A timer overflow FFFF to 0000
2. An output compare, 16-bit register is compared to the timer counter value
3. An input capture of a rising or falling edge on a TCAP port pin

If, for example, the output compare interrupt was not being implemented, it could be used as a pseudo watchdog. But in this case, the timer would make a compare with the output compare registers, then check the system to see if all was OK. If the system was functioning correctly, then a return from interrupt would be executed. If there was an abnormality, then a RESET of the stack pointer and a jump to the initialization routine would be made. The system check could be a token passing routine as described in **DEFENSIVE SOFTWARE**.

Optimum placement of watchdog updates within the program is no simple task and generally will be the last procedure before finalizing the program. Normally, the user should examine the software flow and timing for all interrupt routines and subroutines, critical and non-critical applications. Ideally, one watchdog refresh in the whole program would be best, but because MCUs have large programs and more on-chip functionality, this is rarely achievable. If possible, the watchdog refresh routines should never be placed in interrupt or subroutines, but should be placed strategically in the main loop. Take care that the refresh rate is not too large or the chances of recovery from a runaway condition will be decreased.

## DEFENSIVE PROGRAMMING

By simple defensive programming, an MCU's immunity performance can be increased greatly and in some cases by an order of magnitude. The beauty of defensive programming is that it is inexpensive to implement and, if done correctly, can save hardware costs in PCB layout as well.

### *Refreshing Port Pins*

One of the simplest examples of defensive programming is continuously updating of input/output pins and important registers. In most MCU applications, the software will have a main loop which will be regularly executed. In MCU architecture, the I/O pins generally will be laid out near the bond pads, which are consequently near the edge of the die. When noise of a certain amplitude is subjected to a MCU, then it will propagate through to the silicon from the die edges inward. This means that logic at the edge of the die are most vulnerable to external noise sources such as the input/output circuits. So, by simply updating the data register and data direction registers at regular intervals, the threat of malfunction can be reduced.

### *Polling Inputs*

Another idea is to filter noise on input pins by reading the pin several times and taking the average result as the correct reading of that pin. A typical application of a keyboard is polled in software and then read again several times within a 10-ms period to ensure that a true keypress had taken place. This type of polling is known as debounce protection. For high-frequency polling, an example of HC05 code on a port pin is:

```
                    BSR         Read_RX_pin
                    execute on Carry Flag which will return with RX value.
                    ..... more s/w

Read_RX_pin:
not_a_one:          BRSET       RX, PortA, not_a_zero
                    BRSET       RX, PortA, not_a_zero
                    BRSET       RX, PortA, not_a_zero
* A logic zero has been read three times, so this is 0.
                    RTS

not_a_zero:         BRCLR       RX, PortA, not_a_one
                    BRCLR       RX, PortA, not_a_one
                    BRCLR       RX, PortA, not_a_one
* A logic one has been read three times, so this is 0.
                    RTS
```

This routine will read the port in quick succession. If three reads of the pin are the same, then the CPU will return with the carry flag at the state read, an inherent feature of the HCO5 CPU's BRSET and BRCLR. The problem with this routine is if the input pin is in a very noisy environment, it is possible that the CPU would continue in this routine for a long time, which may not be desirable in time-critical applications.

A better routine would be to read the input pin several times and increment a register (the Accumulator in this case). If the result is lower than a number, then accept it as a zero; and if the result is higher, then accept it as a one. This routine will always exit in a finite time, ensuring that all other functions of the application are not hampered.

```
Read_RX_pin:
                    CLRA                ;clear Accumulator
                    LDX         #$06    ;X register is loaded with number of reads
                    BRCLR       RX, PortA, no_inc
                    INCA
                    DECX
                    BNE         Read_again
                    RTS

**************** main ****************
                    JSR         Read_RX_pin
                    CMPA        #3
                    BLO         it_was_zero
                    BHI         it_was_one
```

### *Token Passing*

In critical tasks, token passing ensures that the task has been approached in a controlled manner rather than a runaway condition. A simple example of token passing (see Figure 9) uses an unused location in RAM. At certain time intervals in an application, the software could roll a one through a number of unused RAM locations. Before any critical task is executed, a check on the rolling ones results would be made. If the test passes, the task would be executed. For example, if the test failed, the MCU has approached this task from an unexpected jump, and a jump to the initialization routine would be made.

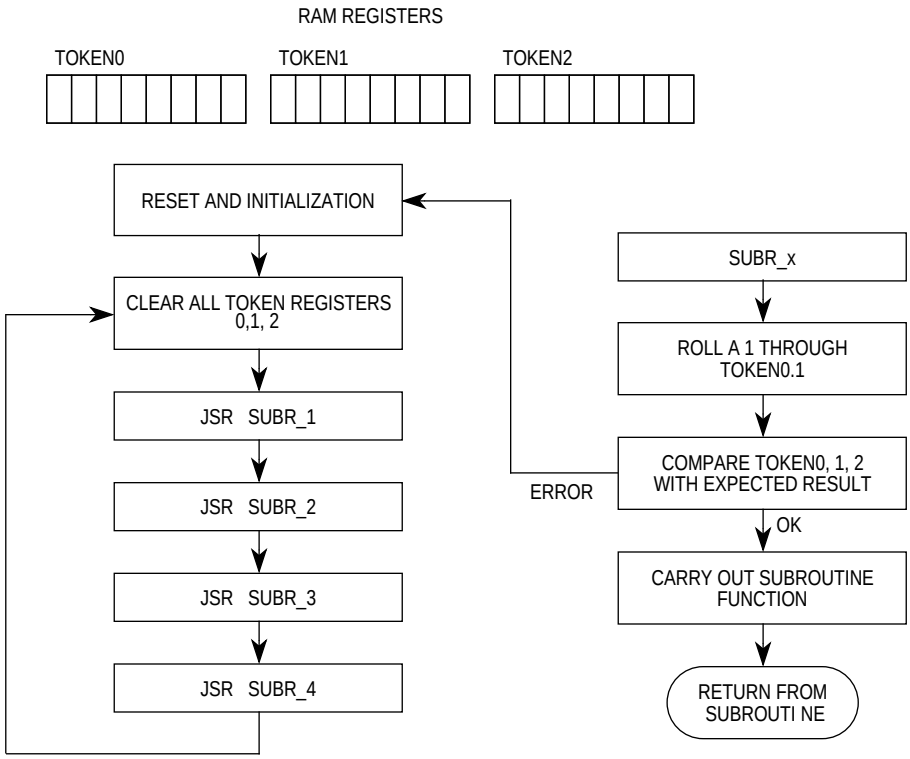


**Figure 9. Simple Example of Token Passing**

### *Unused Memory*

In most applications, some program memory will be unused. If in some cases the program counter is corrupted and jumps to unused memory, then some control on what should happen can be made. For example, a block of 500 bytes is unused out of 16 Kbytes of program space. If we fill the unused memory with the SWI instruction, then any non-intentional access to the unused area will result in the MCU fetching the SWI vectors, which would be programmed to jump to a known start position. When the SWI is needed for other functions in the MCU's application, fill the unused area (in an HC05 MCU) with the example here:

```
                ORG $3600
Unused_area:    9D          NOP
                9D          NOP
                CC 21 21    JMP Known_place
                9D          NOP
                9D          NOP
                CC 21 21    JMP Known_place
                etc.

                ORG $2121
Known_place:    CC 01 00    JMP Reset_routine
```

**NOTE**

The unused area is filled with 9D instruction and a conditional jump to a Known_place. When the MCU goes to this known place, it will then be forced to jump to the initialization routine. The reason for this is that if the MCU's program counter jumps to an instruction to address $3604, for instance, then it will fetch the instruction for $21, a BRN (branch never) instruction which the CPU will never make. The program will then execute the NOP on address $3606 and eventually pick up the jump to Known_place. In the case of an 8-K memory map with 500 bytes unused, the same code also will work if the address of Known_place was $0121. This happens because the CPU would mask off A13 as a zero on an 8-K ROM device.

## TESTING THE IMMUNITY PERFORMANCE

Today many EMC test centers will conduct on any electrical products the regulated immunity and emissions tests that are required by the countries where the products will be sold. Generally, these tests can be completed in a few days. If the product fails, however, increased expense and design time will be required that will delay getting the product to market. To avoid failing the EMC regulations, some inexpensive testing can be done in the lab environment. For MCU applications, two types of conductive testing give the best results. If an MCU application passes the conductive immunity tests, it is unlikely that the application would fail a radiated immunity test.

### *ESD Gun*

An electrostatic discharge (ESD) gun is a relatively inexpensive tool that will help find any high-frequency susceptibility in any digital circuit. Use of this gun is designed primarily to ensure that designs are not damaged with the high static voltages humans generate and discharge onto applications by contact. The ESD gun is available in battery-powered portable form and can be intelligently programmed to different voltage magnitudes. Because of its portability, the ESD gun can be used in many applications to test

immunity performance. The ESD gun normally provides the user with two types of tips: air-discharge and contact-discharge (see Figure 10). The air-discharge tip usually is spherical, whereas the contact-discharge tip is conical to a point.

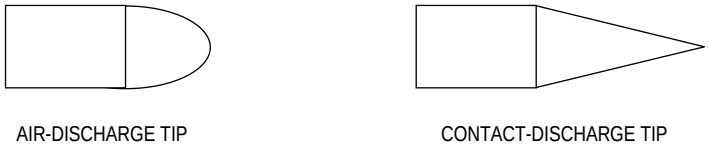


**Figure 10. Example of Air-Discharge and Contact-Discharge Tips**

Using contact discharge, the tip can be placed on a PCB track or device pin, and then the gun trigger can be pressed. Most ESD guns will limit the contact discharge to < 8 kV and allow single or multiple pulses to be applied by one trigger press. Using this contact discharge on the power supply lines and external connections will exhibit any design weaknesses. The normal process is to connect the tip to the track, and while the application is running, fire off random noise spikes initially at lower than 1 kV and increasing the voltage until failure occurs. Contact discharge also is better for repeatability.

Air discharges can be set from 2 kV to > 25 kV, which should be applied only to external connections. This type of ESD testing is applied usually to areas where the application will be subject to ESD discharges caused by charged humans approaching the equipment. This type of ESD testing can cause repeatability problems due to the angle of the tip being applied and the speed of the tip approaching the discharge point. Applying air-discharge ESD to internal tracks or MCU pins can cause damage and is, therefore, recommended only on PCB or application areas that will be subjected to ESD caused by humans.

### *Transient Burst Generator – (IEC 801-4)*

This very severe test consists of a generator that places a train of pulses of ESD-type waveforms on the AC mains power. These pulses can be programmed up to 4 kV. This type of testing normally is carried out on electrical circuits that get their power from the mains. If the application fails this type of testing, usually a redesign of the power supply is needed, although sometimes a poorly decoupled MCU will fail this test even though the power supply design may be good. Passing this very aggressive test ensures that the design generally will pass any other immunity tests.

## SUMMARY

This application note gives designers an introduction to designing MCU circuits with EMC considerations included in the design phase. EMC problems can be and normally are made of more than one design flaw. When possible, by designing EMC improvements – be it hardware or software design – will ensure that the final product will perform well for both immunity and radiation. Each point raised in this application note will increase immunity performance, and checking each point raised one by one will ensure that the MCU application will perform well in EMC. If following all these points still does not achieve the EMC performance required, then more expensive solutions normally will be the only way to improve, such as multi-layer PCB, better filtering in power supply, and mechanical shielding of external connections. These type of solutions need a more in-depth EMC knowledge, which most engineers can acquire from EMC textbooks and EMC design courses offered by many EMC test houses and regulators.

# BIBLIOGRAPHY

*EMC for Product Designers* by Tim Williams (Butterworth-Heinemann Ltd.)

*Noise Reduction Techniques* by Henry W. Ott (Wiley-Interscience) in Electronic Systems

## *How to Reach Us:*

**Home Page:**
www.freescale.com

**E-mail:**
support@freescale.com

**USA/Europe or Locations Not Listed:**
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

**Europe, Middle East, and Africa:**
Freescale Halbleiter Deutschland GmbH
Technical Information Center
Schatzbogen 7
81829 Muenchen, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

**Japan:**
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064
Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

**Asia/Pacific:**
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
+800 2666 8080
support.asia@freescale.com

***For Literature Requests Only:***
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com



freescale™
semiconductor