Hands-on Workshop: EMC in MCU Applications
APF-IND-T0288

Dennis Lui
Application Engineer,
AP IMM AE Team

June 2013
Objectives

• Aware of the cost of EMC fixing.
• Understand the basic knowledge on EMC.
• Learn how to apply EMC improvement techniques in system level, hardware board and software design.
• Acquire EMC problem solving skill through a practical real case study on PCB layout analysis and enhancement.
Introduction
  • Cost of EMC Fixing
  • Product Design Flow Consideration

Fundamental
  • Background and Basic Mechanisms
  • Chip level EMC Consideration
  • Standard EMC test setup

EMC Improvement Techniques
  • System level EMC Consideration
  • Hardware Techniques
  • Real Cases Sharing
  • Defensive Software Techniques
Agenda

EMC Real Case Study
- PT60 Touch Board
- Circuit Analysis
- PCB Layout Analysis – Exercise
- PCB Layout Issues and Modifications
- Additional PCB Layout Enhancement

Application Note
Summary
Q&A
Introduction
Introduction

• In recent times, there has been a tremendous increase in the use of electronic and programmable electronic devices.

• Electromagnetic Compatibility of a full system has become one of the major technical issues developers have nowadays.

• If it is ignored early in the design cycle, and problems are encountered during testing or product in the field, fixes become very expensive.
Cost of EMC Fixing
Product Design Flow Consideration

1. Customer Request → Industrial Design
   - Electronic
     - Specification
     - Schematic Design
     - PCB Layout
     - Prototype
     - Pre compliance EMC Test
     - Pilot Run
     - Compliance EMC Test
     - Mass Production
     - Product in Field
   - Mechanical
     - Specification
     - Design Documentation
     - Tooling
     - Preliminary Sample
     - Final Sample

2. Electronic / SW Fix → Mechanical Fix
3. Very High Cost for EMC Fix
Fundamental 基本原理
Background and Basic Mechanisms
**Definition**

**EMC** is the ability of a device, equipment or system to function satisfactorily in its electromagnetic environment equipment without introducing intolerable electromagnetic disturbances to anything in that environment.

**EMS** 被其它干扰

(EMC 电磁兼容性)

**EMI** 干扰其它

(EMI 是电磁干扰)

**EMS** 是设备、设备或系统不能在电磁干扰下正常工作的能力。注意，这种干扰就是缺乏抗电磁性。

**EMI** 定义为由于电磁干扰而对设备、设备或系统性能的恶化。 (EMI 一个系统对另一个系统或其它系统的影响已被研究了很长时间。)
Background

Technology Change lets the physical dimension change and the logic become faster and faster, the EMC problems then arise (more sensitive with noise).

- 0.7µm, 2 metal layers, 5V
- Up to 200,000 devices on a chip
- CPU frequency 50MHz

10 years of evolution

- 0.18µm, 6 metal, 2V
- Up to 250,000,000 devices
- CPU frequency 1GHz

- 40 pins
- 1000 pins
Background

- Voltage supply decrease, Current amplitude keeps constant and Faster switching lets $di/dt$ stronger which will increase many EMC problems.

- Voltage supply decreases
- Current amplitude keeps constant
- Faster switching

Stronger $di/dt$

Increased EMC problems
Fundamental

Chip Level EMC Consideration
Basic Mechanism

(1) Conducted mode

Bus or supply lines propagate the parasitic signal

(2) Radiated mode

Interconnects serve as antennas for emitting radiated energy
Basic Mechanism

EFT/ESD Performance affecting area

EMI source → Coupling path → Receptor

Control emissions
(Reduce noise source level)
(Reduce propagation efficiency)

Control susceptibility / immunity
(Reduce propagation efficiency)
(Increase receptor immunity)
Chip Level EMC Consideration

Current switching $\Rightarrow$ parasitic emission

$\Delta V = L \frac{\Delta i}{\Delta t}$

Supply (V)

$\Delta V = 10nH(100mA)/(1ns) = 1V$

1V loss

time (ns)
Chip Level EMC Consideration

Example of a long power supply interconnect:

- L increases ⇒ new resonance in emission spectrum

Activated block

Vdd

Vss

long supply interconnect

Effect of serial resistance and inductance on the simple model

2nd resonance

Simple model

Simple model + L & R on chip

10dBμV/div

10

f (MHz)

100

1000
Chip Level EMC Consideration

Place power pairs at mid-die:

- less inductance
- pairs minimizes outward coupling

Almost no coupling: cancellation when currents are identical

Strong Coupling/Emission
**Chip Level EMC Consideration**

- Package major contribution: **inductance**

<table>
<thead>
<tr>
<th>Packaging</th>
<th>Definition</th>
<th>Inductance</th>
<th>Capacitance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dual In Line (DIL)</td>
<td>2 – 15 nH</td>
<td>1 – 10 pF</td>
<td></td>
</tr>
<tr>
<td>Shrink Dual In Line (SDIL)</td>
<td>1 – 10 nH</td>
<td>1 – 10 pF</td>
<td></td>
</tr>
<tr>
<td>Small Outline Package (SOP)</td>
<td>1 – 7 nH</td>
<td>1 – 7 pF</td>
<td></td>
</tr>
<tr>
<td>Quad Flat Pack (QFP)</td>
<td>3 – 7 nH</td>
<td>2 – 5 pF</td>
<td></td>
</tr>
<tr>
<td>Ball Gate Array (BGA)</td>
<td>0.5 – 10 nH</td>
<td>1 – 10 pF</td>
<td></td>
</tr>
<tr>
<td>Fine Pitch Ball Gate Array (FBGA)</td>
<td>0.5 – 10 nH</td>
<td>1 – 20 pF</td>
<td></td>
</tr>
<tr>
<td>Mold Chip Scale Package (MCSP)</td>
<td>0.5 – 5 nH</td>
<td>1 – 15 pF</td>
<td></td>
</tr>
</tbody>
</table>

- **Lead Bonding Pins Balls** equivalent to wire over a ground plane: **inductance**

  - \[ L = 0.5 - 1\,\text{nH/mm} \]
  - \[ C = 0.2\,\text{pF/mm} \]
Fundamental
Standard EMC Test Setup
IEC6100-4-4 EFT Test Setup

- Impulse Noise Simulator
- Insulation Transformer
- Insulation Sheet
- EUT
- Ground Reference Plane
IEC61000-4-2 ESD Test Setup

Temperature and Humidity control

ESD Gun (Contact discharge)

Load (100W)

Isolation Transformer

220V to 110V Transformer

ESD Noise Simulator

EUT (MCU)

Ground Reference Plane

Horizontal Coupling Plane

Temperature and Humidity control

ESD Gun (Contact discharge)

Load (100W)

Isolation Transformer

220V to 110V Transformer

ESD Noise Simulator

EUT (MCU)

Ground Reference Plane

Horizontal Coupling Plane
EMC Improvement Techniques
System Level EMC Consideration
Common MCU Failure Mode

• Most Home Appliance customers are more concerned about the Immunity / Susceptibility (EFT/ESD) aspect

• Common MCU Failure mechanism for EFT/ESD test
  − MCU momentarily runs away.
  − MCU Reset occur.
  − MCU hangs, Interrupt / Reset can recover.
  − MCU hangs, only Power ON/OFF can recover.
  − MCU hangs, latch-up occurs, might cause silicon and other circuit element damage.

The EFT/ESD failure is not only due to the MCU but also caused by many other factors such as components selections, circuit design, PCB layout .....etc.
EMC Improvement Techniques

• Hardware Techniques
  - Component Selection & Circuit Design Basic
  - Line Termination Techniques
  - Micro-controller Consideration
  - PCB layout Techniques

• Defensive Software Techniques
  - Using Watch-Dog function to prevent MCU core lock
  - Update all output and critical status registers periodic
  - Fill unused memory with “NOP”. “Jump” to known place
  - Always re-confirm edge triggered interrupt

Software does not eliminate the transient or noise. It can only attempt to control the MCU response to the transient or noise.
EMC Improvement Techniques

Hardware Techniques
Component Selection

- **Component Packages**
  - SMD > Radial Leaded > Axial Leaded ---- Lower parasitic

- **Resistors**
  - Carbon > Metal > Wire Wound ---- Lower Inductance (Depends on application)

- **Capacitors**
  - Bypass (reduce transient circuit demand): Aluminum / Tantalum [10 - 470uF]
  - Decoupling (localized source of DC power, reduce switching noise): Ceramic (Low ESR, right self-resonant freq and place as close as possible to Vdd & Vss of device) [0.01 – 0.1uF]

- **Inductors**
  - Forms a link between magnetic & electric fields
  - Closed-loop > Open loop
  - e.g. Ferrite bead provides 10db attenuation over high freq. (i.e. low in DC)

- **Diodes**
  - Solve for impedance mismatching (signal reflection & ringing)
  - Slow down the fast rising & falling edges of the signal (reduce emission)

Each I/O port had added the diode protection circuit
Line Termination

• When a circuit is operating at high speeds, the impedance matching between the source and destination is very important. Because mismatching will cause signal reflection and ringing. The excess RF energy will radiate or couple to other parts of the circuit, causing EMI problems. Termination of signals help to reduce these undesirable effects.

• Termination not only reduce signal reflection and ringing by matching the impedance between source and destination, but can also to slow down the fast rising and falling edges of the signals (i.e. emission).

<table>
<thead>
<tr>
<th>Termination Type</th>
<th>Relative Cost</th>
<th>Delay Added</th>
<th>Power Required</th>
<th>Critical Parameters</th>
<th>Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Series</td>
<td>Low</td>
<td>Yes</td>
<td>Low</td>
<td>RS = Z0 = R0</td>
<td>Good DC noise margin</td>
</tr>
<tr>
<td>Parallel</td>
<td>Low</td>
<td>Small</td>
<td>High</td>
<td>R = Z0</td>
<td>Power consumption is a problem</td>
</tr>
<tr>
<td>RC</td>
<td>Medium</td>
<td>Small</td>
<td>Medium</td>
<td>R = Z0, C = 20 to 600pF</td>
<td>Check bandwidth and added capacitance</td>
</tr>
<tr>
<td>Thevenin</td>
<td>Medium</td>
<td>Small</td>
<td>High</td>
<td>R = 2 \ Z0</td>
<td>High power for CMOS</td>
</tr>
<tr>
<td>Diode</td>
<td>High</td>
<td>Small</td>
<td>Low</td>
<td>—</td>
<td>Limits overshoot; some ringing at diodes</td>
</tr>
</tbody>
</table>
MCU Consideration

- **Background**
  - **Reduce die size**: Faster transistors > Faster rise & fall times => Harmonic content
  - **Speed increase**: Power increase => Circuit design & PCB layout (to reduce EMC)

- **I/O Port**
  - Input only port pin: **Properly termination** (add series resistor and filtering capacitor)
  - Unused I/O pin: Set o/p without connection OR terminating with resistor to Vdd/Vss

- **IRQ pin**
  - Most sensitive pin and need to be terminated in fixed state (e.g. pullup + capacitor)

- **Reset pin**
  - POR by RC time delay (2K < R < 10K)
  - Diode clamp (prevent voltage over Vdd & provide faster discharge time) ---- Pullup + Capacitor + Diode

---

**Loop area as small as possible**
MCU Consideration

- Oscillators
  - MCU build-in inverter for external crystal / ceramic resonator
  - Need to use proper value of feedback resistor Rb, series resistor Rs, C1 & C2 --- Those values depends on both characteristic of build-in inverter and external crystal/ceramic resonator

<table>
<thead>
<tr>
<th>Clock Source</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ceramic Resonator</td>
<td>Lower cost</td>
<td>Sensitive to EMI, humidity and vibration. Drive circuit matching.</td>
</tr>
<tr>
<td>Crystal</td>
<td>Low cost</td>
<td>Sensitive to EMI, humidity and vibration. Drive circuit matching.</td>
</tr>
<tr>
<td>Crystal Oscillator Module</td>
<td>Insensitive to EMI and humidity. No additional components or matching issues.</td>
<td>High cost. High power consumption. Large size. Sensitive to vibration.</td>
</tr>
<tr>
<td>RC Oscillator</td>
<td>Lowest cost.</td>
<td>Sensitive to EMI, humidity and vibration. Poor temperature and supply voltage rejection. Usually large size.</td>
</tr>
<tr>
<td>Silicon Oscillator</td>
<td>Insensitive to EMI, humidity, and vibration. Fast startup. Small size. No additional components or matching issues.</td>
<td>Temperature sensitivity generally worse than crystal and ceramic resonator. Some have high power consumption.</td>
</tr>
</tbody>
</table>

Internal oscillator is better EMC performance
PCB Layout Consideration

• Background
  - PCB is an inherent part of the system and does not add extra cost
  - Good PCB layout gains better EMC performance in a cost saving manner (at the beginning)

• PCB General Guideline
  - Increase separation between tracks to minimize crosstalk by capacitive coupling
    ▪ Minimize crosstalk & noise coupling by adjacent traces (magnetic flux coupling)
    ▪ Place sensitive & high freq traces far away from high noise Vdd tracks
  - Maximize PCB capacitance by placing Vdd & Vss tracks in parallel
  - Vias: Kept to minimum in ground and high speed signal, equal no. in add/data bus
  - 45° angled tracking: Avoid field concentration at inner edge
  - Constant track width: reflections & line impedance imbalances
  - Priority: Ground > Power > Sensitive signal > Data
  - Decouple local supplies ICs:
    ▪ Localized decoupling capacitor (reduce switching noise propagating along the supply rail)
    ▪ Bypass capacitor (low freq filter, potential reservoir for sudden power demands)
PCB Layout Consideration

- **PCB General Guideline**
  - **Minimize loop areas:** Keeping signal tracks and its ground return close together.
    > minimize the ground loop > avoid potential aerial loops (particularly for high speed single-ended signal)
  - **RF current reference plane:**
    - Lower impedance of return path => Better EMC performance of PCB
    - Long return path create mutual coupling because RF current from load to source
    - Return path as short as possible, loop area as small as possible
  - **Widen Vdd & Vss tracks to reduce Common Mode Impedance (60mil min.)**
  - **Segmentation:** Physical separation to reduce coupling between different types of circuit, particular Vdd & Vss tracks (Power Source > Digital > Analog > DC > Interface circuits)
EMC Improvement Techniques
Real Case Sharing
Case 1: PCB Layout EMC Review

- Board: Air Conditioner double-layer control board
- MCU: MC9S08AW60CFUE (64-QFP)
- Problem:
  - Missing the bypass capacitor in the power source
  - Ground plane split-off
    - High CM impedance
    - Degrade effectiveness of decoupling and filtering effect
  - Large ground loop
  - Large power loop
  - Wrong ground connection for resonator
Case 2: MCU Control Board EMC (ESD) Review

- Board: UPS double-layer control board + single-layer power board
- MCU: MC9S08AC60CFU (64-QFP)

Original (+4.0KV/-4.0KV) => Modified (+6.0KV/-6.0KV) in double-layer board
Case 3: MCU Control Board EMC (EFT) Review

- Board: Air Conditioner single-layer control board
- MCU: MC68HC908AB32CFU (64-QFP)

Original (+4.25KV/-3.75KV) => Modified (+8.0KV/-7.75KV) in single-layer board
EMC Improvement Techniques
Defensive Software Techniques
Defensive Software Design

- The software design cannot change the physical media which couples the noise into the system, or reduce the absolute magnitude of noise generated from external sources.

- The software must be able to identify a particular event if it is a false alarm triggered by noise sources or it is a normal driven event and then make a smart decision on corresponding actions.

- Good defensive software design is one of the key factors to improve overall performance, system protection and operating stability in noisy environments (e.g. EMC).
System Configuration

Software can act as Digital Filter to suppress EMC noise

Always check / confirm input “High/Low” for Polling inputs and Edge Trigger Interrupts

Update All Status Registers, DDR, etc… once every 50/60 Hz
Lost of 50/60 Hz cycles = Power Failure
Implementations

- Enable Watch-Dog to avoid code runaway.
- Refresh data direction registers periodically.
- Fill unused memory to avoid code runaway.
- Define all interrupt vectors even those that are not used.
- Select Frequency Locked Loop (FLL) engaged mode.
- Always re-confirm edge triggered event.
- Enable input glitch filter (PT60 build-in feature).
- Enable slew rate control on output port.
Enable WATCH-DOG

• The Watch-Dog (WDOG) function forces a system reset, when the application software fails to execute as expected.

• To prevent a system reset from the WDOG timer when it is enabled, application software must reset the WDOG counter periodically.

• It is recommended to put the WDOG refresh routine in the main loop instead of sub-routines and interrupt routines.

• There is a new feature for WDOG function in PT60 which allows the user to reconfigure the parameters used by the module.
#define __RESET_WATCHDOG() (void)(WDOG_CNT = 0xA602U, WDOG_CNT = 0xB480U)

void main(void) { 
 
Sys_Init();
PE_Load_level_init();
MicrowaveInit();

wdog_unlock(); // executing an unlock sequence
WDOG_CS1 = 0xA0; // Set WDOGA = 1 to allow reconfigure watchdog
WDOG_CS2 = 0x01; // Select internal 1 kHz as watchdog clock
WDOG_TMRH = 0x03; // Set watchdog counter to 1000
WDOG_TMRH = 0xE8;
WDOG_WINH = 0; // disable window mode option
WDOG_WINL = 0;
EnableInterrupts;

for(;;) {
DisableInterrupts; // disable interrupts
__RESET_WATCHDOG(); // Reset the watchdog counter
EnableInterrupts; // enable interrupt
MicrowaveTask(); // Application main task
} /* loop forever */
  /* please make sure that you never leave main */
} /* end of Main */
Refresh Data Direction Registers

• The input or output direction state for each port pin should be recovered to the expected condition, if it has been changed by any transient noise accidentally.

• The 50Hz or 60Hz periodic signal output from AC power supply through an optical coupling circuit can be used as a trigger signal for I/O direction registers update.

• The variable mStatusRegisterUpdate_d is set to TRUE at each falling edge of the 50Hz or 60Hz signal and then clear to FALSE at the end of the refresh register routine.

• The Port C is configured with input and output multiplex function and the directional status is updated in another key scanning routine.
void StatusRegisterUpdate(void) {
    if (mStatusRegisterUpdate_d == TRUE) {
        IOstatusRegPTAIE_Port = ~IOstatusRegPTADD_Value;
        IOstatusRegPTAOE_Port = IOstatusRegPTADD_Value;
        IOstatusRegPTBIE_Port = ~IOstatusRegPTBDD_Value;
        IOstatusRegPTBOE_Port = IOstatusRegPTBDD_Value;
        /* Port C is used as Input and Output port and refresh by key scanning routine */
        //    IOstatusRegPTCIE_Port = ~IOstatusRegPTCDD_Value;
        //    IOstatusRegPTCOE_Port = IOstatusRegPTCDD_Value;
        IOstatusRegPTDIE_Port = ~IOstatusRegPTDDD_Value;
        IOstatusRegPTDOE_Port = IOstatusRegPTDDD_Value;
        IOstatusRegPTEIE_PORT = ~IOstatusRegPTEDD_Value;
        IOstatusRegPTEOE_Port = IOstatusRegPTEDD_Value;
        IOstatusRegPTFIE_Port = ~IOstatusRegPTFDD_Value;
        IOstatusRegPTFOE_Port = IOstatusRegPTFDD_Value;
        IOstatusRegPTHIE_PORT = ~IOstatusRegPTHDD_Value;
        IOstatusRegPTHOE_Port = IOstatusRegPTHDD_Value;
        mStatusRegisterUpdate_d = FALSE;
    }
}

Fill Unused Memory

• Unused memory, Flash or RAM should be filled with a pre-defined content such that the MCU does not execute any unexpected instruction when the normal execution flow is disturbed by external noise sources.

• It is recommended to fill all unused memory with illegal opcode (e.g. 0x8D) or No Operation (NOP) instruction.

• The unused memory can be filled by adding the FILL option in the linker parameter file (e.g. Project.prm)
/* This is a linker parameter file for the mc9s08pt60 */
/* CodeWarrior will pass all the needed files to the linker by command line. But here you may add your own files too. */

NAMES END
/* Here all RAM/ROM areas of the device are listed. Used in PLACEMENT below. */

SEGMENTS
  Z_RAM = READ_WRITE 0x0040 TO 0x00FF FILL 0x9D;
  RAM   = READ_WRITE 0x0100 TO 0x0BCF FILL 0x9D;
  RAM_CODE = READ_ONLY 0x0BD0 TO 0x103F FILL 0x9D;
  ROM   = READ_ONLY 0x3200 TO 0xF5FF FILL 0x9D;
  ROM1  = READ_ONLY 0x1040 TO 0x2FFF FILL 0x9D;
  ROM2  = READ_ONLY 0xFF80 TO 0xFFAF FILL 0x9D;
  EEPROM = READ_ONLY 0x3100 TO 0x31FF FILL 0x9D;
  FLASH_TO_RAM = READ_ONLY 0xF600 TO 0xFBFF RELOCATE_TO 0x0BD0;
  USER_PARAM = READ_ONLY 0xFC00 TO 0xFDFF;

/* INTVECTS Interrupt Vectors */
END

Reserved for
Define All Interrupt Vectors

• The reason to define the interrupt vectors for each unused interrupt function is to allow the MCU to jump into a pre-defined interrupt routine and

• back to previous execution step correctly when a particular unused interrupt associate flag is mistriggered by a noise source.

• The interrupt function for each unused interrupt can be the same, so just one dummy interrupt routine can be used for all unused interrupt functions.
Define All Interrupt Vectors Sample Code

```c
void (* near const _vect[])(void) @0xFFB0 = { /* Interrupt vector table */
    Cpu_Interrupt,          /* Int.no. 39 Vnvm (at FFB0) Unassigned */
    Cpu_Interrupt,          /* Int.no. 38 Vkbi1 (at FFB2) Unassigned */
    Cpu_Interrupt,          /* Int.no. 37 Vkbi0 (at FFB4) Unassigned */
    Cpu_Interrupt,          /* Int.no. 36 Vtsi (at FFB6) Unassigned */
    Cpu_Interrupt,          /* Int.no. 35 Vrtc (at FFB8) Unassigned */
    Cpu_Interrupt,          /* Int.no. 34 Viic (at FFBA) Unassigned */
    Cpu_Interrupt,          /* Int.no. 33 Vspi1 (at FFBC) Unassigned */
    Cpu_Interrupt,          /* Int.no. 32 Vspi0 (at FFBE) Unassigned */
    Cpu_Interrupt,          /* Int.no. 31 Vsci2txd (at FFC0) Unassigned */
    Cpu_Interrupt,          /* Int.no. 30 Vsci2rxd (at FFC2) Unassigned */
    Cpu_Interrupt,          /* Int.no. 29 Vsci2err (at FFC4) Unassigned */
    Cpu_Interrupt,          /* Int.no. 28 Vsci1txd (at FFC6) Unassigned */
    Cpu_Interrupt,          /* Int.no. 27 Vsci1rxd (at FFC8) Unassigned */
    Cpu_Interrupt,          /* Int.no. 26 Vsci1err (at FFCA) Unassigned */
    Cpu_Interrupt,          /* Int.no. 25 Vsci0txd (at FFCC) Unassigned */
    Cpu_Interrupt,          /* Int.no. 24 Vsci0rxd (at FFCE) Unassigned */
    Cpu_Interrupt,          /* Int.no. 23 Vsci0err (at FFDO) Unassigned */
    Cpu_Interrupt,          /* Int.no. 22 Vadc (at FFD2) Unassigned */
    Cpu_Interrupt,          /* Int.no. 21 Vacmp (at FFD4) Unassigned */
    Cpu_Interrupt,          /* Int.no. 20 Vmtim1 (at FFD6) Unassigned */
    Cpu_Interrupt,          /* Int.no. 19 Vmtim0 (at FFD8) Unassigned */
    Cpu_Interrupt,          /* Int.no. 18 Vftm0ovf (at FFDA) Unassigned */
    Vftm0ch1_Interrupt      /* Int.no. 17 Vftm0ch1 (at FFDC Used */
    Vftm0ch0_Interrupt      /* Int.no. 16 Vftm0ch0 (at FFDE) Used */
    Cpu_Interrupt           /* Int.no. 15 Vftm1ovf (at FFE Unassigned */
    Cpu_Interrupt           /* Int.no. 14 Vftm1ch1 (at FF Unassigned */
    Cpu_Interrupt           /* Int.no. 13 Vftm1ch0 (at FFE4 Unassigned */
    Cpu_Interrupt           /* Int.no. 12 Vftm2ovf (at FFE6 Unassigned */
    Cpu_Interrupt           /* Int.no. 11 Vftm2ch5 (at FFE8 Unassigned */
    Cpu_Interrupt           /* Int.no. 10 Vftm2ch4 (at FFEA Unassigned */
    Cpu_Interrupt           /* Int.no.  9 Vftm2ch3 (at FFEc Used */
    Vftm2ch2_Interrupt      /* Int.no.  8 Vftm2ch2 (at FFEf Used */
    Cpu_Interrupt           /* Int.no.  7 Vftm2ch1 (at FFP0 Unassigned */
    Cpu_Interrupt           /* Int.no.  6 Vftm2ch0 (at FFP2 Unassigned */
    Cpu_Interrupt           /* Int.no.  5 Vftm2flt (at FFP4 Unassigned */
    Cpu_Interrupt           /* Int.no.  4 Vcl1k (at FFP6 Unassigned */
    Cpu_Interrupt           /* Int.no.  3 Vlvd (at FFP8 Unassigned */
    Cpu_Interrupt           /* Int.no.  2 VirgVwdog (at FFPA Unassigned */
    Cpu_Interrupt           /* Int.no.  1 Vswl (at FFFC) Unassigned */
    Startup                /* Int.no.  0 Vreset (at FFFE) Reset vector */
};
```
Select FLL Engaged Mode

• It is recommended to enable the FLL engaged mode with external reference clock in the internal clock source (ICS) module which provides clock source option for the MCU.

• For example, a 4 MHz crystal oscillator is used as the reference clock and the target bus frequency is set to 4MHz.
  - Ref clock = 4MHz Xtal / 128 = 31.25kHz
  - FLL output = Ref clock x 512 = 16MHz
  - Final bus clock = FLL output / 4 = 4MHz.
Select FLL Engaged Mode (cont.)

- The advantages of the frequency conversion using the FLL module instead of directly using the external crystal oscillator as the bus clock are:

  - The impact of transient noise glitch on high frequency clock source (direct using crystal oscillator) is more significant compared to a low frequency clock source (divided by 128) in terms of the glitch width against the clock cycle.

  - In general, the response of the FLL module is not fast enough to react to such kind of short pulse noise due to the lowpass filter characteristic.
Select FLL Engaged Mode Sample Code

```c
#if (EXT_CLK_CRYST == 8000)
 /* 8MHz */
 ICS_C1_RDIV = 3; /* now the divided frequency is 8000/256 = 31.25K */
#elif (EXT_CLK_CRYST == 4000)
 /* 4MHz */
 ICS_C1_RDIV = 2; /* now the divided frequency is 4000/128 = 31.25K */
#else
 #error "Error: crystal value not supported!
#endif

/* change FLL reference clock to external clock */
ICS_C1_IREFS = 0;

/* wait for the reference clock to be changed to external */
asm{
    nop
    nop
}
while(ICS_S & ICS_S_IREFST_MASK);

/* wait for FLL to lock */
while(!(ICS_S & ICS_S_LOCK_MASK));

/* now FLL output clock is 31.25K*512 = 16MHz */

#if (BUS_CLK_HZ == 16000000)
 // now system/bus clock is 16MHz
 ICS_C2_BDIV = 0;
#elif (BUS_CLK_HZ == 4000000)
 // now system/bus clock is 8MHz
 ICS_C2_BDIV = 1;
#endif

/* clear Loss of lock sticky bit */
ICS_S |= ICS_S_LOLS_MASK;
```
Re-confirm Edge Triggered

• Multiple reading on input data for each edge triggered interrupt service is almost an essential technique to confirm if the input event is valid and driven by determined sources.

• The 50Hz or 60Hz signal coupling from the AC power line is applied to a timer input capture pin in MCU and the interrupt service routine will be called for each falling edge detected at this pin.

• Multiple reading of the pin status is defined in a “for loop” to ensure all readings from the pin status are the same before the TRUE flag is set up for this event.
Re-confirm Edge Triggered (Cont.)

• The timing slot between each successive reading inside the for loop should be adjusted with some kind of irregular pattern such that an even distributed noise pattern will not be recognized as a valid event.

• A simple random delay function is inserted between each reading such that the overall repeat period is not consistent.

• The random delay variable is a free running counter value captured, when there is an interrupt trigger event from the zero-crossing input pin.
/* Random Delay Loop */
uint8_t RandomDelay(void){
    uint16_t random_16bit = RANDOM_COUNTER;
    mRandomDelayCount = TPMxCnVLvalue(random_16bit);
    mRandomDelayCount &= gRandomDelayCountMask_c;
    return mRandomDelayCount;
}

uint8_t ZeroTriggerDebounce[gNumberOfZeroTriggerDebounce_c];
uint8_t iRead = 0;

for (iRead = 0; iRead < gNumberOfZeroTriggerDebounce_c; iRead++){
    uint8_t idelay;
    idelay = RandomDelay();
    while(idelay > 0){
        --idelay;
        asm(nop);
    }
    ZeroTriggerDebounce[iRead] = ZeroTrigger_GetPinValue();
    if (iRead != 0){
        if((ZeroTriggerDebounce[iRead] != ZeroTriggerStateIdle) &&
        (ZeroTriggerDebounce[iRead] == ZeroTriggerDebounce[iRead - 1])){
            mZeroTriggerRead = ZeroTriggerDebounce[iRead];
        }else {
            mZeroTriggerRead = ZeroTriggerStateIdle;
        }
    }else {
        mZeroTriggerRead = ZeroTriggerStateIdle;
    }
}
Input Glitch Filter (PT60 Build-in Feature)

• The Input Glitch Filter is a new feature in PT60 MCU that provides, a simple low-pass filter for each port pin that is configured as a digital input.

• The glitch width threshold can be easily adjusted between 1~4096 BUSCLKs.

• This configurable filter provides an adaptive way to handle different types of transient noises with deterministic pulse width in nature which are difficult to handle by traditional analog filters.
Input Glitch Filter Sample Code

```c
#ifdef PortFilterEnable
    setReg8Bits(PORT_FCLKDIV, 0x02);   // set FLTDIV1 to Bus clock divided by 8
    setReg8Bits(PORT_IOFLT0, 0x55);    // select FLTDIV1 for Port A/B/C/D
    setReg8Bits(PORT_IOFLT1, 0x55);    // select FLTDIV1 for Port E/F/G/H
#endif
```
Slew Rate Control

• Slew rate control can be enabled for each port pin by setting the corresponding bit in one of the slew rate control registers (PTxSEn).

• Slew rate control limits the output transition rate and reduces EMC emissions

• Slew rate control has no effect on pins which are configured as inputs.
Slew Rate Control Sample Code

```c
/* PTASE: PTASE7=1,PTASE6=1,PTASE4=1,PTASE3=1,PTASE2=1,PTASE1=1,PTASE0=1 */
  setReg8Bits(PTASE, 0xDF);

/* PTBSE: PTBSE7=1,PTBSE6=1,PTBSE5=1,PTBSE4=1
PTBSE3=1,PTBSE2=1,PTBSE1=1,PTBSE0=1 */
  setReg8(PTBSE, 0xFF);

/* PTCSE: PTCSE7=1,PTCSE6=1,PTCSE5=1,PTCSE4=1
PTCSE3=1,PTCSE2=1,PTCSE1=1,PTCSE0=1 */
  setReg8(PTCSE, 0xFF);

/* PTDSE: PTDSE5=1,PTDSE4=1,PTDSE3=1,PTDSE2=1,PTDSE1=1,PTDSE0=1 */
  setReg8Bits(PTDSE, 0x3F);
```
EMC Real Case Study
PT60 Touch Board
Background

Application: Home Appliance
Product: Induction Stove
MCU: MC9S08PT60VLH (64-LQFP)
Boards:
• Main Board – Host Controller and Power Supply
• Touch Board – PT60 and LED driver
EMC Issues:
• EFT – FAIL 2kV (System Reset and display flickering)
• ESD – FAIL 20kV (System Reset and no LED display)
System Descriptions

- The system consists of a main board using proprietary solution for power stage control and a touch board using PT60 microcontroller device with touch sensing module as user input interface.
- The 5V power supply and host controller are located on main board. The supply and control signals are connected to the interface board thru a single flat cable.
- The PT60 on touch board is used for user touch key detection and command transmission to main board host controller thru I2C bus.
- The touch board also has a dedicated LED driver IC for 7-segment and discrete LED driving and controlled by a serial bus separated from PT60.
Product Structure

- Touch Interface
- 7-Seg
- Flat Cable
- Main Board
- Touch Board
- PT60
- LED Driver
EMC Real Case Study
Circuit Analysis
EMC Real Case Study

PCB Layout Analysis
PCB Layout Analysis - Exercise

• Review the original PCB layout and highlight all EMC critical signal paths or circuits on the layout.
• Apply the EMC knowledge to identify all existing EMC related issues on the layout.
• Recommend a solution or modification for each issue.
EMC Real Case Study
PCB Layout Issues and Modifications
EMC Real Case Study
Additional PCB Layout Enhancement
Application Note
Application Note

- **AN4438**
  - EMC Design Considerations for MC9S08PT60

- **AN4476**
  - System Design Guideline for 5V 8-bit families in Home Appliance Applications

- **AN4463**
  - How To Develop a Robust Software in Noise Environment

- **AN2321**
  - Designing for Board Level Electromagnetic Compatibility

- **AN2321_GB**
  - Designing for Board Level Electromagnetic Compatibility (Chinese Version)

- **AN2764**
  - Improving the Transient Immunity Performance of Microcontroller-Based Applications

- **AN1050**
  - Designing for Electromagnetic Compatibility (EMC) with HCMOS Microcontrollers

- **AN1259**
  - System Design and Layout Techniques for Noise Reduction in MCU-Based Systems

- **AN1263**
  - Designing for Electromagnetic Compatibility with Single-Chip Microcontrollers
Application Note

- **AN1705**
  - Noise Reduction Techniques for Microcontroller-Based Systems
- **AN2015**
  - Power-On, Clock Selection, and Noise Reduction Techniques for the Motorola MC68HC908GP32
- **AN1744**
  - Resetting Microcontrollers During Power Transitions
- **EB413**
  - Resetting MCUs
- **AN1783**
  - Determining MCU Oscillator Start-up Parameters
- **EB396**
  - Use of OSC2/XTAL as a Clock Output on Motorola Microcontrollers
- **AN1706**
  - Microcontroller Oscillator Circuit Design Considerations
- **EB398**
  - Techniques to Protect MCU Applications Against Malfunction Due to Code Run-Away
Summary
Summary

- Cost of the EMC fixing – consider EMC early in the design cycle.
- EMC – EMI (Conducted & Radiated modes) and EMS [EFT, ESD, Conducted & Radiated modes]]
- EMC problem: Faster switching (reduce dimension) & V decrease, current constant (product requirement) => Increase \( \frac{di}{dt} \)
- EMC Consideration: Hardware & Defensive Software Techniques
  - Hardware:
    - Component selection, placement, circuit design, and line termination
    - MCU consideration and PCB layout techniques
  - Software:
    - COP, Periodic Register Update, Fill NOP/Jump to known place and Re-confirm edge triggered interrupt.
- EMC Practical Example:
  - Floorplan/Segmentation (Grouped by power domain, Bypass/Decoupling, Filters)
  - Power Distribution: (GND routes first, Minimize vias, Minimize GND impedance, Vdd and GND in parallel)
  - Bypassing: (Local source to limit voltage variations, connect to capacitor first, Minimize loop area)
  - Decoupling: (Remove noise between components, connect to capacitor first, Minimize loop area, connect to Vdd and GND pin of MCU as close as possible)
  - Input Port: (Filters, and termination, Minimize loop area with GND and Vdd)