Software-related EMI test pattern auto-generation for 2-stage pipeline microcontroller

Shih-Yi Yuan
dept. of communication engineering,
Feng Chia University,
Taichung, Taiwan, R.O.C.
syyuan@fcu.edu.tw

Yung-Chi Tang
Bureau of Standards, Metrology and Inspection
Taipei, Taiwan, R.O.C.
yc.tang@bsmi.gov.tw

Cheng-Chang Chen
Bureau of Standards, Metrology and Inspection
Taipei, Taiwan, R.O.C.
chang.chen@bsmi.gov.tw

***Abstract*—Time-domain measurement on microcontroller (µC) conducted electromagnetic interference (cEMI) is an essential part for software-related EMI (SW-EMI) modeling. To make a SW-EMI model, several signal process procedures should be done. One of the procedures is the time-domain waveform test pattern generation. Due to the SW-related issues, the test patterns must go through all of test pattern of the target µC machine codes (instructions). However, an efficient algorithm for automatic test pattern generation for SW-EMI is not available.**

This paper proposes an efficient pattern generation algorithm of the testing programs, which reduces the test pattern number from the permutation of all test patterns to some combination test patterns without reducing the pattern coverage. The generated test pattern size is 50% reduced by the proposed algorithm. This algorithm is dedicated to SW-EMI modeling and is implemented by MATLAB.

Keywords—software-related EMI, waveform cutting algorithm

I. INTRODUCTION

As technology progress faster than ever, microcontrollers (µC) are required by many devices, and the electromagnetic (EMI) issues are more important than ever. However, electronic products are mostly digitized and software (SW) controllable, the SW-related EMI (SW-EMI) becomes more important. Therefore, SW-EMI of electronic products is controlled by SW and indirectly controls the total system EMI. A time-domain conducted EMI (cEMI) measurement of µC is essential for the SW-EMI modeling. Several digital signal process (DSP) stages (Fig. 1) are done before the SW-EMI modeling analysis become possible. These procedures are detailed in [1].

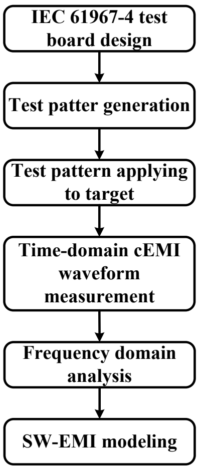

IEC 61967-4 test board design
Test patter generation
Test pattern applying to target
Time-domain cEMI waveform measurement
Frequency domain analysis
SW-EMI modeling

Fig. 1. DSP procedure for SW-EMI model building process

Generally, the internal architecture of microcontroller (µC) is unknown to EMI modelers. Different PCB boards can be used to characterize µC SW-EMI behaviors [1]. The SW-EMI behaviors are very fast and time-varying. The measurement results are continuous waveforms and should be observed on oscilloscope for fast changing fluctuation noises. These waveforms cannot be directly used to find the relationships among software and EMI and should be identified and cut to single machine code.

Before the measurement and analysis procedures, testing programs must be firstly 'burn' into the target µC and drive the µC to function. After the µC is driven by program(s), the switching noise can be observed by spectrum (frequency domain signals) or oscilloscope (time domain signals).

This paper proposed an automatic and efficient test program generation algorithm for switch noise test pattern generation. The generated test program size is very small. This algorithm is dedicated to SW-EMI modeling and is implemented by MATLAB. Although MATLAB is a general purpose DSP platform and the efficiency is not high enough, it can be used as a preliminary algorithm development. Thus, a high efficiency and accuracy time-domain waveform test pattern generation algorithm is very suitable for the algorithm development time-domain waveform.

This paper is organized as follows. Section II introduces the concept of SW-EMI modeling procedure and the importance of the instruction waveform test pattern automatic generation algorithm. Section III introduces the proposed algorithms. Section IV is the experiment result. Section V gives conclusions.

II. SW-EMI MODELING STEPS

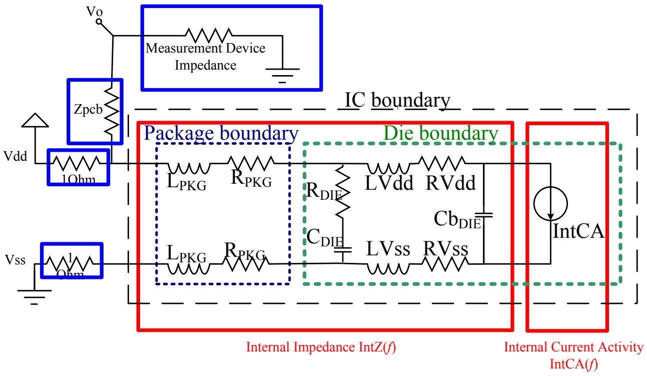


Fig. 2. Basic concept of IEC-62433 (ICEM)

The proposed method is based on IEC-62433 [2]. This is an international standard for Integrated Circuit EMI (IC-EMI) modeling. The concept of IEC-62433 is shown in Fig. 2. The Internal Inductances are separately lumped as package-level inductances (L_{PKG}) and chip-level inductances (L_{Vdd}). The chip-level capacitances are further modeled as die-level parasitic capacitance (C_{DIE}) and die-level designed capacitance (Cb_{DIE}). The resistors are also modeled and named accordingly.

There are 2 major parameters to be estimated: IntZ (the internal impedance) and IntCA (internal current activity). IEC-62433 assumes that the EMI behavior is periodically quasi-static over every clock cycle. The 'granularity' of the quasi-static period is the clock period and is proved to be a roughly adequate for most IC-EMC models. It has been successfully used to model the power/ground signal fluctuations in many µC, Application-specific integrated circuit (ASIC), and programmable devices within the range between 1MHz–2GHz.

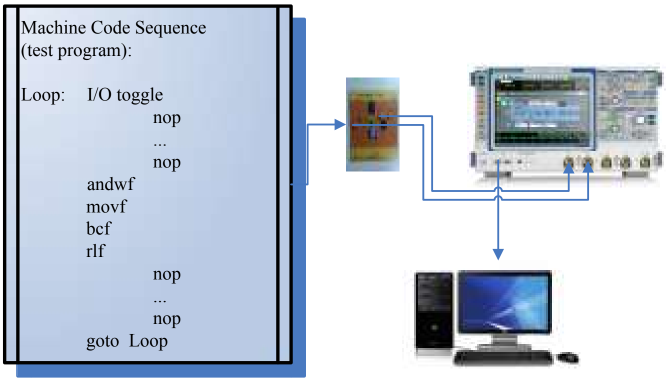


Fig. 3. Time-domain instruction waveform measurement setups

The measurement setups and the testing program are shown in Fig. 3. After programing the testing code, the time-domain cEMI waveform is captured by oscilloscope which includes the testing code's cEMI behaviors, the clock signal, and I/O marker signal. Clock is used to identify the code execution begin and the I/O marker signal is used to identify the whole sequence of code executing. And other sequence of the signal analysis can be continued as Fig. 1 described.

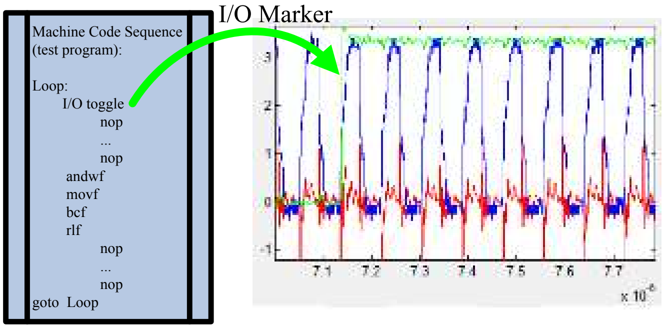


Fig. 4. Time-domain instruction waveform, clock signal, and I/O marker

An example of VDD instruction waveform (Fig. 4) is measured according to [3]. The instruction waveform (red line) is generally periodic for every 4 clock cycles (blue line). The first line in the loop of the test program in Fig. 4 is an I/O toggle machine code. This code is used to generate an "I/O marker" (green line) to indicate the beginning of the program loop. By using the I/O marker and clock signals, each instruction waveform can be easily identified (Fig. 4). The identified instruction waveforms are windowed and aligned for further process. By the help of IO marker and clock signals in Fig. 4, the cutting algorithm and aligning algorithm are developed for automatic instruction waveform signal process. These algorithms are omitted here.

It should be noted that Fig. 4 is just one example. Up to 100 different testing programs are designed through the same procedures to identify all instruction waveforms. It means more than 100 testing programs are automatically generated. The auto-generation algorithm and procedure for testing programs is very important.

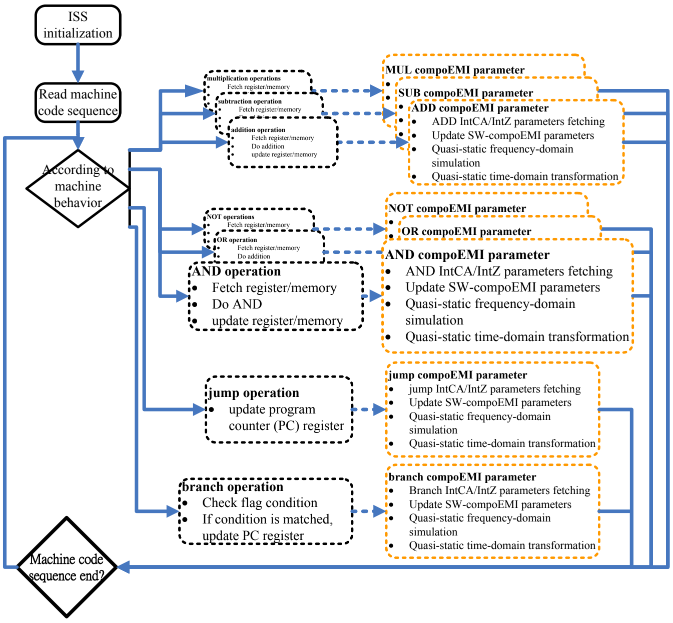

ISS initialization
Read machine code sequence
According to machine behavior
multiplication operations
Fetch register/memory
subtraction operation
Fetch register/memory
addition operation
Fetch register/memory
Do addition
update register/memory
MUL compoEMI parameter
SUB compoEMI parameter
ADD compoEMI parameter
ADD IntCA/IntZ parameters fetching
Update SW-compoEMI parameters
Quasi-static frequency-domain simulation
Quasi-static time-domain transformation
NOT operations
Fetch register/memory
OR operation
Fetch register/memory
Do addition
AND operation
Fetch register/memory
Do AND
update register/memory
NOT compoEMI parameter
OR compoEMI parameter
AND compoEMI parameter
AND IntCA/IntZ parameters fetching
Update SW-compoEMI parameters
Quasi-static frequency-domain simulation
Quasi-static time-domain transformation
jump operation
update program counter (PC) register
jump compoEMI parameter
jump IntCA/IntZ parameters fetching
Update SW-compoEMI parameters
Quasi-static frequency-domain simulation
Quasi-static time-domain transformation
branch operation
Check flag condition
If condition is matched, update PC register
branch compoEMI parameter
Branch IntCA/IntZ parameters fetching
Update SW-compoEMI parameters
Quasi-static frequency-domain simulation
Quasi-static time-domain transformation
Machine code sequence end?

Fig. 5. The modified ISS procedure

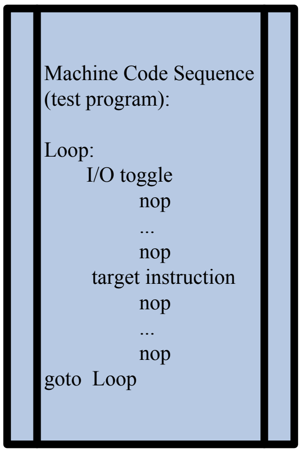
The simulator used in this paper is a modified instruction set simulator (ISS) [4] shown in Fig. 5 with corresponding machine code's sub-simulators (Fig. 5, orange dotted blocks) inside the simulation loop. During the simulation, the simulator not only does a normal ISS simulation but also the corresponding quasi-static window SW-EMI IntCA/IntZ updating and simulation. After one quasi-static period is simulated, it continues on next quasi-static period iteration. The design of the ISS is omitted in this paper.

A. *Original Auto-gen algorithm for 2-stage pipeline µC: permutation of all test programs*

If there are n machine codes in a µC and the µC is not pipelined, the algorithm is rather easy. It generally follows the pseudo-code of Fig. 6. The concept is simple. It continues to find the 'not generated machine code' inside set M and output the testing program to test the code. After the machine code is outputted, the machine code is removed from M and the algorithm iterates until the set M is empty. The efficiency is O(N) where N is the number of instructions.

```
Set machine code set M as all the legal machine codes
Set output set O as empty
while (M is not empty) do
{
    Ouput header lines of the test program
    select m∈M, M=M\{m}
    Output m op testing lines
    Output tail lines
    O=O∪{m}
}
```

Fig. 6. pseudo-code for non-pipeline μC SW-EMI testing program auto-generation algorithm



```
Machine Code Sequence
(test program):

Loop:
      I/O toggle
              nop
              ...
              nop
       target instruction
              nop
              ...
              nop
goto  Loop
```

Fig. 7. Testing program example for μC SW-EMI characterization

The generated testing program by Fig. 6 is something like Fig. 7. The program is divided into 3 parts: the header, the body, and the tail of the program. The header includes an IO toggle code for identifying the beginning of the loop start and numbers of 'nop' (no-operation) code. The body of the program is the target instruction to be characterized, such as add (addition), sub (subtraction), etc. The tail of the program is a back jump code to form an infinite loop.

III. The Proposed Automatic Test Pattern Generation Algorithm

A. *SW-EMI consideration for Pipelined μC*

A 2-stage pipelined μC takes 2 machine codes inside the μC during execution. Thus, the testing program should consider the machine code sequence to characterize the whole SW-EMI behaviors.

B. *The auto-gen algorithm which reduces test pattern number without reducing test coverage for 2-stage pipeline μC*

The proposed algorithm is a re-arrangement of the machine code testing program which will reduce the test pattern number from the permutation of all patterns to some combination testing patterns without reducing test coverage.

Consider a 2-stage pipelined μC with only 3 machine codes {a, b, c}, the total number of the testing program shall be the *permutation of the individual testing patterns*. In this example, totally 9 programs are needed: a-a, a-b, a-c, b-a, b-b, b-c, c-a, c-b, and c-c. Here, the test program 'x-y' means the machine code sequence inside the testing program should be x first and y afterward besides the header and tail lines. All of the testing patterns are.

For a μC with 50 machine codes – it is a very simple μC, generally, a RISC μC like ARM has more than 200 machine codes. The number of testing programs can be up to 50x50=2500! This is not only a wasting of storage resource for the testing programs but also the testing time is too long for SW-EMI characterizations.

Take the 3-instruction μC as an example again. If the program is written as (a-a)-(a-b)-(a-c)-…(c-c) (the bracket is for human reading), the testing program can test all the a-a, a-b, b-a, …, etc. in one single program. It will need 2xN^2 machine codes to do the SW-EMI characterization.

If we look carefully that there are some repeating testing sequences inside the testing sequence. For example, (a-a)-(a-b) … needs only a-a-b-… will do the same characterizations. If all the machine codes are carefully arranged, a program like: a-a-b-a-c-b-b-c-c-a will do all the 9 tests. The number of the machine code inside the testing program can be 9 instead of 18 – half of the machine codes are reduced. For a μC with many machine codes, the time saving for SW-EMI characterization is large.

The algorithm is described as follows. Assume there are n machine codes {1..n}, the arrangement of the machine code in one testing program is shown in Fig. 8.

$$
\begin{array}{l}
1_{1\ 2\ 13\ \cdots 1\ n} \\
2_{2\ 3\ 2\ 4\ \cdots 2\ n} \\
\quad\vdots \\
k_{k\ k+1\ k\ k+2\ \cdots k\ n} \\
\quad\vdots \\
n-1_{n-1\ n} \\
\quad n_{1}
\end{array}
$$

Fig. 8. The proposed SW-EMI test pattern automatic generation algorithm

The big and small number in Fig. 8 is just for reading. According to Fig. 8, the actual permutation of a five machine code μC {1,..5} is arranged as: 1-1-2-1-3-1-4-1-5-2-2-3-2-4-2-5-3-3-4-3-5-4-4-5-1. The testing program's sequence can be found that all 5x5=25 patterns are tested by 25 machine codes only.

IV. Experiment Results

The measurement setups and test boards are designed according to the IC-cEMI measurement standard (IEC 61967-4) [3]. The oscilloscope is R&S RTO1002 (Fig. 9). The testing boards are shown in Fig. 10 and the testing program is shown in Fig. 3.

The target μC contains 67 machine codes to be characterized. The final testing program is totally 4489 (67^2)

testing codes inside the program. Comparing to the original 8978 machine codes, the testing program length is greatly reduced (TABLE I).

TABLE I. Machine code reduction comparison

	Original	Proposed	reduction
Machine code count	8978	4489	50%

When the testing program is running, the instruction waveform is recorded into the oscilloscope. The oscilloscope has 5G sample/sec and the memory depth is 1G samples. Due to the memory depth, the program has to be divided into fragments with the size of 200 machine codes.

Fig. 9. Oscilloscope

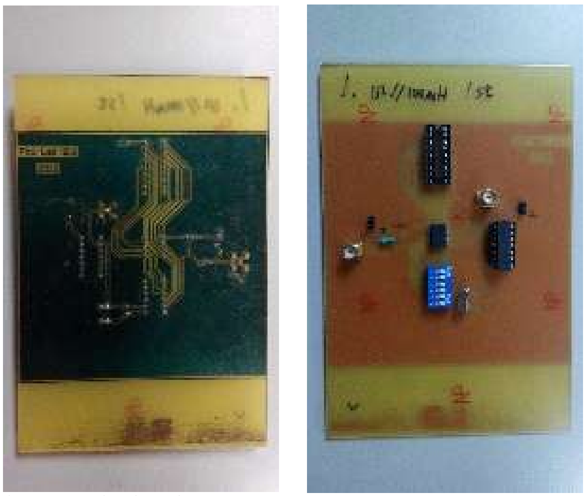


Fig. 10. Testing boards

TABLE II. Time saving during SW-EMI measurement

	Original	Proposed	reduction
Time needed (min)	230	450	~3.66 Hour

This means the proposed algorithm generates about 23 testing programs while the original algorithm generates 45 testing programs. Since the programs are downloaded, measured, collected individually and manually, less number of testing programs is desirable. In this case study, every program can be measured at about 10 minute's interval in our lab. The total time saving is very large (TABLE II, about 4 hours saved).

According to [1], the PDN of different machine code can be analyzed through the proposed algorithm and other procedures described in Fig. 1. It is found that the internal impedance (IntZ) of the target μC is time-varying.

V. Conclusion

Time-domain microcontroller (μC) electromagnetic interference (EMI) measurement is an essential part for software-related EMI (SW-EMI) modeling. To make a SW-EMI model, several signal process procedures should be done for the measure results. One of the procedures is the time-domain waveform test pattern generation. Due to the SW-related issues, the test patterns must go through all of test programs of the target μC machine codes (instructions). When the number of instruction (n) and the pipeline stage (m) get higher, the test pattern permutation size grows with n^m speed. However, an efficient algorithm for automatic test pattern generation for SW-EMI is not available.

This paper proposes an efficient test pattern generation algorithm for SW-EMI modeling of a 2-stage pipelined μC. This algorithm can reduce the test pattern number from the permutation of all test patterns to only some combinations of the test patterns without reducing model pattern coverage. The generated test pattern size is reduced 50% by the proposed algorithm. In our Lab, this means 3~4 hours SW-EMI measurement and characterization reduction.

Acknolowdgement

This work was technically and financially supported by Bureau of Standards, Metrology and Inspection (BSMI), Taiwan, Republic of China.

References

[1] Shih-Yi Yuan and Shry-Sann Liao, “Automatic Conducted-EMI Microcontroller Modeling,” 9th International Workshop on Electromagnetic Compatibility of Integrated Circuits (EMC Compo) Dec. 15-18, Nara, Japan, 2013.

[2] IEC 62433 "Models of Integrated Circuits for EMI behavioral simulation," [Online] http://www.iec.ch

[3] IEC 61967 "Integrated circuits - Measurement of electromagnetic emissions, 150 kHz to 1 GHz," [Online] http://www.iec.ch

[4] S. Y. Yuan, H. E. Chung, and S. S. Liao, “A Microcontroller Instruction Set Simulator for EMI Prediction,” IEEE Trans on EMC, vol. 51, pp. 692-699, 2009.

[5] Shi-Yi Yuan, Wei-Yen Chung, Cheng-Chang Chang, Chiu-Kuo Chang, "Software-related EMI Behavior of Embedded Microcontroller," *2014 IEEE International Symposium on Electromagnetic Compatibility*, Raleigh, NC, USA, Aug. 3-8, 2014.

[6] Shih-Yi Yuan, Jiun-Jia Huang, Chia-Yuan Hsu, Shry-Sann Liao, Chi-Chin Tang, and Haw-Yu Wu, "IC-EMC Model Extension Based on Internal Impulse Response Function," Asia-Pacific EMC-Symposium (APEMC), Singapore, May 21-24, 2012.