WCET Analysis of Automotive Buses using WCC

Dominic Oehlert, Heiko Falk
Institute of Embedded Systems
Hamburg University of Technology
Hamburg, Germany
Email: {Dominic.Oehlert,Heiko.Falk}@tuhh.de

Abstract—Bus systems build the backbone of the communication systems in modern cars. This leads to the fact that the effect of these bus systems have to be included during the WCET analysis of software running on these platforms. This paper shows how two bus systems commonly used in the automotive domain can be included into the WCET analysis using the WCC.

I. INTRODUCTION

Modern vehicles contain complex heterogeneous multilevel communication structures. These interconnection networks are necessary to connect the numerous sensors, electrical control units (ECU) and actuators. Such networks typically do not consist of one single bus, but different bus types which are connected using gateways. Each bus type has its own specific characteristics, varying in its transmission speed, arbitration policy, maximum number of users, maximum bus length and many more. The actual choice of the bus type of a certain node is often a trade-off between the financial costs (as a more complex bus is usually more expensive per node) and its functional and temporal requirements.

Programs which are executed in hard real-time systems stand in the center of several crucial requirements in order to guarantee a certain safety. One of these aspects is the Worst-Case Execution Time (WCET) of a program. The WCET of a program describes the maximum time the program needs from the start until its termination. This WCET has to be lower than given boundaries (which are typically specified by the environment) in order to ensure the correct behavior of the system.

As the interconnection network greatly influences the execution time of a program, it obviously is also a vital factor of the WCET of a program. Especially the transmission rates, arbitration policies and intermediate gateways can influence the WCET of a program in positive or negative ways. Additionally, also concurrent bus users can influence the timing if the arbitration policy does not fully follow a fixed time division multiple access (TDMA) scheme.

The WCET-aware C Compiler (WCC) [1] is a compiler framework which features several WCET-centered, as well as regular average-case execution time (ACET)-centered optimizations. Beside its interface to the AbsInt aiT [2] WCET analyzer, it also includes an own WCET analyzer by Kelter [3]. This analyzer framework is capable of deriving sound upper timing boundaries for ARM7TDMI multicore systems. In the following, we show how systems using common buses in the automotive domain can be analyzed using the WCC.

II. WCET ANALYZER

The WCET analyzer by Kelter follows the typical steps of a static timing analysis (STA):

1) Control flow graph recreation based on the disassembly
2) Value analysis of the registers
3) Microarchitectural analysis
4) Path analysis

In contrast to the majority of other WCET analyzers, it also includes a dedicated timing model of different bus types during the microarchitectural analysis. It features the timing analysis of up to 8 concurrent ARM7TDMI processors sharing a bus with one of the following arbitration policies:

- Fair Arbitration (Round Robin): The arbiter rotates the highest bus priority after each bus access.
- Static Priority: Every bus master has a fixed priority according to which the arbiter grants accesses.
- TDMA: The arbiter holds a schedule consisting of time slots of which each one has an exclusive owner.
- Priority Division: A generalization of TDMA. For each time slot, each bus master is assigned a unique priority. Additionally, bus masters given a priority of 0 are excluded from the arbitration, enabling the emulation of "classical" TDMA.

All mentioned bus arbitration policies are configurable in terms of timing properties and their characteristics. Using these basic featured arbitration schemes, it is possible to analyze systems featuring commonly used buses like Controller Area Network (CAN) or FlexRay if configured accordingly. Since the analyzer is tightly coupled within the compiler, detailed knowledge (e.g., loop bounds, indirect jump targets) derived from, e.g., the code selection, can be exploited.

III. EXEMPLARY ARCHITECTURES

An exemplary system architecture shown in Figure 1. Each ECU has a private instruction and data memory, containing the whole program code and private data. The system has four
bus masters, each connected to the shared bus. Additionally, a shared memory is attached to the bus which can be used for communication or shared data. Such a system architecture can be adjusted using configuration files of the WCC, enabling a versatile analysis.

A. Controller Area Network

The CAN communication protocol highly resembles a static priority bus arbitration policy. In case multiple bus masters try to transmit on the bus concurrently, the master with the highest priority will eventually win the arbitration. According to the CAN high-speed specification [4], the maximum transmission rate is 1 Mbps.

To reflect the CAN arbitration scheme, the arbitration policy of the WCET analyzer described in Section II is set to Static Priority. All CAN messages are assumed to have a fixed payload length. Additionally, a fixed timing overhead is added to every access to account for the transmission of the arbitration field, control field, etc. Depending on the environment to analyze, two possibilities exist to handle stuff bits: 1) Assume that all data to be sent is already encoded accordingly, such that no stuff bits are inserted anymore. 2) Assume the worst case, such that a frame of $n$ bits gets the timing of $(n+\lceil \frac{n}{4} \rceil)$ bits (as in the worst case after every fourth bit a stuff bit will be inserted; special fields are neglected here for the sake of simplicity). Timing differences between the transmission rate of the CAN bus and the clock frequency of the microcontroller can be adapted using an adjustable clock ratio inside the configuration file.

B. FlexRay

An exemplary FlexRay communication cycle is depicted in Figure 2. It has four slots in its static segment, one slot dedicated for each bus participant. Subsequently, the exemplary communication cycle has four minislots in its dynamic segment (one for each bus participant). The symbol window and network idle time slots follow accordingly. FlexRay features a bit rate of up to 10 Mbps according to its protocol specification [5].

We model a FlexRay communication cycle using the internal WCET analyzer of the WCC as a Priority Division arbitration policy. Currently, only the passive bus topology with a single-channel setup can be analyzed. The exemplary configuration is shown in Figure 3. The slots of the static segment are modeled as equally sized TDMA slots, resulting in the identical behavior and characteristic as the original segment. The dynamic slot is modeled using the Static Priority arbitration policy. Referring to Figure 2, the exemplary dynamic segment is fixed to accommodate a maximum of one actual message being sent. This circumstance is modeled as a single static priority slot, which is arbitrated at the very beginning and has a static length which is equal to the maximum length of the dynamic segment. This is slightly pessimistic due to two aspects: 1) The dynamic segment has a fixed length. 2) Messages which would become ready between the actual start of the dynamic segment and their dedicated minislot are blocked until the next cycle. Nonetheless, it allows a safe analysis with a minor pessimism.

The symbol window and network idle time are not required to be modeled in a detailed way since they are simply modeled as blocking slots with a fixed length, resembling the same timing as in the FlexRay communication cycle.

The FlexRay communication cycle shown in Figure 2 is just one illustrated example. Communication cycles with different parameters (especially regarding the dynamic segment) can also be modeled safely, yet with a few modification in the configuration.

IV. CONCLUSION

By configuring the internal WCET analyzer of the WCC accordingly, it is possible to analyze architectures featuring field buses which are commonly used in the automotive domain. This enables the compilation, analysis and potential WCET-aware optimizations to be executed by a single tool, allowing the plethora of information gathered at each step to be exploited by another.

In the presentation we will show a sample evaluation for both presented bus architectures using bundled benchmarks from the MRTC suite [6].

REFERENCES