CAN frame time-stamping - supporting AUTOSAR time base synchronization -

Florian Hartwich, Robert Bosch GmbH

The ever increasing number of electronic control units in modern cars, connected by networks to form a distributed computing system, led to the development of an open industry standard for automotive E/E architectures, AUTOSAR (AUTomotive Open System ARchitecture). It standardizes a layered software architecture, interfaces, APIs, and a runtime environment that allow a modular design method where re-usable software components may be transferred inside the system. The execution of software tasks that are distributed between different ECUs requires that the ECUs share a common time base. AUTOSAR defines methods how to synchronize the time bases between ECUs that are connected to the same network.
The new CiA 603 standard specifies a hardware time-stamping concept to be imple-mented in future CAN controllers. This concept is compatible with the AUTOSAR syn-chronization method, allowing to use both software and hardware time stamping in the same network. The hardware time stamping is independent of interrupt response times and results in higher accuracy of the time base synchronization. This paper de-scribes the new time-stamping concept and its implementation into CAN IP modules.

Introduction

In currently existing AUTOSAR systems, time base synchronization is implemented without dedicated hardware support. When the new CiA 603 standard is imple-mented into CAN controllers, future ECUs will have this dedicated hardware support for time base synchronization. This paper describes which hardware-based time-stamping functions are needed to improve the accuracy of the synchronization in a way that allows the new ECUs to be inte-grated into existing systems.

AUTOSAR synchronization method

In AUTOSAR systems, time is generally represented as a 64 bit number. The actual time value in an ECU is given by adding the value of a 32 bit free running timer counter to the value of a 64 bit time base register. An ECU may be linked to several time domains, with different time base registers but sharing the same timer counter. Each time domain has one time master and several time slaves. The time master synchronizes the time slaves by propa-gating, over the communication network, the time base value to the time slaves.

A time base can be distributed between networks that are connected by a time gateway. The time gateway receives the time base as time slave from one network and propagates it as time master to the other networks. In the following only the synchronization of the time base over the CAN bus is regarded.

Synchronization over CAN

In the first step of the synchronization procedure, the time master saves the actual time T_0 at the beginning of the procedure in seconds-portion s(T_0) and nanosec-onds-portion ns(T_0), as well as the actual value T_0_C of its timer counter (a 32 bit number). The time master writes s(T_0) into the transmit buffer for the synchroni-zation message SYNC and requests the CAN controller to transmit SYNC. When the CAN controller has successfully transmitted SYNC, this event triggers the capture the actual timer counter value as Tx_Stamp. From this the time master cal-culates T_Tx, which is the time from s(T_0) to the end of SYNC's transmission: T_Tx = ns(T_0) + ns(Tx_Stamp - T_0_C).

In the second step of the synchronization procedure, the time master writes T_Tx (a 32 bit number representing nanoseconds) into the transmit buffer for the follow-up message FUP. The data of FUP is com-plemented by two additional bits that sig-nal whether there was an overflow of the timer counter or in the calculation of T_Tx. For the time master, the synchronization procedure ends when the CAN controller has transmitted FUP.

SYNC and FUP are transmitted using the same CAN identifier; additional coding in the data field distinguishes SYNC from FUP, identifies the time domain, and ena-bles error checking. While an ECU uses only one CAN identifier if it is time master for different time domains, the CAN proto-col requires that other time masters (of other time domains) on the same CAN bus use different identifiers.

A time slave starts the synchronization procedure at the reception of SYNC, which triggers the capture of its timer counter value as RX_Stamp and provides the se-conds-portion of the time master's T_0. The capturing of Tx_Stamp in the time master and Rx_Stamp in the time slaves is triggered in all nodes by the end of the same CAN data frame, SYNC. The different nodes see this event with a phase shift of less than one CAN bit time.

The time slaves enter the second step of the synchronization procedure at the re-ception of FUP. This message enables the time slave to calculate, based the value of its timer counter TC, the received s(T_0), and its Rx_Stamp, the actual time T_a: T_a = s(T_0) + T_Tx + ns(TC – Rx_Stamp).

Repeated synchronizations allow the time slaves to adjust their local clock speeds. It is not necessary to increment the timer counter in all nodes at the same speed, because in the SYNC or FUP messages, all time information is transformed into real time units, seconds in SYNC and nano-seconds in FUP.

Advantage of time-stamping in hardware

AUTOSAR's specification of time synchro-nization over CAN is based on synchroni-zation messages that trigger interrupts at frame transmission (time master) and re-ception (time slaves). The interrupt service routines capture and compare the values of free running counters and calculate, with the help of a follow-up (FUP) message, the actual time offset between time master and time slaves.

The accuracy of this method depends on the interrupt response times after the synchronization message. The synchrony between the time stamps Tx_Stamp and Rx_Stamp is worsened by latency jitter.

When the timer counters are captured in hardware, directly triggered by the CAN controllers, instead of being captured by the interrupt service routines, latency jitter is avoided and the accuracy of the syn-chronization is improved.

The purpose of the new CiA 603 standard is to specify, beyond the functions already specified in ISO 11898-1, which functions CAN controllers should provide to support the AUTOSAR synchronization method.

CAN time-stamping in hardware

In ISO 11898-1, time-stamping is specified for the support of ISO 11898-4, TTCAN. These time stamps are captured at the start of a frame and they are 16 bit num-bers, using the CAN bit time as time steps. In current AUTOSAR systems, time stamps are captured at the end of frames, by the message's transmission or reception interrupt service routines. They are 32 bit numbers, using smaller time steps.

The gradual, non-disruptive integration of new nodes with CAN time-stamping in hardware into existing systems requires that the hardware time stamps are also captured at the end of the frames. To achieve the necessary precision, the time stamps need to be 32 bit numbers, captured from timer counters with time steps of less than one CAN bit time. These fea-tures enable hardware-based time-stamping nodes to participate in the synchronization procedure with software-based time-stamping nodes in the same network.

CiA 603 specifies that time stamps are captured at the end of frames, when the frame becomes valid according to the CAN protocol. That is the last-but-one bit of the end-of-frame field for the received SYNC messages and the last bit of the end-of-frame field for the transmitted SYNC messages. These are the same conditions that trigger the message's transmission or reception interrupt flags. The one CAN bit time difference (plus the signal delay from the receiver's ACK to the transmitter) between the two triggers is well known and can be considered in the time slave's cal-culations.

Time stamps are captured from a free-running 32 bit wide counter that is incremented in steps of at least 1 ns and at most 1 μs; it counts upwards and overruns to zero. The counter may be inside the CAN controller or outside; its value can be read anytime by the software. Several CAN controllers may share the same timer counter.

It is not necessary to store a time stamp for each message transmitted on the CAN bus. The time master needs a time stamp only for the transmitted SYNC messages, its capture can be controlled by that message's transmit buffer configuration. A time slave also needs to store time stamps only for the SYNC messages, but storage is needed for two time stamps since the CAN controller's acceptance filtering cannot distinguish between SYNC and FUP messages that use the same CAN identifier.

In CiA 603, it is mandatory to provide storage for at least 2 Rx_Stamps and at least one Tx_Stamp, or at least 2 time stamps if storage is shared between them. In order to be able to support multiple time bases concurrently, it is recommended to provide at least four times the mandatory minimum storage. AUTOSAR systems may have up to 16 synchronized time bases.

Separate time-stamping unit

Not all existing CAN controllers support time stamping of messages. If they do, time stamps are usually 16 bit wide and are stored inside the message buffer structure. If the position is not configurable, the time stamps are captured at the start of frame.

Changing the width of the stored time stamps to 32 bits (half of a Classical CAN data field) would require restructuring and enlarging the CAN message storage area. The CAN driver software would need to be adapted to the new structure.

The solution to this problem is to implement the new hardware time stamping function not into the CAN controller itself, but into a separate module, a Time Stamping Unit TSU. The CAN controller is only minimally modified, keeping its controller host interface unchanged.

The interface between the CAN controller and the TSU can be kept simple. The CAN controller provides trigger signals to capture the time stamps and the TSU provides information that indicates which time stamps belong to which messages. If there is more than one CAN controller, they may share one TSU, otherwise each CAN controller is connected to a dedicated TSU.

The TSU has its own controller host interface CHI, to configure and control its function and to read the captured time stamps. The TSU may include the free running timer counter with an optional prescaler, alternatively, an external timer counter may be connected. The timer counter value may be cascaded from one TSU to the next and it may be used as time base for legacy time-stamping with less resolution.

The time stamps are stored inside the TSU in a circular buffer, addressed by a counter. The elements of the circular buffer can also be read by via the CHI. Each time a capture is triggered, the address counter is incremented; the counter overflows to zero. The address counter value is provided to the CAN controller where it is stored with the message buffer, instead of the 32 bit time stamp itself.

The number of time stamps stored in the TSU can be decided by a generic parameter, changing the size of the module. The interface signals of the TSU are, with the exception of the address counter width, not changed by the size of the circular buffer.

The TSU may optionally include software debug support, flags that show whether a time stamp register contains new data or whether unread data was overwritten.

controllers designs that already support (shorter) message time stamps, this counter value can be stored instead of the time stamp, generally for all messages or only for SYNC messages.

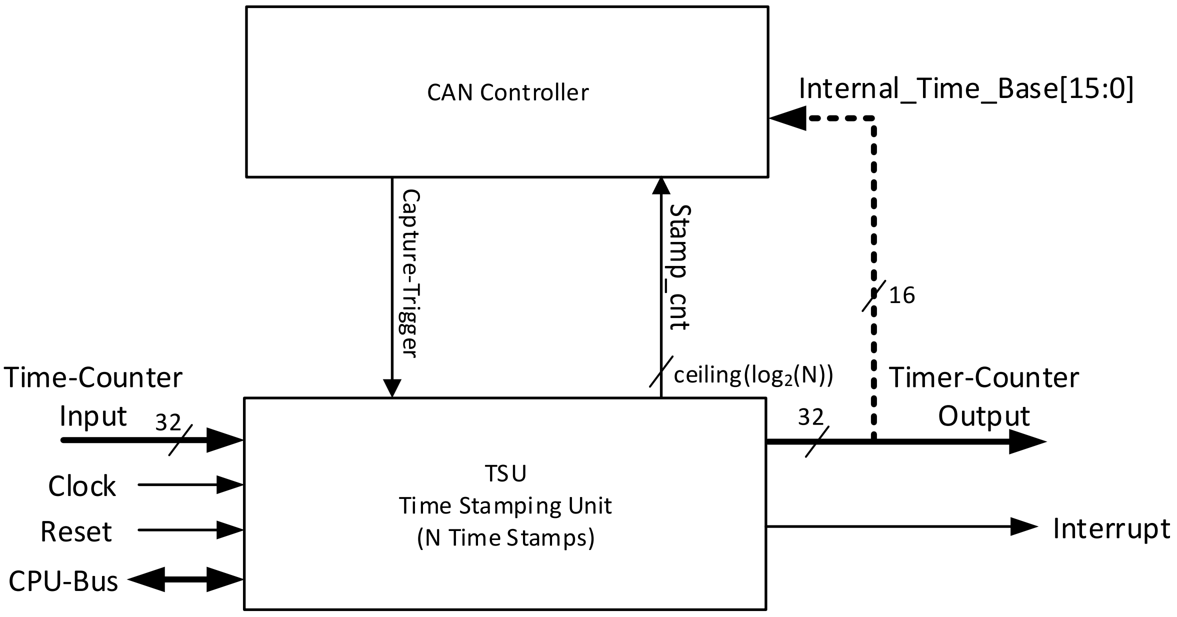

Figure 1: External Time Stamping Unit

An example for the interface between TSU and CAN controller is shown in Figure 1. The timer counter input vector is not needed when the TSU implements an internal timer counter. If several TSUs are cascaded, they share the same timer counter.

The TSU's interrupt output may optionally be used to signal the capture of a new time stamp or when a time stamp register was overwritten before it was read. It is not needed for time base synchronization.

The CAN controller activates the capture trigger for relevant messages, e.g. when a message is received that is recognized as SYNC message by CAN's acceptance filtering or when it is transmitted from a correspondingly configured transmit buffer.

The cyclic stamp counter value (three bits wide for storage of 8 time stamps) shows into which time stamp register the currently triggered time stamp is stored. In CAN

Conclusion

In CAN networks, the accuracy of AUTOSAR's method for time synchronization can be improved when the CAN controller captures time stamps directly triggered by CAN frames. The new standard CiA 603 specifies the additional features CAN controllers need to provide in order to support the AUTOSAR time synchronization.

Introduction of the additional features into existing CAN controller designs could require structural changes that would also affect low-level driver software.

The implementation of CiA 603 into existing and new CAN controller designs is simplified when the new features are moved into a separate time stamping unit. The changes to the CAN controller are minimal, reducing verification effort while allowing to adapt the time stamp number.

References

[1] CiA 603, Frame time-stamping, Requirements for network time management
[2] AUTOSAR Release 4.2.2, Specification of Time Synchronization over CAN

Florian Hartwich
Robert Bosch GmbH
AE/PJ-SCI
Postfach 13 42
DE-72703 Reutlingen
Tel.: +49-07121-35-2594
Fax: +49-0711-811-5142594
florian.hartwich¤de.bosch.com
www.can.bosch.com