

# Druckbetankung

## Effiziente Steuergeräte-Programmierung über FlexRay

In neuen Fahrzeuggenerationen kommt zunehmend der FlexRay-Bus zum Einsatz. Dadurch eröffnet sich die Möglichkeit, notwendige Software-Updates der Steuergeräte über FlexRay durchzuführen. Damit kürzere Programmierzeiten erreicht und schlussendlich Kosten eingespart werden, müssen Systemdesigner und Entwickler allerdings auf eine sorgfältige Konfiguration sowohl der Software als auch des FlexRay-Schedules achten.

Von Pascale Morizur und Peter Liebscher

Neben der bereits erfolgten Serieneinführung von FlexRay bei BMW stehen zurzeit zahlreiche weitere Anwendungen basierend auf AUTOSAR-kompatiblen Basis-Software-Modulen in der Evaluierung. Wichtige Voraussetzung für den Serieneinsatz ist, dass die Steuergeräte-Software schnell und einfach reprogrammiert werden kann – zu jeder Zeit und ohne Ausbau der Steuergeräte. Damit der Übergang von der Entwicklung zum Seriensteuergerät reibungslos funktioniert, sollte spätestens beim C-Muster ein geeigneter FlexRay-Flash-Bootloader in Verbin-

dung mit einem ODX-Datencontainer (Open Diagnostic data eXchange) verwendet werden. Im CAN-Umfeld ist diese Vorgehensweise bereits Standard, für FlexRay wird sie zunehmend eingesetzt. Um festzustellen, welche FlexRay-Konfigurationen zur besten Leistung des FlexRay-Flash-Bootloaders führen, ist eine detaillierte Analyse unabdingbar.

### ■ Grundlagen des Flashens

Flashen bezeichnet den Prozess der Übertragung von Software-Code und -Daten in den Flash-Speicher des Steu-

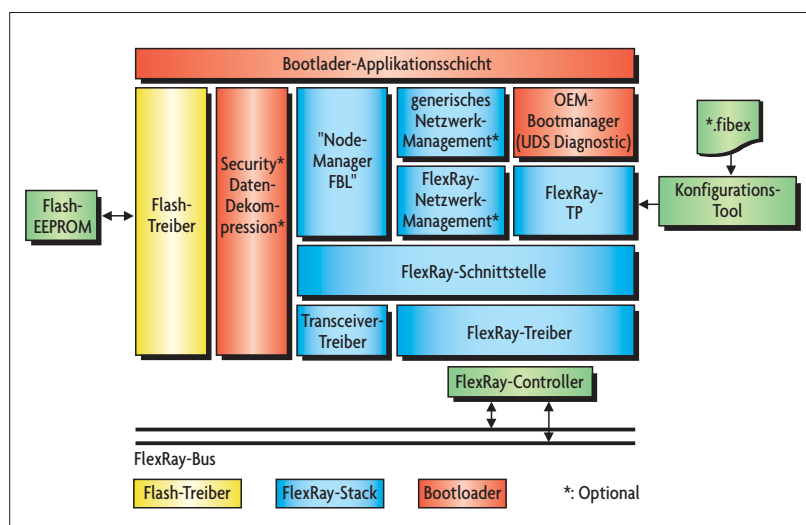
ergerätes. Im Gegensatz zum EEPROM-Speicher lassen sich Flash-Speicher nur blockweise löschen und beschreiben. Während der Entwicklung kann die Datenübertragung über einen Debugger erfolgen, wobei das Steuergerät ausgebaut werden muss. Im Unterschied dazu stößt in der Fertigung und im Servicebereich ein Flash-Tool den Flash-Vorgang über ein Diagnoseprotokoll wie UDS (Unified Diagnostic Services) an, der auf der Mikrocontroller-Seite vom FlexRay-Flash-Bootloader ohne Ausbau des Steuergeräts durchgeführt wird.

Beim Flashen müssen sowohl die Anforderungen des OEM als auch des Steuergeräte-Zulieferers berücksichtigt werden. Derzeit sind dies vor allem kurze Programmierzeiten, geringer Speicherbedarf und die Sicherheit vor Manipulation. Der FlexRay-Flash-Bootloader wird im Flash-Speicher des Mikrocontrollers als eigenständige, nicht löschbare Anwendung gespeichert: Entweder wird die Applikation oder der FlexRay-Flash-Bootloader ausgeführt. Der Vector-FlexRay-Flash-Bootloader [1] besteht aus konfigurierbaren Embedded Software-Modulen, die gemäß Bild 1 in folgende Funktionsblöcke unterteilt sind:

- ▶ Flash-Treiber,
- ▶ Bootloader,
- ▶ FlexRay-Stack.

### Flash-Treiber

Die Aufgabe des hardware-spezifischen Flash-Treibers besteht darin, die bisherigen Daten im Flash-Speicher zu löschen und die neu empfangenen Flash-Daten hineinzuschreiben. Er wird am Anfang des Flash-Vorgangs über das Bussystem ins RAM geladen und nach dem Ende wieder gelöscht.



! Bild 1. Vector-FlexRay-Flash-Bootloader – Übersicht über die Software-Module und funktionalen Blöcke.

(Alle Bildquellen: Vector Informatik)

### Bootloader

Auch der hardware-unabhängige Bootloader kontrolliert den Flash-Ablauf. Basierend auf dem für den CAN-Bus langjährig eingesetzten Vector-Flash-bootloader CANflb ist er in OEM-spezifischen Ausprägungen verfügbar. Sein grundsätzlicher Ablauf entspricht dem HIS-Standard (Herstellerinitiative Software). Der Prüfung der Zugangsberechtigung (Seed & Key) und dem Hochladen des Flash-Treibers über das Bussystem ins RAM folgen verschiedene Phasen (Bild 2): Nach dem Löschen der zu flashenden Bereiche empfängt das Steuergerät blockweise die Flash-Daten, dekomprimiert sie, übernimmt sie ins RAM und schreibt sie in den Flash-Speicher. Vor der endgültigen Freigabe erfolgt eine Prüfung auf vollständigen und korrekten Datenempfang sowie eine Verifizierung der Vertrauenswürdigkeit. Dieser Ablauf geschieht OEM-spezifisch im Modul „Bootloader-Applikationsschicht“ (Bild 1), wobei das Modul OEM-Bootmanager den Zugang zu den standardisierten UDS-Diensten ermöglicht und das Security-Software-Modul die Einhaltung geltender Sicherheitsmechanismen bewirkt.

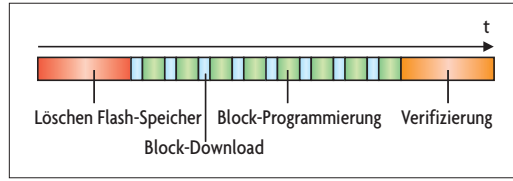


Bild 2. Phasen eines Flash-Vorgangs.

benötigten Funktionen. Durch diese Optimierungen ist ein geringer Speicherbedarf sichergestellt. Der Vector-FlexRay-Flash-Bootloader kann auch auf anderen AUTOSAR-kompatiblen FlexRay-Stacks aufsetzen.

Die Software-Module FlexRay-Interface, FlexRay-Driver und „Node Manager FBL“ aus dem FlexRay-Stack sorgen für die korrekte Umsetzung der Flash-Daten, die der Bootloader über den FlexRay-Bus empfängt. Da die Größe dieser Daten weit über die Grenze einer einzelnen FlexRay-Botschaft (max. 254 byte) hinausgeht, ist die Segmentierung in TP-Frames durch ein „FlexRay Transport Protocol“ (TP) erforderlich.

Optional ist der Einsatz eines FlexRay-spezifischen Network Managements, das während des Flash-Vorgangs alle Steuergeräte im FlexRay-Cluster wach hält und anschließend koordiniert in den Schlafzustand führt, damit sie nach dem Trennen des Flash-Tools problemlos wieder geweckt werden können.

### FlexRay-Stack

Der FlexRay-Stack ist für die Kommunikation mit dem Flash-Tool zuständig. Er ist AUTOSAR-kompatibel und enthält nur die für den Flash-Vorgang

### Flashen über FlexRay

Obwohl FlexRay eine hohe Datenrate von bis zu 10 Mbit/s bietet, führen nicht aufeinander abgestimmte Konfigurationen der Software und des FlexRay-Schedules zu einer schlechten Übertragungsrates beim Flashen. Daher müssen Bootloader, Transport-Protokoll und FlexRay-Schedule sorgfältig konfiguriert werden.

### FlexRay-Schedule

Der deterministische FlexRay-Bus unterliegt einem festen Schedule (Zeitplan), der sich nach erfolgter Synchronisation, wie in Bild 3 dargestellt, zyklisch in Runden wiederholt. Dieser Sche-

dule wird in der Designphase festgelegt, ist in einer FIBEX-Datei definiert und gilt für alle am FlexRay-Bus angeschlossenen Steuergeräte. Eine Runde besteht aus bis zu 64 Zyklen. Jeder Zyklus teilt sich in das obligatorische statische und das optionale dynamische Segment auf. Die Segmente wiederum sind in Slots unterteilt, in denen Frames gesendet und empfangen werden. Alle Frames in statischen Slots sind gleich lang und werden immer zum festen Zeitpunkt übertragen. Die dynamischen Frames dagegen haben eine individuelle Länge, die Übermittlung erfolgt nur bei Bedarf.

Derzeit ändert kein OEM zum Flashen den FlexRay-Schedule, weil dazu ein Neustart des kompletten FlexRay-Clusters nötig wäre. Daher erfolgt der Flash-Vorgang im Rahmen des normalen FlexRay-Schedules. Die dazu verwendeten TP-Frames werden entweder im statischen oder im dynamischen Segment übertragen. Der Transfer im statischen Segment hat den Nachteil, dass die meist auf die Bedürfnisse der Applikation abgestimmte Frame-Länge fest ist. Zudem ist auch die Zahl der FlexRay-Slots, die für das Flashen übrig bleiben, eingeschränkt.

Auch im dynamischen Segment ist die Anzahl der FlexRay-Slots, die für das Flashen genutzt werden können, durch die Konfiguration eingeschränkt. Jedoch ist hier die Größe jedes gesendeten Frames flexibel. Beispielsweise stehen bei einer Aufteilung von 60 Prozent statischem und 40 Prozent dynamischem Segment bei einer typischen Zykluslänge von 5 ms den dynamischen FlexRay-Slots max. 1,9 ms (ohne „Network Idle Time“ von typischerweise 100  $\mu$ s) zur Verfügung. In dieser Zeit lassen sich verschiedene TP-Frames mit unterschiedlicher Länge mit Sicherheit übertragen, etwa vier TP-Frames mit je 254 byte oder zwölf TP-Frames mit je 32 byte – jeweils inklusive Transport-Protokoll-Header. Für weitere dynamischen Frames bleiben bei diesen Konstellationen etwa 780  $\mu$ s übrig.

### Transport Protokoll und Download-Raten

Das Flash-Tool ermittelt mit dem UDS-Diagnose-Dienst „Request Down-

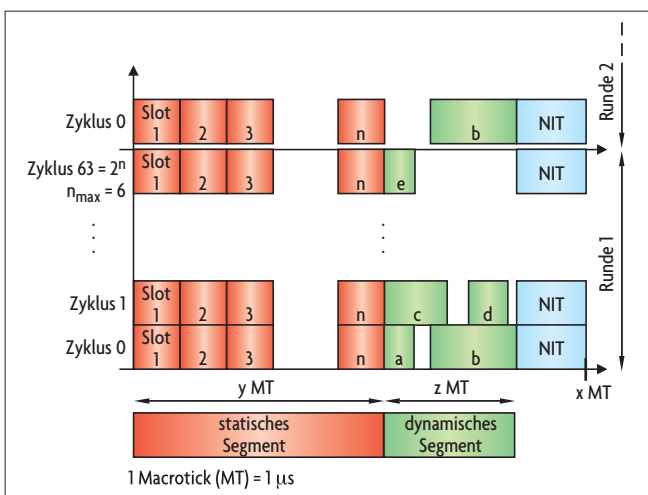


Bild 3. FlexRay-Schedule.

load“ die maximale Blockgröße des betreffenden Steuergerätes und unterteilt anschließend die zu übertragenden Daten in einzelne Blöcke. Bestimmt wird die Größe der Blöcke zum einen von der für die Programmierung verfügbaren RAM-Puffergröße und zum anderen vom betroffenen Flash-Speicher, der eventuell nur in 16-byte-Bereichen programmiert werden kann. Erst dann segmentiert der „Transfer Data“-UDS-Diagnosedienst jeden Block in FlexRay-Frames. Die Verwaltung der dabei entstandenen Frames erfolgt durch einen Header, der zusätzlich zu den Nutzdaten gemäß dem ausgewählten Transport-Protokoll innerhalb jedes FlexRay-Frames mit übertragen wird. Seine Größe ist vom Transport-Protokoll und dessen OEM-spezifischer Umsetzung abhängig. Grundsätzlich ergeben lange Frames ein günstiges Verhältnis von Nutzdaten (Payload) zu Protokoll-Header.

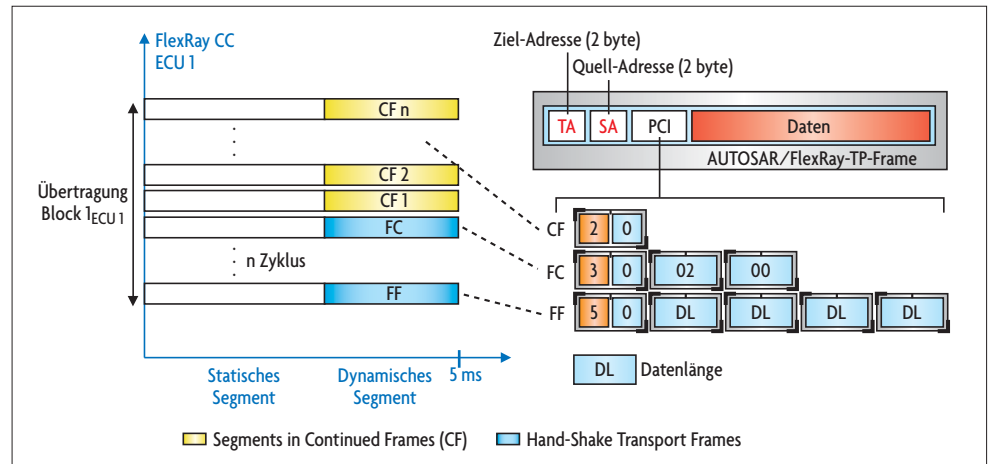
**Bild 4** zeigt den prinzipiellen Ablauf der Übertragung von Flash-Daten im „Transfer Data“-Dienst durch das Transport-Protokoll. Pro Block kommen folgende TP-Frames zum Einsatz:

- ▶ Das Flash-Tool sendet einen First Frame (FF) über das TP an das Steuergerät, um einen Kommunikationskanal aufzubauen. Dieser enthält die Angabe der Blockgröße sowie erste Nutzdaten.
- ▶ Darauf antwortet das Steuergerät möglichst im nächsten FlexRay-Zyklus mit einem Flow Control (FC), der die Blockgröße bestätigt. Eine ineffiziente TP-Implementierung verlängert die Wartezeit auf den FC-Frame, was die Leistungsfähigkeit des FlexRay-Flash-Bootloaders beeinträchtigt.
- ▶ Es folgt der Transfer der restlichen Nutzdaten des betroffenen Blocks mit sequenziell übertragenen Consecutive Frames (CF).

Gegebenenfalls wiederholt sich je nach Konfiguration der Austausch von FC-Frames und CF-Frames ohne weitere FF so lange, bis alle Blöcke übertragen wurden.

Zwei TP-Standards stehen momentan zur Verfügung: ISO/TP (ISO 15765-2) und AUTOSAR/FlexRay-TP V2.1. Des Weiteren ist ein ISO/FlexRay-TP (ISO 10681-2) in Vorbereitung.

Das ISO/TP ist im CAN-Umfeld weit verbreitet und schreibt für die TP-



**Bild 4.** Prinzipieller Ablauf des „Transfer Data“-Dienstes mit dem AUTOSAR/FlexRay-TP.

Botschaften eine Länge von genau 8 byte vor, davon 1 byte Header und eine maximale Blocklänge von 4095 byte. Sollte im Extremfall der FlexRay-Schedule nur einen Slot pro Zyklus (5 ms) vorsehen, so würde die reine Übertragungsgeschwindigkeit mit dem ISO/TP nur max. 1,4 kbyte/s betragen. Damit wären alle Vorteile der prinzipiell schnellen FlexRay-Übertragung hinfällig, denn die vergleichbare Download-Rate über CAN-Low-Speed (125 kbyte/s) beträgt 6 kbyte/s bzw. über CAN-High-Speed (500 kbyte/s) bis zu 17 kbyte/s.

Aus diesem Grund hat AUTOSAR für FlexRay ein besser geeignetes Transport-Protokoll definiert. Mit einer für die TP-Frames konfigurierbaren, jedoch zur Laufzeit nicht veränderbaren maximalen Länge von 254 byte, einem Header von im Mittel 5 byte pro CF-Frame und einer Blockgröße von maximal 4 Gbyte ermöglicht das AUTOSAR/FlexRay-TP 2.1 einen höheren Datendurchsatz. In **Bild 5** sind Download-Raten einer 128-kbyte-Nutzlast zu sehen. Sie wurden mit verschiedenen Frame- und Blockgrößen bei einer FlexRay-Zyklus-Länge von 5 ms, einem schnell-

len Hand-Shake-Verlauf (3 Zyklen) und ohne Acknowledge Frames ermittelt. Eine Blockgröße von 4082 byte mit 4 TP-Frames mit jeweils 254 byte/Zyklus im dynamischen Segment führen zu einem Download mit 110 kbyte/s und damit zu dem 6-fachen der CAN-High-Speed-Übertragung.

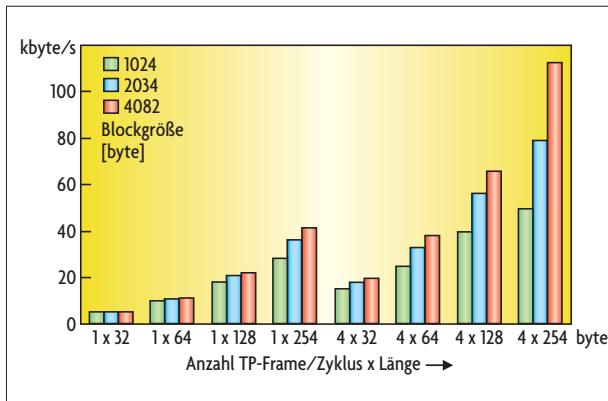


Bild 5. Maximale FlexRay-Download-Rate einer 128-kbyte-Payload mit dem AUTOSAR/FlexRay-TP.

### Optimierung des Flash-Vorgangs

Die richtige Konfiguration des FlexRay-Schedules und des Transport-Protokolls decken nur einen Teil der Optimierungsmöglichkeiten ab, die sich beim Flashen über FlexRay anbieten. Weitere Aspekte sollen darüber hinaus beachtet werden.

#### FlexRay-spezifische Optimierungen

Die Länge der TP-Frames und ihre Zuordnung zu FlexRay-Slots erfolgt in der Konfigurationsphase. Die Größe der Blöcke bestimmt dagegen das Flash-Tool während der Laufzeit in Abhängigkeit des vom Steuergerät im UDS-Diagnose-Dienst „Request Download“ übertragenen Maximums. Beide Parameter beeinflussen die Download-Rate und müssen daher sorgfältig ermittelt werden. Bild 6 zeigt, unter gleichen Bedingungen wie bei

Zu beachten ist aber, dass die Anzahl der TP-Frames, die für die Übertragung eines Blocks erforderlich sind, direkt von der Blockgröße abhängt. Wenn der letzte TP-Frame jedes Blocks nur wenige Nutzbytes und etliche Nullbytes enthält, so sollte für eine höhere Effizienz bei konstanter Frame-Länge die Blockgröße verkleinert werden. Dies ist im Bild 6 mit vier TP-Frames mit jeweils 128 byte/Zyklus deutlich zu erkennen.

Bild 7 zeigt für einen etwa 3 kbyte großen Block diesen Effekt bei größeren variablen Frame-Längen. In diesem Bereich erreicht eine TP-Frame-Länge von 244 byte die beste Leistung, da hier die Blockgröße kaum Einfluss auf die Download-Rate hat. Frames mit 254 byte würden die Download-Rate nicht verbessern. Beträgt hingegen die Frame-Länge 243 byte und wird die Blockgröße von 3090 auf 3106 byte erhöht, so sinkt die Download-Rate sogar um 14 Prozent.

#### Paralleles Flashen mehrerer Steuergeräte

Die Download-Raten der Bilder 4, 6 und 7 betreffen ausschließlich die Übertragung

Bild 5, den Einfluss unterschiedlicher Blockgrößen auf die Download-Rate für verschiedene TP-Frame-Längen. Prinzipiell resultiert bei FlexRay die größte Download-Rate aus den längsten Frames und dem größten Block.

der Flash-Daten über den FlexRay-Bus. Sie beinhalten keine Löscho- oder Programmierzeiten des Flash-Speichers. Diese dauern im Allgemeinen länger als die FlexRay-Übertragung und hängen vom Mikrocontroller, dessen Taktfrequenz und von der zum Flashen verwendeten Blockgröße ab. Die Tabelle zeigt zwei FlexRay-Mikrocontroller im Vergleich. Die Zeiten wurden bei der Programmierung einer 128-kbyte-Nutzlast in 1024 byte großen Blöcken ermittelt.

Beim Mikrocontroller A dauert zum Beispiel die Programmierung eines 2034 Bytes großen Blocks vier Mal so lange wie seine FlexRay-Übertragung (Bild 5). Die gesamte Flash-Performance könnte wesentlich verbessert werden, wenn das Flash-Tool die Programmier- und Löschozeiten ausnutzen würde, um gleichzeitig mehrere an FlexRay angeschlossene Steuergeräte parallel zu programmieren. Die Anzahl der erforderlichen FlexRay-Slots sollte sich jedoch hierbei nicht erhöhen. Voraussetzung dafür ist ein geeignetes TP, wie das zurzeit in Arbeit befindliche ISO/FlexRay TP.

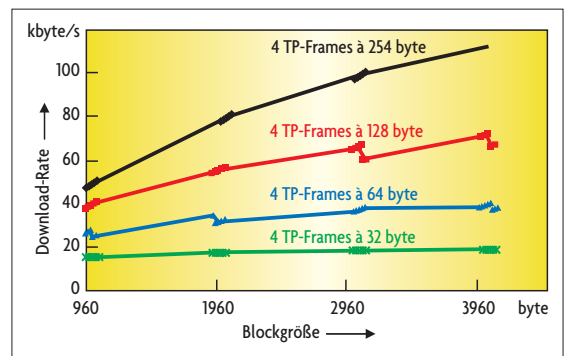


Bild 6. FlexRay-Download-Rate verschiedener TP-Frame-Längen mit dem AUTOSAR/FlexRay-TP.

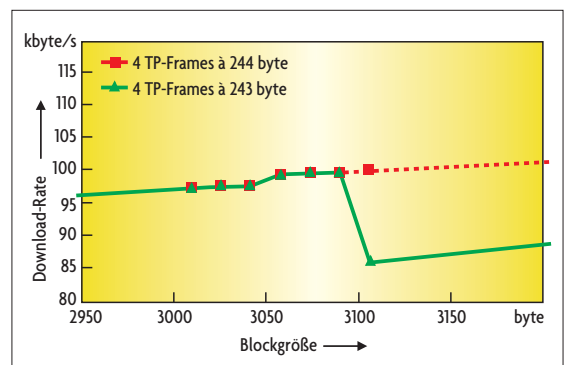


Bild 7. Einfluss von Blockgröße und TP-Frame-Länge auf die Download-Raten mit dem AUTOSAR/FlexRay-TP.

	Lösch-Rate (kbyte/s)		Programmier-Rate (kbyte/s)	
	4 MHz	16 MHz	4 MHz	16 MHz
Mikrocontroller A	43,2	43,2	15,6	15,4
Mikrocontroller B	111	121	42,7	80

### I Lösch- und Programmieraten von Flash-Speichern

#### Optimierung der Verifizierung

Damit die Daten während des Flashens der Steuergeräte vor einer möglichen Manipulation bewahrt sind, sorgen Zugangsschutzmechanismen für Sicherheit. Neben den bisher üblichen Methoden des „Seed & Key“-Verfahrens kommen zusätzliche kryptografische Funktionen wie RSA (Asymmetrisches Kryptosystem) zum Einsatz. Durch geheime Schlüssel wird eine Signatur innerhalb eines Trust-Centers ermittelt, welche dann zusätzlich im Flash-Vorgang gesendet wird. Der Bootloader berechnet die erwartete Signatur und vergleicht diese mit der übertragenen. Dieses Verfahren setzen einige Fahrzeughersteller bereits seit mehreren Jahren erfolgreich mittels des Vector-CANfl ein, woraus sich ein Quasi-Standard entwickelt hat.

#### ■ Anwendungsfälle

Der Reprogrammievorgang des Steuergeräts deckt nur einen Teilaspekt des gesamten Flash-Prozesses ab. OEMs und Zulieferer erstellen ihre Konzepte, wie sie in den Entwicklungs-, Test- und Integrationsphasen sowie in der End-of-Line-Programmierung und in der Werkstatt das Flashen anwenden. Die unterschiedlichen Anforderungen in diesen Prozessen erschweren eine Standardisierung. Um dennoch alle mit dem Flashen in Verbindung stehenden Teilaspekte bestmöglich zu verbinden, bietet ASAM das ODX-F-Format als Teil des ODX-Standards [2]. Es definiert einen Standard zur genanten Speicherung aller notwendigen Informationen in einem ODX-F-Container. Dabei werden reine Flash-Daten (Nutzlast), individuelle Metadaten, Datenformate, Wertebereiche und fertige, herstellerspezifische Flash-Jobs im XML-Format in Flash-Datencontainern abgelegt. Dies gewährleistet sowohl einen reibungslosen Austausch von Informationen

zwischen den Prozess-Teilnehmern als auch die Wahrung der Prozess-Sicherheit.

Ein weiterer Vorteil ist die Modularisierung von Flash-Daten, wie sie für das

nachträgliche Flashen von einzelnen Steuergeräte-Funktionen benötigt wird. Die Diagnose-Container (ODX-D) und die Flash-Container werden mit so genannten Autoren-Tools wie CANDelaStudio und CANDelaFlash von Vector Informatik aus den Diagnosebeschreibungen erstellt, wobei die Flash-Jobs in den ODX-D-Containern enthalten sind. Die als gemeinsame Basis verwendeten Diagnose-Beschreibungen mittels CDD- oder XML-Dateien erlauben die Generierung von redundanzfreien Daten und ermöglichen die Vermeidung von Inkonsistenzen.

FlexRay bietet eine Datenübertragungsrate von bis zu 10 Mbit/s, die beim Flashen jedoch nicht voll ausgeschöpft werden kann. Mit aufeinander abgestimmten Konfigurationen von Bootloader, Transport-Protokoll und dem FlexRay-Schedule ist es aber heute bereits möglich, sechs Mal schneller als über CAN zu flashen. Dies unterstützt Vector mit dem FlexRay-Flash-Bootloader, der die bereits für den CAN-Bus bewährten OEM-spezifischen Algorithmen enthält. Sollte der Flash-Job die Programmierzeit jedes einzelnen Flash-Blocks dazu nutzen, um parallel Daten zu weiteren Steuergeräten zu übertragen, so könnte man die gesamte Flash-Performance noch vervielfachen. Diese Steigerung ist sowohl beim Flashen über CAN als auch über FlexRay möglich, wobei FlexRay dazu ein spezielles, sehr flexibles TP erfordert.

In Zusammenarbeit zwischen ISO und AUTOSAR wird zurzeit an einem neuen ISO/FlexRay-TP-Standard gearbeitet, der die erforderlichen FlexRay-Slots auf ein Minimum beschränken und die Frame-Länge zur Laufzeit je nach Inhalt anpassen soll. Dadurch wird bei paralleler Programmierung von Steuergeräten die verfügbare Übertragungsrate optimal ausgenutzt. Dies ist besonders beim parallelen Flashen über FlexRay in Sub-Netz-

werke wie CAN oder LIN interessant. Ohne spezielles Fachwissen und Erfahrung gestaltet sich die Optimierung des Flash-Vorgangs schwierig. Für ODX und in Zukunft auch für FlexRay bietet Vector mit CANDelaStudio, CANDelaFlash, CANape, CANDito und CANDitoFlash eine vollständige, durchgängige Werkzeugkette für alle Aspekte des Flashens an. *sj*

#### Links

- [1] [www.vector-informatik.com/vi\\_flexray\\_flashbootloader\\_de.html](http://www.vector-informatik.com/vi_flexray_flashbootloader_de.html)
- [2] [www.elektroniknet.de/home/automotive/technik-know-how/uebersicht//test-entwicklungstools/richtiges-flashen-fuer-jede-aufgabenstellung](http://www.elektroniknet.de/home/automotive/technik-know-how/uebersicht//test-entwicklungstools/richtiges-flashen-fuer-jede-aufgabenstellung)



**Dipl.-Ing. Pascale Morizur**

studierte Physik-Elektronik an der Grande Ecole ICPI in Lyon (F). Nach ihrem Abschluss 1986 arbeitete sie 10 Jahre bei MAN-Nutzfahrzeuge in der Vorentwicklung im Bereich CAN, J1939 und Diagnose. Jetzt ist sie bei Vector als Product Management Engineer im Bereich FlexRay-Embedded-Software-Komponenten tätig.

[pascale.morizur@vector-informatik.de](mailto:pascale.morizur@vector-informatik.de)



**Dipl.-Ing. (FH)  
Peter Liebscher**

hat Nachrichtentechnik an der FH in Esslingen studiert. Seit 2002 betreut er als Business Development Manager bei der Vector Informatik GmbH die Produktlinie Embedded Software Components.

[peter.liebscher@vector-informatik.de](mailto:peter.liebscher@vector-informatik.de)