Ausgewählte Kapitel eingebetteter Systeme
TTP und FlexRay

Richard Membarth

14.06.2006

Inhaltsverzeichnis

1 Einleitung

1.1 Allgemein

Das Time-Triggered Protocol (TTP/C) und FlexRay sind zwei echtzeitfähige Kommunikationsprotokolle zur Verbindung elektronischer Module von verteilten, fehlertolleranten Systemen. Die Protokolle wurden mit dem Ziel entwickelt in sicherheitskritischen Systemen eingesetzt zu werden und harter Echtzeit zu genügen. Damit erfüllen sie den "Automotive Class C" Anforderungen der SAE[1]. Anders als das weit verbreitete Kommunikationsprotokoll CAN[2] sind TTP und FlexRay nicht ereignisgesteuert, sondern zeitgesteuert. Dadurch wird in Verbindung mit einem geeignetem Zugriffsverfahren eine deterministisches Verhalten erreicht.

1.2 Geschichte

TTP entstand an der TU Wien und wurde dort über 20 Jahre lang entwickelt, bevor vor kurzem die TTTech Computertechnik AG die Wartung und Weiterentwicklung von TTP und darauf basierten Lösungen übernahm. Das Haupteinsatzgebiet von TTP sind Steuersysteme im Automobilbereich, der Luftfahrt, in Industrieanlagen und in Antriebssystemen. So kommt TTP z.B. im Kabinendrucksystem des neuen Airbus A380 zum Einsatz oder in der Bahnhofssignalsteueranlage ELEKTRA 2 von Alcatel. Im Automobilbereich spielt das sogenannte X-by-Wire, das die Ersetzung von mechanischen Übertragungswegen durch elektronische Funktionen ohne mechanische Sicherungssysteme zum Ziel hat, eine große Rolle. Während bei dem in der Luftfahrt verwendete Fly-by-Wire der Steuerknüppel bereits seit den 70er Jahren vollständig mechanisch vom Stellmotor entkoppelt ist, wird heutzutage noch an entsprechenden Lösungen für Drive-by-Wire gearbeitet. Einige Autohersteller wie Audi, PSA Peugeot oder Renault haben sich zunächst auch für TTP als Kommunikationsprotokoll für ihre Drive-by-Wire Entwicklungen entschieden.

FlexRay wurde vom FlexRay–Konsortium entwickelt, welches 2000 von den Firmen BMW, DaimlerChrysler, Motorola und Philips mit dem Ziel gegründet wurde, eine neue Kommunikationstechnologie zu gründen, die den steigenden technischen Anforderungen an die Buskommunikation in Fahrzeugen gerecht wird, da TTP als nicht flexibel genug betrachtet wurde. Dennoch wurden die Grundprinzipien von TTP übernommen und darauf aufbauend FlexRay entwickelt, in das vor allem noch das ereignisgesteuerte Kommunikationsprotokoll ByteFlight von BMW mit einfloss. Bis 2004 schlossen sich alle anderen Automobilhersteller dem FlexRaykonsortium an, auch diejenigen, die sich zunächst für TTP aussprachen.

Im Gegensatz zu TTP befindet sich FlexRay noch in einem frühen Stadium der Entwicklung und wurde zum ersten mal im 2006 erschienenen BMW X5 verwendet, jedoch nicht im

[1] Society of Automotive Engineers
[2] Controller Area Network

ursprünglich angedachte Einsatzgebiet für sicherheitskritische Anwendungen in harter Echtzeit. Erste den Fähigkeiten von FlexRay angemessene Anwendungen sind erst im Jahr 2008 zu erwarten. Das extra für den Automobilbereich entwickelte FlexRay scheint sich auch dort gegen TTP durchgesetzt zu haben, wohingegen TTP in anderen Gebieten weiterhin vertreten ist und zum Einsatz kommt.

2 Aufbau und Funktionsweise von TTP und FlexRay

Da TTP und FlexRay in großen Teilen gleich aufgebaut sind, wird die grundsätzliche Struktur und Funktionsweise anhand von TTP erklärt, und nur dort, wo es Unterschiede gibt, die explizit genannt.

2.1 Architektur

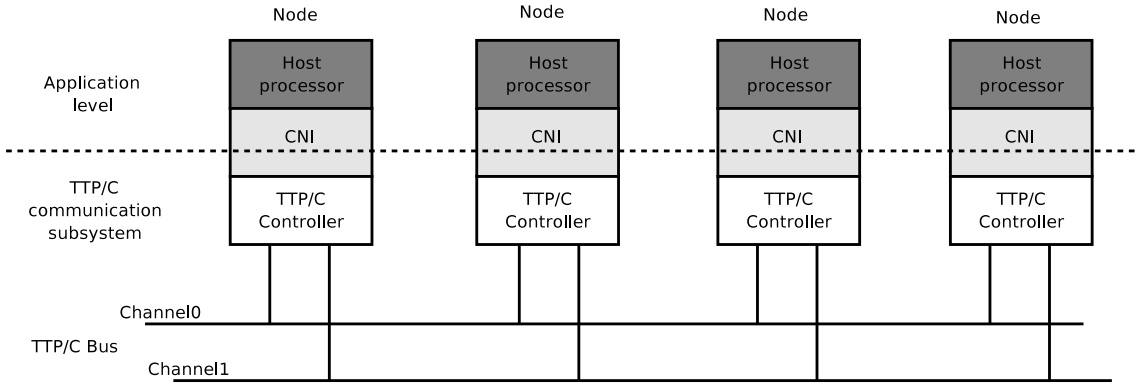

Abbildung 2.1: Aufbau eines TTP-Clusters (bei FlexRay analog)

Ein Kommunikationssystem, das über TTP bzw. FlexRay kommuniziert ist in Abbildung 2.1 dargestellt und wird auch Cluster gennant. Mehrere verteilte Knoten sind über einen TTP/C–Bus verbunden und können über diesen Nachrichten austauschen. Ein Knoten besteht dabei aus folgenden Elementen:

- TTP/C Controller: führt das TTP–Protokoll aus
- CNI[1]: Schnittstelle um Nachrichten mit dem Host–Prozessor auszutauschen (bei FlexRay CHI[2] genannt)
- Host Processor: Prozessor, auf dem verteilte Anwendungen laufen und die über TTP kommunizieren.

Bei TTP und FlexRay ist kein Übertragungsmedium explizit vorgegeben, so dass sowohl ein elektrisches als auch optisches Übertragungsmedium verwendet werden kann.
Als Kommunikationstopologien erlauben TTP und FlexRay Bus– und Sterntopologien, wobei

[1] Communication Network Interface
[2] Controller Host Interface

diese auch miteinander gekoppelt werden können. Die beiden Kommunikationskanäle werden jeweils getrennt behandelt, d.h. bei redundanter Datenübertragung muss der Host Processor überprüfen, ob bei beiden Kanälen auch die gleichen Daten angekommen sind. FlexRay erlaubt zusätzlich noch eine hybride Struktur, d.h. dass ein Kanal an einen Bus angeschlossen ist und der andere an einen Stern.

Bei TTP gibt es neben der hier betrachteten Variante TTP/C, die harter Echtzeit genügt, noch die abgespeckte Variante TTP/A. Diese bietet lediglich weiche Echtzeit und erfüllt nur die Anforderungen für Class A der SAE, da bei TTP/A kein eigener Prozessor für den TTP Controller vorgesehen ist und dieser mit dem Host Processor verschmolzen wird.

2.2 Medienzugriff

Da TTP und FlexRay zeitgesteuert sind, dürfen die einzelnen Knoten nur zu zuvor festgelegten Zeitpunkten auf das Übertragungsmedium zugreifen und Daten senden. Dazu ist es essentiell, dass alle Knoten eine gemeinsame Zeitbasis haben. Deshalb ist in beiden Protokollen ein fehlertoleranter Algorithmus integriert, der die Uhren der einzelnen Knoten synchronisiert und der gesamte Cluster somit die gleiche globale Zeit zur Basis hat. Die kleinste globale Zeiteinheit ist der sogenannte Makrotick, der abhängig von den einzelnen Knoten aus mehreren Microticks, der kleinsten lokalen Zeiteinheit besteht. TTP und FlexRay bieten aber auch die Möglichkeit die Uhren über eine externe Quelle zu synchronisieren.

2.2.1 TTP

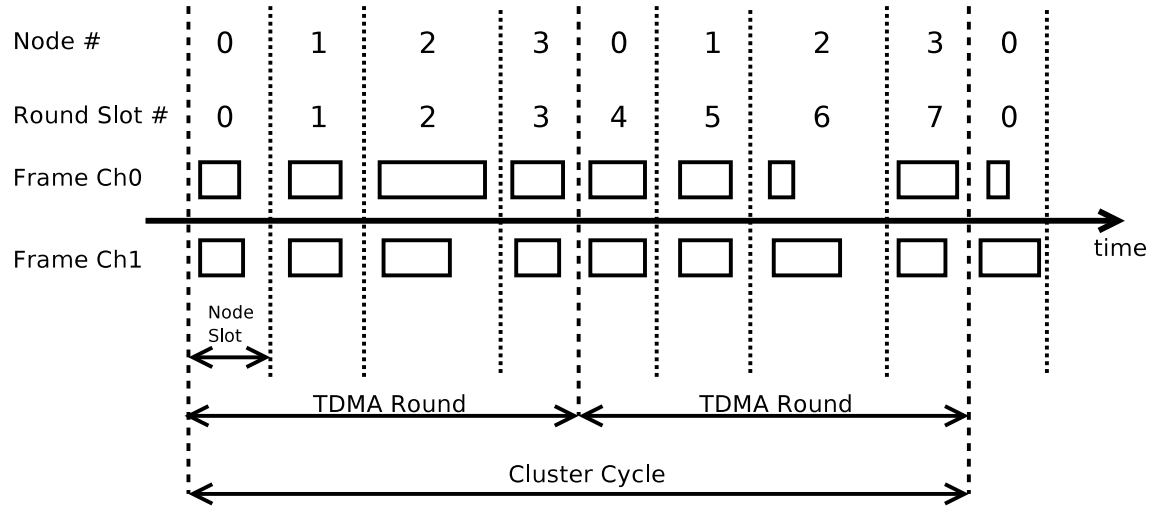

Abbildung 2.2: Media Access Scheme (TTP)

Das Medienzugriffsverfahren bei TTP ist das TDMA[3] wie es in Abbildung 2.2 dargestellt ist: In einer TDMA Runde (`TDMA Round`) kann ein jeder Knoten in einem ihm zugewiesenem Zeitschlitz (`Node Slot`) auf das Übertragungmedium zugreifen und dort auf beiden Kanälen (`Ch0` und `Ch1`) senden. Mehrere logisch zusammengehörige TDMA Runden bilden einen `Cluster Cycle`, welche periodisch hintereinander ausgeführt werden. Ein Knoten

[3]Time Division Multiple Access

muss jedoch nicht in jeder TDMA Runde senden, so ist es möglich, dass sich zwei Knoten (`multiplexed nodes`) einen Zeitschlitz in einem Cluster Cycle teilen und einer in der ersten Runde sendet und der andere in der zweiten. Knoten die niemals Daten senden, sog. `passive nodes` werden v.a. zum Monitoring verwendet.

2.2.2 FlexRay

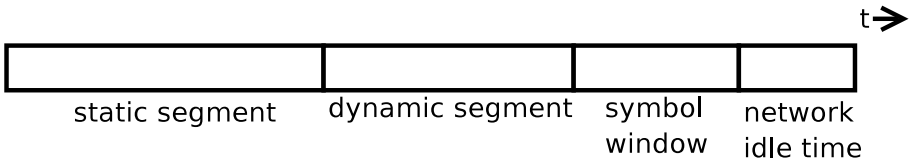

Abbildung 2.3: Aufteilung eines Communication Cycles in Segmente

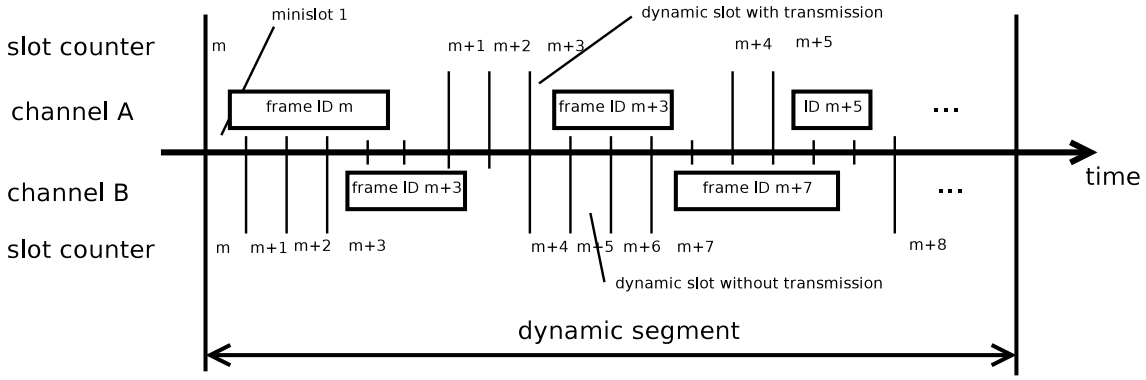

Abbildung 2.4: Struktur des dynamischen Segments

FlexRay benutzt eine Kombination aus TDMA und FTDMA[4] als Medienzugriffsverfahren. Ein Communication Cycle wird wie in Abbildung 2.3 dargestellt in vier Segmente unterteilt:

- statisches Segment: arbeitet analog zu dem von TTP verwendetem TDMA
- dynamisches Segment: arbeitet nach dem FTDMA verfahren (siehe unten)
- symbol window: reserviert für Netzwerkmanagemantfunktionen (z.B. WakeUp)
- network idle time: hier wird u.a. die Synchronisation der Uhren vorgenommen

Das dynamische Segment arbeitet bei FlexRay nach einem flexiblen TDMA, bei dem das Segment in Minislots aufgeteilt ist, die zur Übertragung genutzt werden können, wenn Daten vorliegen (vgl. Abbildung 2.4): Im voraus ist lediglich die Reihenfolge des Buszugriffs der einzelnen Knoten festgelegt, die Länge der Übertragung jedoch nicht. Dabei hat ein jeder Minislot eine minimale Länge. Wird bis zu einem bestimmten Punkt des Minislots keine Übertragung gestartet, wird der Minislot ohne Übertragung beendet und der nächste Minislot

[4]Flexible Time Division Multiple Access

beginnt. Das Ende eines Minislots mit Übertragung wird durch eine fetgelegte Zeit ohne Übertragung bestimmt. Ein Knoten darf dabei nur dann Senden, wenn seine `Frame ID` gleich dem `slot counter` ist.

Die minimale Länge des statischen Segments beträgt 2 TDMA–Slots. Dies ist notwendig um die Uhren zu syncronisieren, aus dem gleichen Grund muss das network idle time Segment immer vorhanden sein. Dagegen ist das dynamische Segment sowie das symbol window optional. Somit ist es auf der einen Seite eine rein auf TDMA basierte Kommunikation möglich, auf der anderen eine Kommunikation die das statische Segment lediglich zur Uhrensyncronisation nutzt und das dynamische wie ein eventgesteuerte Kommunikation mit priorisierter Reihenfolge.

Jedes Bit wird bei FlexRay über mehrere Microticks hinweg übertragen und der Empfänger speichert die einzelnen Bits und entscheidet per Mehrheitsentscheidung ob eine logische 0 oder 1 übertragen wurde. Dadurch kann trotz eines Rauschens auf der Leitung eine fehlerfreie Übertragung gewährleistet werden.

2.3 Frame Format

2.3.1 TTP

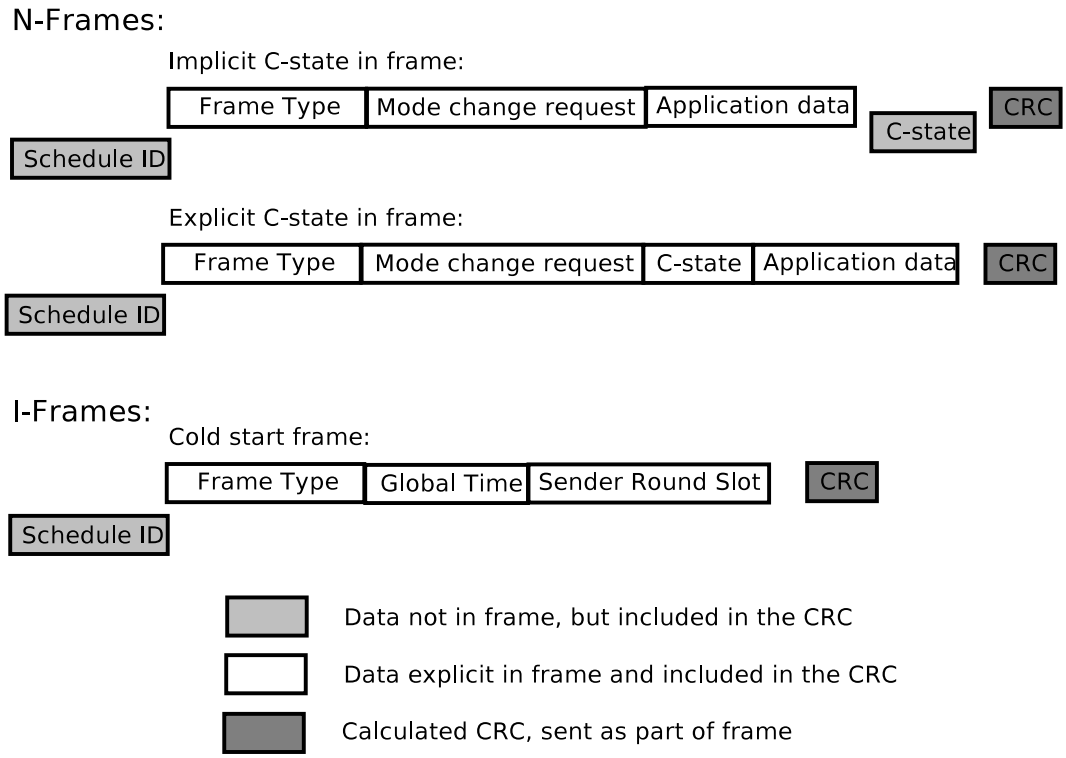

Abbildung 2.5: Frame Format (TTP)

TTP kennt zwei verschiedene Typen von Frames, die in Abbildung 2.5 dargestellt sind: den `I-Frame` (initialization bzw. `cold start frame`), der verwendet wird um einen TTP–Cluster zu starten (enthält Informationen, welche andere Knoten zum integrieren in den TTP–Cluster benötigen, wie z.B. die globale Zeit und die Position des Senders in der TDMA Runde) und die

`N-Frames` (normal frames) für den normalen Datentransfer. Die N–Frames beinhalten neben den eigentlich zu übertragenden Daten, ein `Mode Change Request Feld`, das den Antrag eines Knotens zur Änderung der Betriebsart beinhaltet, und den sogenannten Controller–State (`C-State`), der den aktuellen Zustand des Clusters aus der Sicht eines Knotens beschreibt und in allen Knoten gleich sein muss.

Zusätzlich werden alle Frames durch einen CRC geschützt. In die Berechnung des CRC fließt als Parameter die `Schedule ID` ein, die Clusterweit gleich ist, ein Controller kann somit auch nur innerhalb seines Clusters arbeiten. Bei den N–Frames unterscheidet man zusätzlich noch ob der C–State als Parameter mit in die Berechnung des CRC eingeht oder als Nutzdatum übertragen wird. N–Frames mit expliziten C–State werden benötigt, damit andere Knoten sich in einen bereits laufenden Cluster integrieren können, besitzen jedoch auch weniger Bandbreite für die restlichen Nutzdaten.

2.3.2 FlexRay

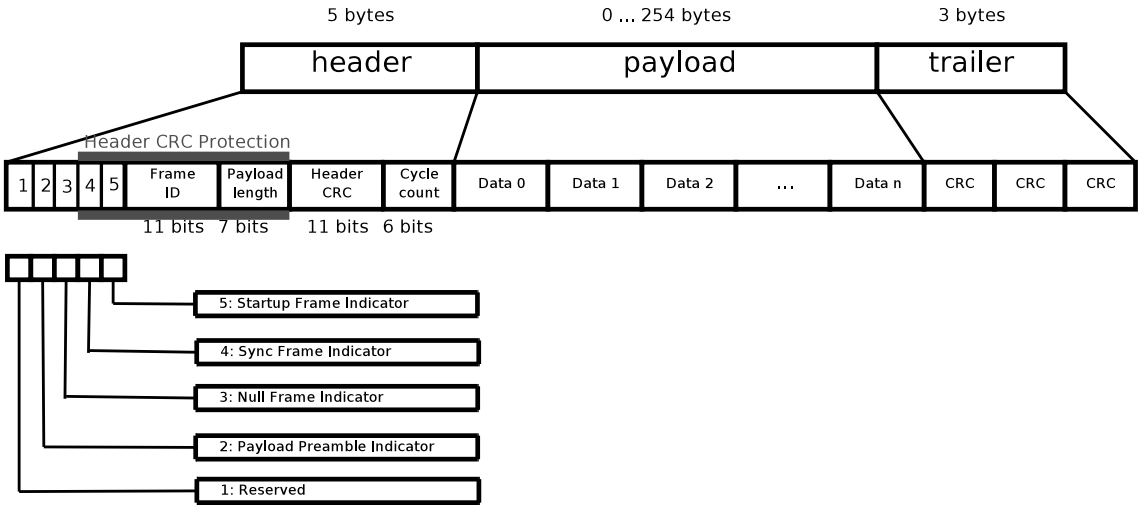

Abbildung 2.6: Frame Format (FlexRay)

Bei FlexRay gibt es im Gegensatz zu TTP nur einen Frame, der, wie in Abbildung 2.6 dargestellt, aus einem Header (5 bytes), Nutzdaten (0-254 bytes) und einem CRC im Trailer (3 bytes) besteht, also maximal 262 bytes lang ist. Der Header beinhaltet u.a. folgende Informationen:

- `payload preamble indicator`: Nutzdaten beinthalten Network Management Data (stat. seg) bzw. Message ID (dyn. seg.)
- `null frame indicator`: Frame beinhaltet keine Nutzdaten
- `sync frame indicator`: Frame soll zur Uhrensyncronisation verwendet werden
- `startup frame indicator`: Frame soll zum Starten des Clusters verwendet werden
- `payload lenght`: Länge der Nutzdaten in Words
- `header CRC`: CRC über payload length, frame ID, startup frame indicator, sync frame indicator

- `cycle count`: kennzeichnet aktuellen Cycle des Clusters

2.4 Host–Schnittstelle

2.4.1 CNI (TTP)

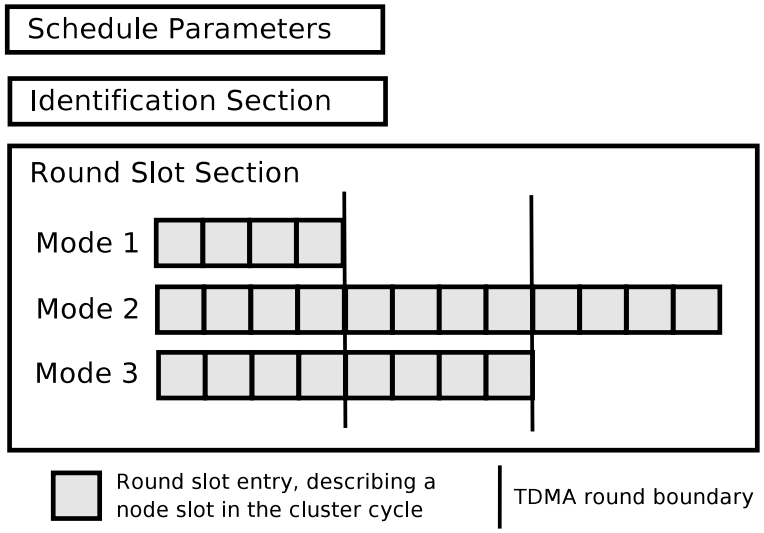

Abbildung 2.7: MEDL (TTP)

Das Communication Network Interface ist die Schnittstelle bei TTP zwischen dem TTP–Controller und dem Host–Processor, wobei das CNI in die drei Bereiche Status–, Kontroll– und Nachrichtenbereich unterteilt werden kann. Der Status– und Kontrollbereich sind normalerweise auf dem Controller lokalisiert und dienen dem Host dazu Informationen über den aktuellen Zustand des Protokolls zu gewinnen und um den Modus, in dem der Controller arbeitet, zu ändern. Der Nachrichtenbereich kann dagegen auch im Speicher des Hosts liegen. Von dort liest der Host die vom Controller empfangenen Nachrichten und speichert dort die Nachrichten ab, die vom Controller gesendet werden sollen.

Um die Kommunikation zu koordinieren wird die MEDL[5] (vgl. Abbildung 2.7) verwendet, die aus folgenden Bestandteilen besteht:

- Schedule Parameters: Notwendige Parameter um den Ablauf des Protokolls zu gewährleisten, den Cluster zu starten und einzelne Knoten in den Cluster zu integrieren.
- Identification Section beinhaltet eine eindeutige Bezeichnung der MEDL, die von externen Programmen oder dem Host verwendet werden kann.
- Round Slot Section: Beschreibt für die Cluster–Cycles eines jeden Modus die einzelnen Round Slots und weist sie den Knoten zu. Zusätzlich wird für jeden Slot spezifiziert wohin im Speicher die empfangenen Daten geschrieben werden sollen, bzw. von wo die zu sendenen Daten gelesen werden sollen.

[5]Message Descriptor List

Die MEDL ist nicht Bestandteil der TTP–Spezifikation und kann in jeder Controllerimplementierung verschieden sein, zudem kann die MEDL vom Host nicht verändert werden und wird in einem ROM abgelegt.

2.4.2 CHI (FlexRay)

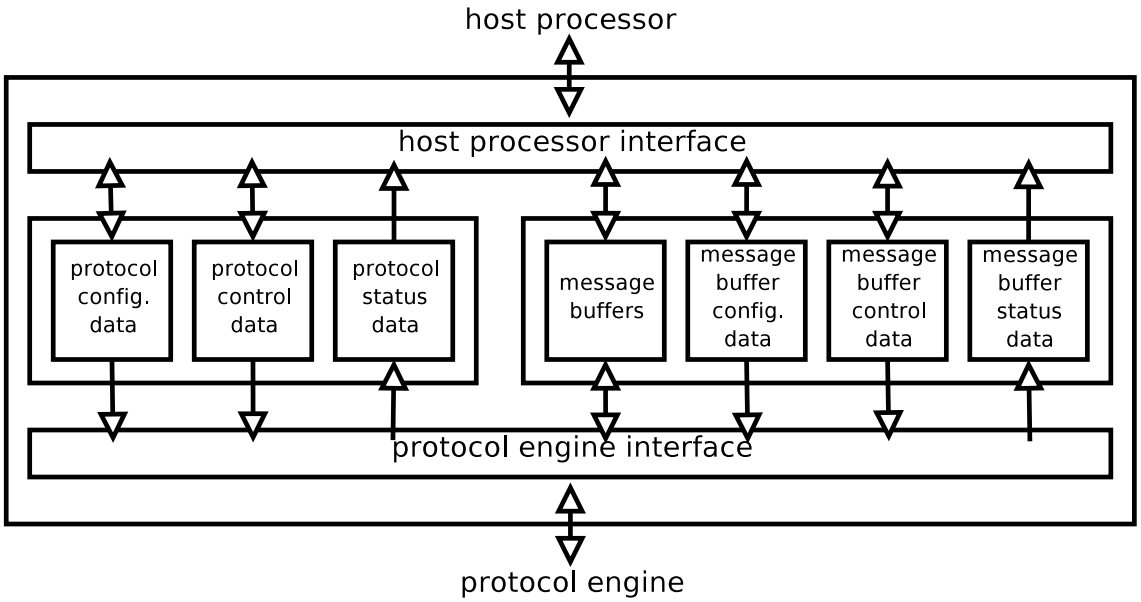

Abbildung 2.8: CHI (FlexRay)

Während bei TTP die Basiskonfiguration fest im Controller verankert ist, kann diese bei FlexRay vom Host über das CHI (siehe Abbildung 2.8) verändert werden. So besitzt das CHI ein `protocol data interface` und ein `message data interface`. Das protocol data interface ist dazu da um Informationen über die aktuelle Protokollausführung zu erhalten und diese zu steuern und besteht aus folgenden Komponenten:

- `protocol configuration data`
- `protocol control data`
- `protocol status data`

Das message data interface dient zur Koordinierung der Nachrichten, welche über FlexRay gesendet bzw. empfangen werden sollen. Dieses enthält Kompenenten analog zum message data interface und noch zusätzlich einen `message buffer` zum Speichern der Nachrichten.

3 Zusammenfassung

In Tabelle 3.1 sind die wichtigsten Eigenschaften von TTP/C und FlexRay in Bezug auf Leistung (Dateneffizienz), Flexibilität, Fehlertoleranz und der Dienste, welche die Protokolle jeweils bieten, gegenübergestellt. TTP ist dabei das ausgereifterte Protokoll und bietet bei weitem mehr Dienste als das noch relativ junge FlexRay. Die Fehlertoleranz ist zudem nur bei TTP formal verifiziert. Auch in der Leistung und Dateneffizienz ist TTP überlegen, was jedoch auf die Kosten von Flexibilität geht, worin die große Stärke von FlexRay liegt. Bei Flexray wird sich aber wohl noch einiges ändern, vorangetrieben von einem Konsortium namhafter Firmen.

	TTP/C	Flexray
Leistung	25MBit/s	10MBit/s
Dateneffizienz	95,8% (10 MBit/s) 78% (100 MBit/s)	45,7% (10 MBit/s) 14,5% (100 MBit/s)
Flexibilität	vorausberechnete MEDL max. ein slot pro Knoten	Knoten lernen MEDL mit der Zeit mehrere slots pro Knoten möglich
Dienste	clock synchronisation membership service (ausschließen defekter Knoten) master–shadow Konfigurationen (fail silence)	clock synchronisation
Fehlertoleranz	formal verifiziert willkürliche Störungen im hub oder den controllern 4 nodes und 2 hubs für fehlerfreien Betrieb benötigt	Folgerung willkürliche Störungen in den controllern 2/3 der Knoten für korrekte Uhrensyncronisation benötigt

Tabelle 3.1: Vergleich von TTP/C und FlexRay

Literaturverzeichnis

[1] TTTech/ttaGroup: “Time–Triggered Protocol TTP/C - High–Level Specification Document Protocol Version 1.1”

[2] FlexRay Consortium: “FlexRay Communications System - Protocol Specification Version 2.1”

[3] TTTech Computertechnik AG: http://www.tttech.com

[4] FlexRay Consortium: http://www.flexray.com

[5] Finn Overgaard Hansen: “Introduction to TTP and FlexRay real–time protocols”