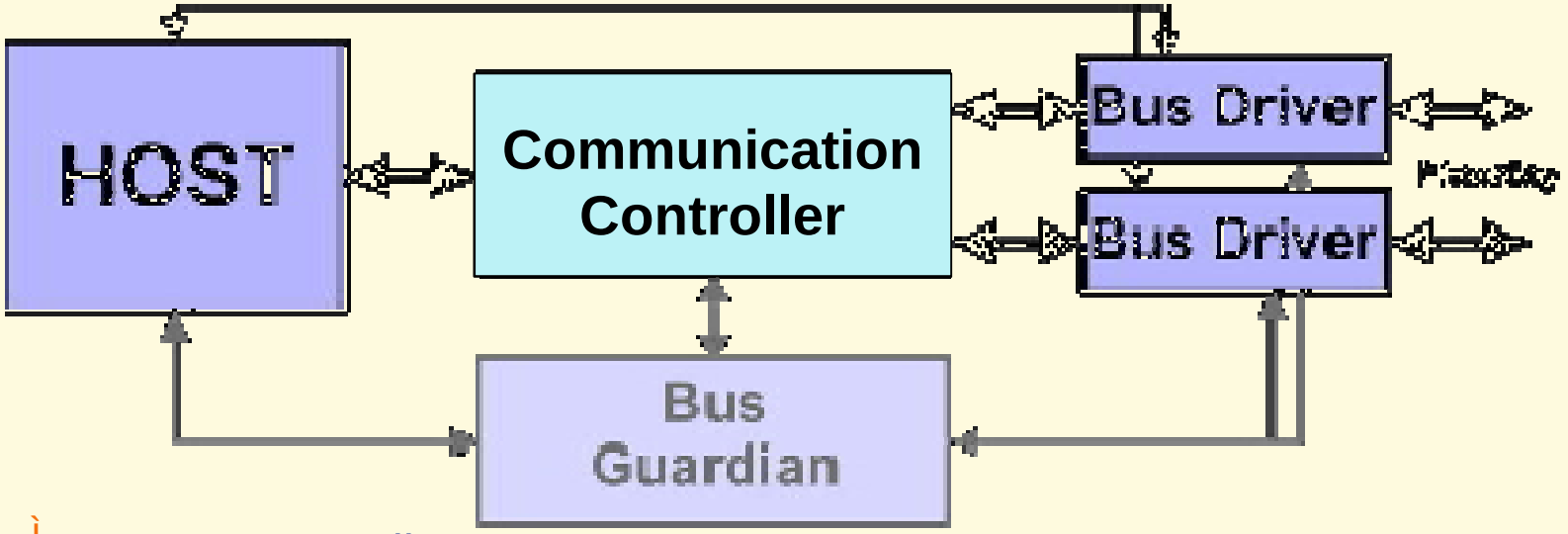
2.1

Komponenten einer FlexRay ECU

FlexRay ECU



- HOST Controller
  - Anwendersoftware
  - Initialisiert Communication Controller
- Communication Controller
  - Synchronisiert sich auf den Bus
  - Versendet und empfängt Daten
- Bus Driver
  - Generiert Übertragungssignal
- Bus Guardian
  - Überwacht Communication Controller und Bus Driver

Bus Guardian:

Hierbei handelt es sich um eine zusätzliche Komponente, die nicht für die Funktion von FlexRay erforderlich ist.
Der Bus Guardian ist eine zusätzliche Sicherheitskomponente, welche die Kommunikation überwacht. Im Fehlerfall kann der Bus Guardian die Bus Treiber sperren und somit verhindern, dass der Communication Controller fehlerhafte Daten auf den Bus sendet.

Communication Controller

Communication Controller

- Synchronisation auf den Bus
- Versendet Daten des Host Controllers
- Stellt empfangene Daten dem Host Controller zur Verfügung

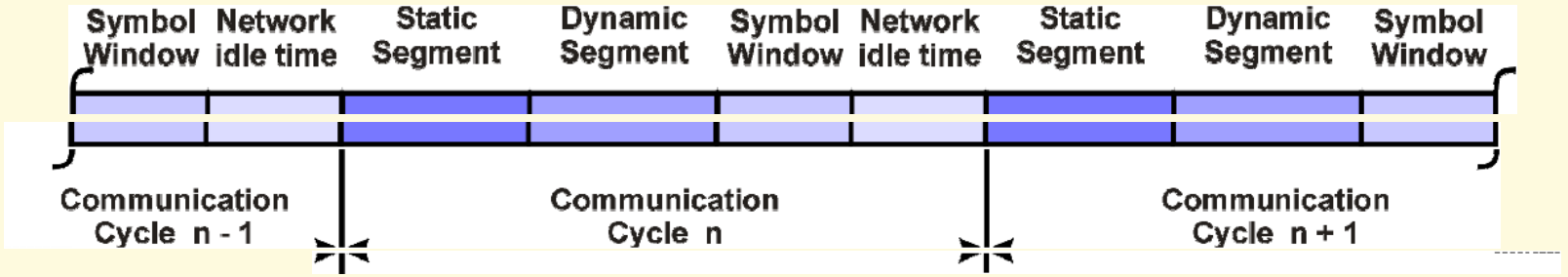
  - Basis für Übertragung ist Protokoll
  - Protokollrelevante Informationen werden angefügt
    - Cycle Counter, CRC

- Konfiguration durch HOST
  - HOST initialisiert den Communication Controller
  - Zugriff nur möglich im Konfigurations-Modus.
    - Während Normalbetrieb sind protokollrelevante Register gesperrt

2.2

FlexRay Übertragungsstruktur

Communication Cycle



- **Datenübertragung bei FlexRay ist zeitgesteuert**
  - Zyklisches Kommunikationsschema
  - Dauer des Communication Cycle festgelegt und konstant
  - Dauer aller einzelnen Segmente festgelegt und konstant

- **FlexRay Communication Cycle besteht aus:**
  - Static Segment
  - Dynamic Segment (optional)
  - Symbol Window (optional)
  - Network Idle Time

Basis für die Datenübertragung ist der Communication Cycle.
Dieser Cycle wiederholt sich ständig.

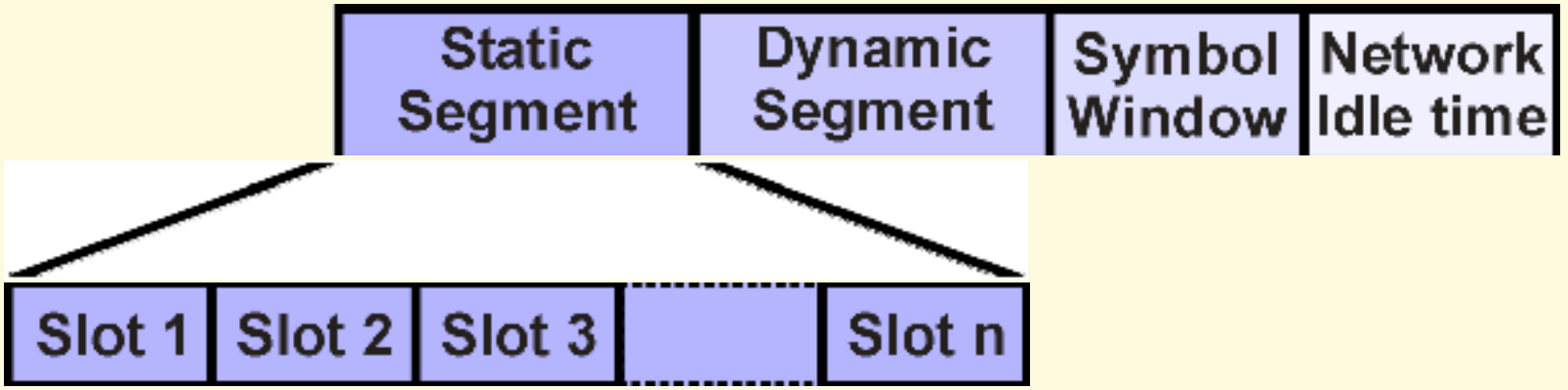
Die Dauer des Communication Cycle ist konfigurierbar zwischen 12µs und 16ms.
Zur Laufzeit ist die Dauer des Communication Cycle konstant.

Die Dauer der einzelnen Segmente ist ebenfalls konfigurierbar.
Zur Laufzeit ist die Dauer der einzelnen Segmente konstant.

Zwingend erforderlich ist das statische Segment und die Network Idle Time.
Im statischen Segment wird die globale Synchronisation realisiert.
Die NIT wird benötigt, um Synchronisationsalgorithmen zu berechnen.

Optionale Segmente wie das dynamische Segment oder das Symbol Window können weggelassen werden.

Static Segment



Statisches Segment

- Statische Slots
  - Zeitfenster für Datenübertragung
- Es darf nur ein Kommunikationsknoten in einem Slot senden
  - Beliebig viele Empfänger
- Alle statischen Slots haben identische Länge

Anzahl und Dauer der statischen Slots ist konfigurierbar und zur Laufzeit konstant.

Die Dauer eines statischen Slots entspricht der Länge, die der zu sendende Frames benötigt, plus Toleranzspielraum.

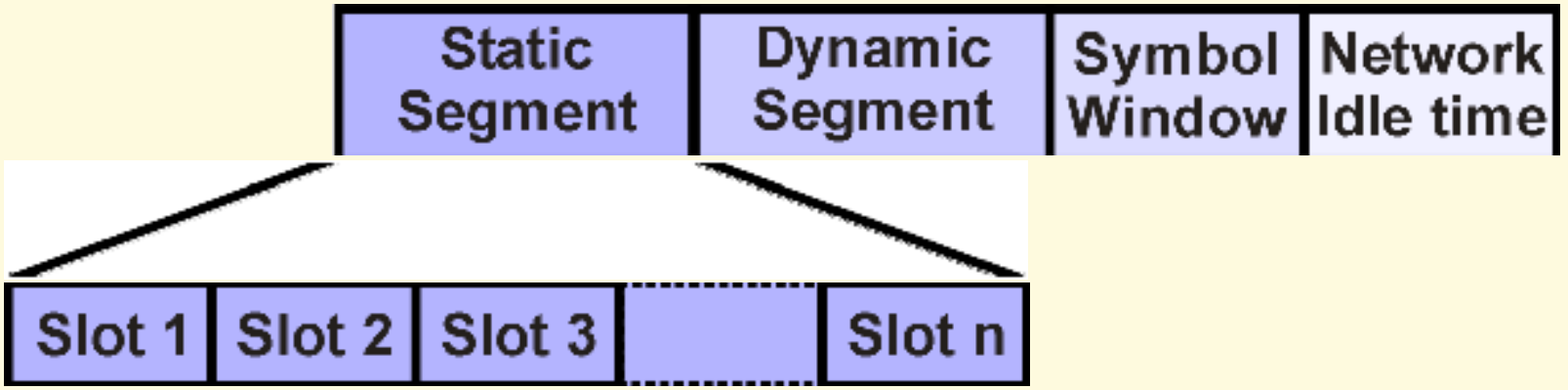
- Frame muss in den statischen Slot „reinpassen“

Somit ist genau die Zeit reserviert die zur Übertragung des Frames benötigt wird.
Ein Teilnehmer der in einem Statischen Slot eine Nachricht sendet hat also in jedem Cycle die Zeit reserviert, um seinen Frame übertragen zu können.
Dadurch ergibt sich ein genau vorhersagbares (deterministisches) Verhalten. Das statische Segment ist somit für die Übertragung von sicherheitsrelevanten Daten geeignet.

Ein Teilnehmer kann für mehrere unterschiedliche statische Slots als Sender konfiguriert werden.

Die maximale Anzahl an statischen Slots sind 1023

Static Segment

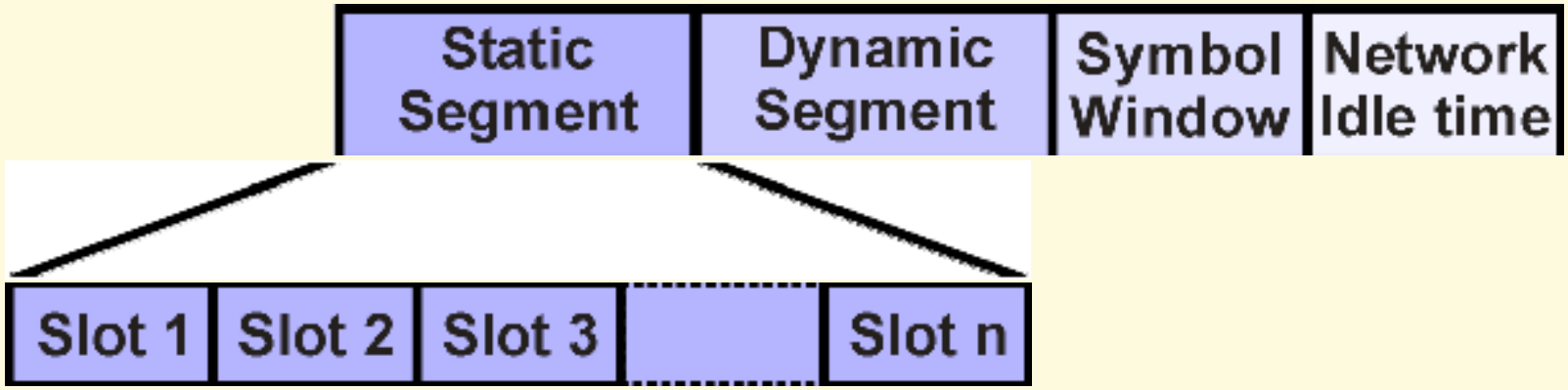


- Konstanter Ablauf basierend auf der Slot ID
  - In jedem Cycle identische Anzahl Slots.
  - Reihenfolge in jedem Cycle identisch
- Time Division Multiple Access (TDMA) Zugriff
  - Zugriff basiert ausschließlich auf Zeitfenster
  - Deterministisches Verhalten

In jedem Cycle werden die statischen Slots in der gleichen Reihenfolge abgearbeitet.
Die Reihenfolge ist durch die Slot ID vorgegeben.

Somit weiß jeder Teilnehmer genau, zu welchem Zeitpunkt eine Nachricht gesendet wird bzw. wann eine Nachricht empfangen werden kann.

Static Segment



- Framestruktur wird in jedem Cycle übertragen
  - Übertragung neuer Daten
  - Zuletzt gesendete Daten werden erneut gesendet
  - Übertragung von Null-Frames
- Es müssen mindestens 2 statische Slots konfiguriert werden
  - Werden für Synchronisation benötigt
- Statisches Segment wird vom Bus Guardian überwacht

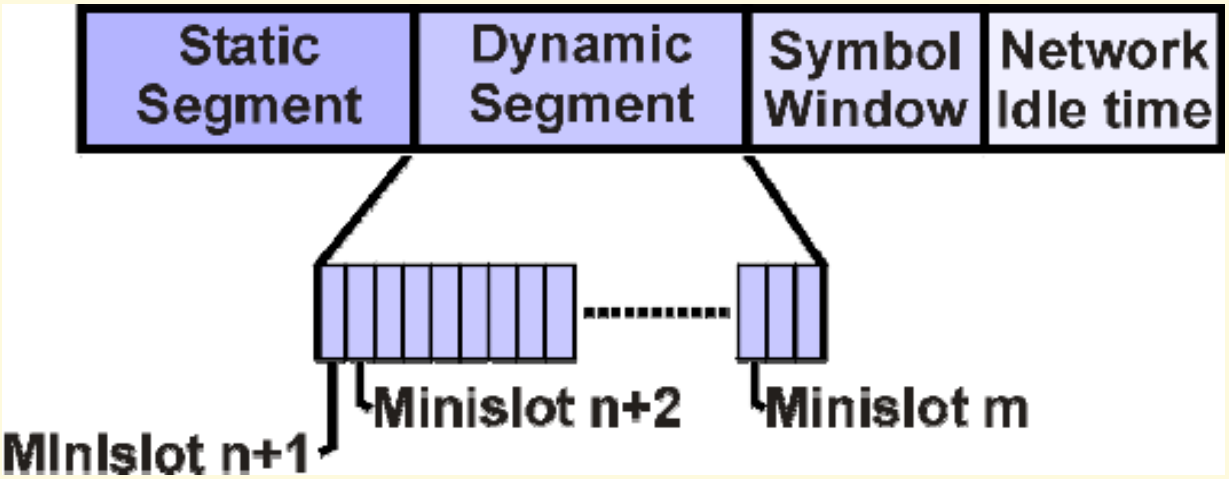
Ist der Communication Controller als Sender für eine statische ID konfiguriert, so legt er automatisch in jedem Cycle im entsprechenden Slot den kompletten Frame auf den Bus.
Ein Nullframe ist durch ein spezielles Bit gekennzeichnet und die Daten im Datenbereich des Frames sind auf ‘0‘ gesetzt.

~~Beispielanwendung~~:
Für eine Applikation ist vorgegeben, dass in jedem Cycle neue Daten gesendet werden sollen. Erhält der Empfänger in einem Cycle nicht wie vorgegeben neue Daten so kann er bereits Rückschlüsse auf die Fehlerquelle ziehen und entsprechend reagieren:

1) Es wird ein Nullframe empfangen
   - Î Das Übertragungsmedium ist in Ordnung
   - Î Der Communication Controller des Senders ist in Ordnung
   - Î Die Sendeapplikation scheint ein Problem zu haben
   - Î Aktuell gibt es keine neuen Daten von diesem Sender
2) Es wird kein Frame empfangen
   1) Es werden keine weiteren Frames auf diesem Kanal empfangen
      - Î Das Übertragungsmedium ist gestört
   2) Es werden noch andere Frames auf diesem Kanal empfangen
      - Î Das Übertragungsmedium ist nach dem Sender unterbrochen
      - Î Oder: Die Sende – ECU ist komplett - oder zumindest der Communication Controller ist ausgefallen

Jeder Sync Node teilt den anderen Teilnehmern seine lokale Zeit durch einen Sync Frame mit.
Für eine erfolgreiche Synchronisation müssen mindestens zwei Sync Frames vorhanden sein (also auch zwei Sync Nodes).
Somit muss das statische Segment mit mindestens 2 Slots konfiguriert werden. In diesen zwei Slots müssen zwei Sync Nodes ihre Sync Frames übertragen und so eine Synchronisation ermöglichen.

Dynamic Segment



Dynamisches Segment

- Minislots
  - Zeitpunkt zu dem Datenübertragung beginnen kann
- Zur Laufzeit wird entschieden, ob Minislot verwendet wird
- Anzahl der übertragenen Daten ist variabel

Das dynamische Segment ist unterteilt in Minislots.
Ein Minislot stellt ein kurzes Zeitfenster dar, zu dem der zugewiesene Teilnehmer mit dem Senden beginnen kann.

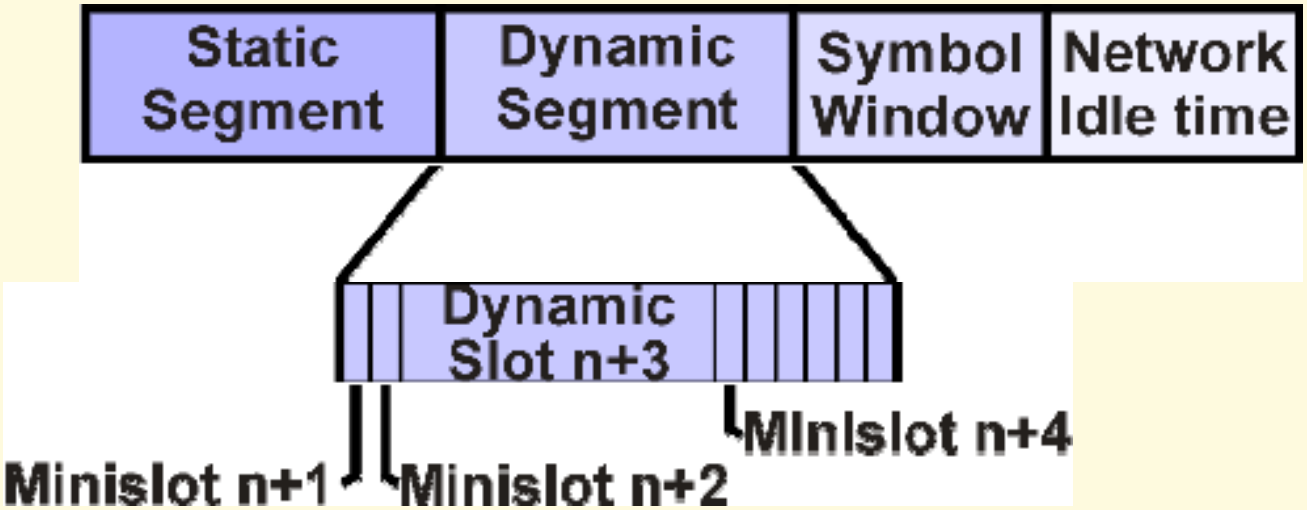
Zur Laufzeit entscheidet der Communication Controller ob er den Minislot verwendet oder nicht.
Verwendet wird ein Minislot nur dann, wenn der Communication Controller Daten zum Senden hat, wenn er also zuvor vom HOST neue Daten zur Verfügung gestellt bekommen hat.

- Im dynamischen Segment werden keine Nullframes versendet

Die Nummerierung der Minislots erfolgt fortlaufend auf das statische Segment.
D.h. der erste Minislot hat die ID = letzte statische ID + 1

Insgesamt (statisch + dynamisch) dürfen maximal 2047 Slot IDs vergeben werden. Davon dürfen maximal 1023 IDs im statischen Segment liegen.

Dynamic Segment



- Übertragung muss im dynamischen Segment beendet sein
  - Latest Dynamic Transmission Start
- “Arbitrierung“ der Nachrichten anhand der Minislot ID

Wenn ein Teilnehmer Daten zum Versenden hat, so wartet der CC auf den entsprechenden Minislot und beginnt dann mit der Übertragung des Frames.

- Der Minislot wird zu einem Dynamischen Slot aufgeweitet.

Die Länge des dynamischen Slots hängt von der Anzahl der aktuell zu übertragenden Daten ab und ist variabel. Alle Teilnehmer erkennen, dass ein Minislot verwendet wird, und warten mit dem Zählen der Minislots. Das Ende des Dynamischen Slots erkennen ebenfalls alle Teilnehmer gleichzeitig und zählen dann synchron die Minislots weiter.

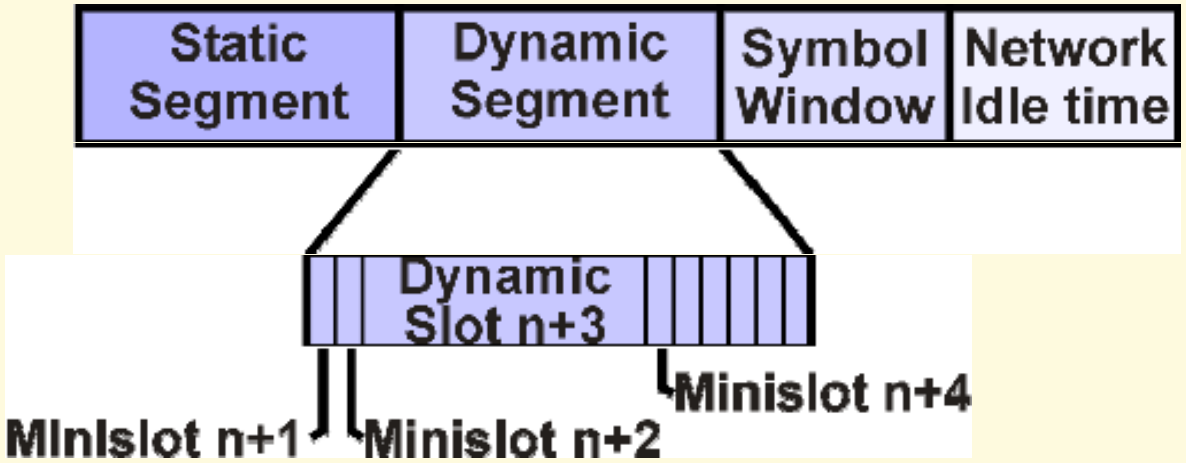
Durch das Senden eines Frames werden die nachfolgenden Minislots nach hinten verschoben. Dadurch, dass die Übertragung zum Ende des dynamischen Segments abgeschlossen sein muss, ergibt sich ein Zeitpunkt innerhalb des dynamischen Segments, zu dem letztmöglich mit der Übertragung begonnen werden kann. Wenn dieser Zeitpunkt überschritten ist “passt der Frame nicht mehr rein“.
Dieser Zeitpunkt wird nicht dynamisch ermittelt sondern ist ein vorzugebender Konfigurationswert. Dem Communication Controller muss mitgeteilt werden, wie lange der maximale Frame ist, der im dynamischen Segment versendet werden soll.
Basierend auf dieser maximalen Framelänge berechnet sich dann die “Latest dynamic transmission start“

Je höher die Minislot ID ist, desto größer ist die Wahrscheinlichkeit, dass in vorausgehenden Minislots bereits andere Teilnehmer ihre Frames versendet haben. Somit nimmt mit zunehmender Minislot ID die Wahrscheinlichkeit zu, dass der Minislot über die “Latest dynamic transmission start“ hinausgeschoben wird und im aktuellen Cycle nicht versendet werden kann.

Besonderheiten:

1) Die erste dynamische ID ist immer deterministisch.
2) Wenn das dynamische Segment so konfiguriert wird, dass es mindestens die Dauer der ersten und zweiten dynamischen ID hat, so ist die erste und die zweite dynamische ID deterministisch.
3) Wenn das dynamische Segment so konfiguriert wird, dass es mindestens die Dauer der ersten, zweiten und dritten …

Dynamic Segment



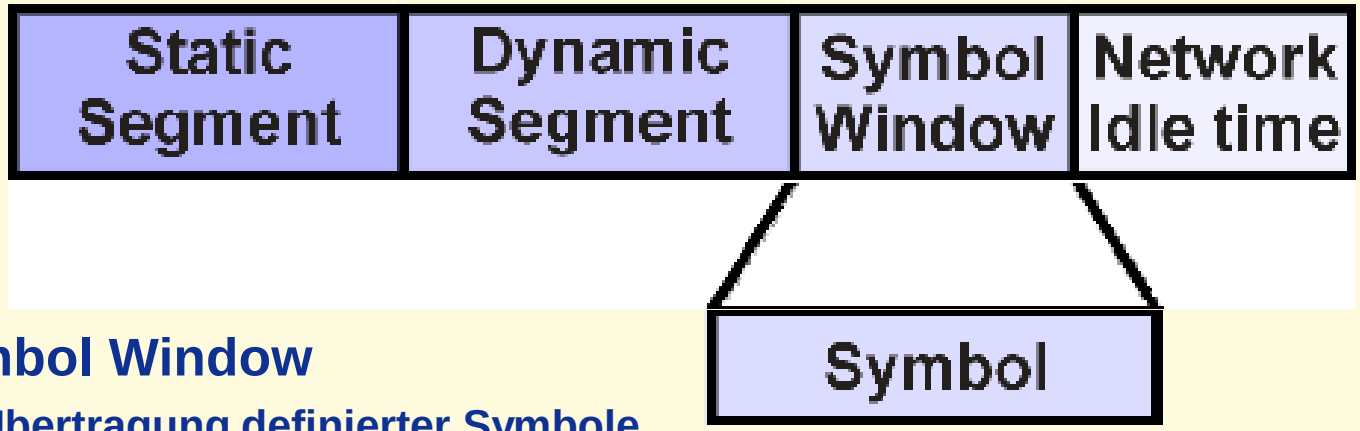
- Dynamisches Segment ist optional
- Flexible Time Division Multiple Access (FTDMA) Zugriff
  - Priorisierter Zugriff basierend auf Datenaufkommen, das in Minislot versendet werden soll
- Dynamische Slots werden nicht vom Bus Guardian überwacht

Einzelne dynamische Slots werden nicht vom Bus Guardian überwacht.
Entweder wird das dynamische Segment komplett freigegeben oder komplett gesperrt.

Das dynamische Segment ist nicht für sicherheitsrelevante Daten vorgesehen.

Formelle Klarstellung:
Die in diesen Unterlagen als Minislot bezeichneten Bereiche des dynamische Segments werden laut Protokoll auch als “dynamic slot without transmission“ bezeichnet.

Symbol Window



Symbol Window

- **Übertragung definierter Symbole**
  - Im Protokoll festgelegte Symbole
  - Nur ein Symbol pro Cycle
  - Media Test Symbol (MTS)
    - 30 Bit ‘Low‘
    - Test des Bus Guardian

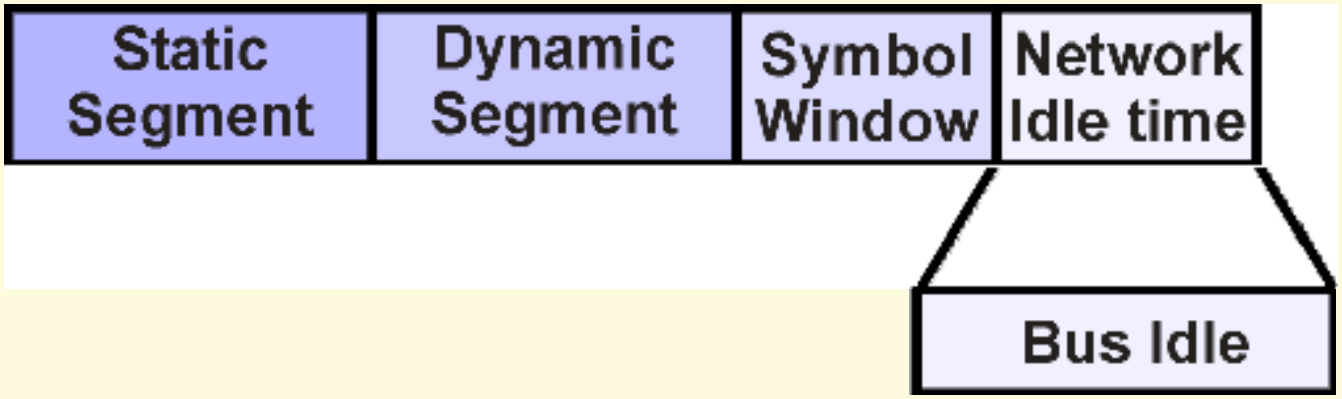
- **Keine Arbitrierung im Symbol Window**
  - Muss im Host realisiert werden

- **Symbol Window ist optional**

Im Symbol Window können nur Symbole übertragen werden, die im Protokoll definiert sind.

Aktuell (Protokoll V2.1a) ist nur das MTS Symbol definiert.
Es dient zum Testen des Bus Guardian.
Da aktuell kein Bus Guardian verfügbar ist, gibt es auch keine Verwendung für das MTS.
Somit gibt es aktuell auch keine Verwendung für das Symbol Window.

Network Idle Time



Network Idle Time

- Zeitraum, in dem keine Daten auf dem Bus übertragen werden
- Zeit für Synchronisierung
  - Durchführen der Synchronisationsalgorithmen
  - Zuweisen der Korrekturwerte

Die Dauer der NIT ist abhängig von der Anzahl der vorhandenen Sync Frames.

Für jeden Sync Frame wird die Abweichung zur lokalen Zeitbasis gemessen und gespeichert.
Am Ende des Statischen Segments hat der Communication Controller eine Liste mit Abweichungen. Aus dieser Liste muss der Mittelwert berechnet werden.

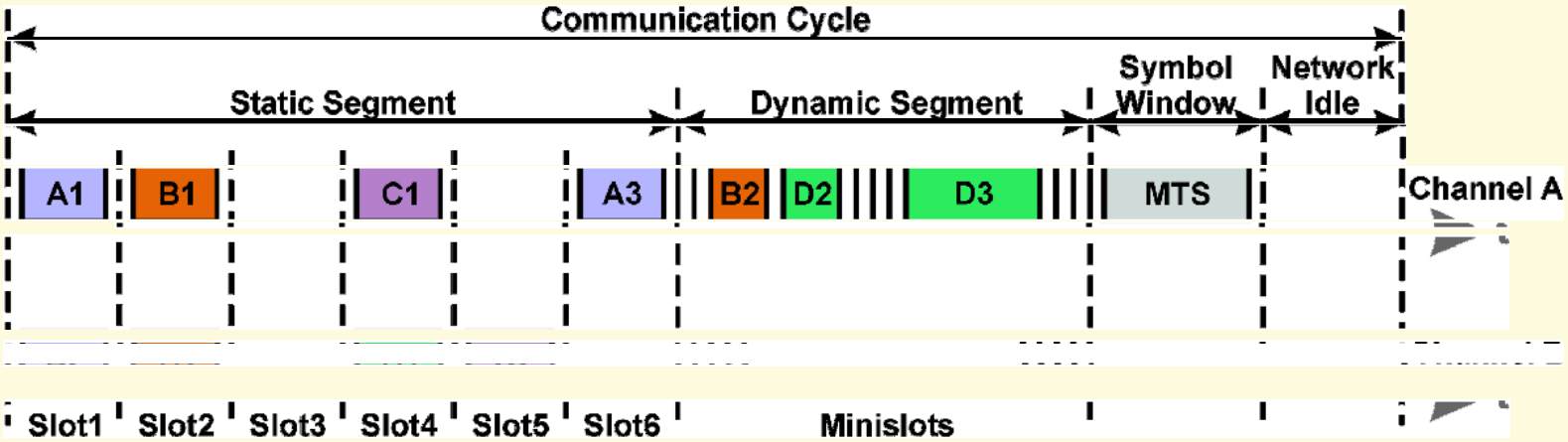
- je länger die Liste ist, desto länger benötigt der Communication Controller für die Berechnung.

Die Dauer der NIT ist zudem auch herstellerspezifisch. Hersteller A kann die Berechnung schneller durchführen als Hersteller B.

2.3

FlexRay Zugriffsstrukturen

Bus Access Static Segment

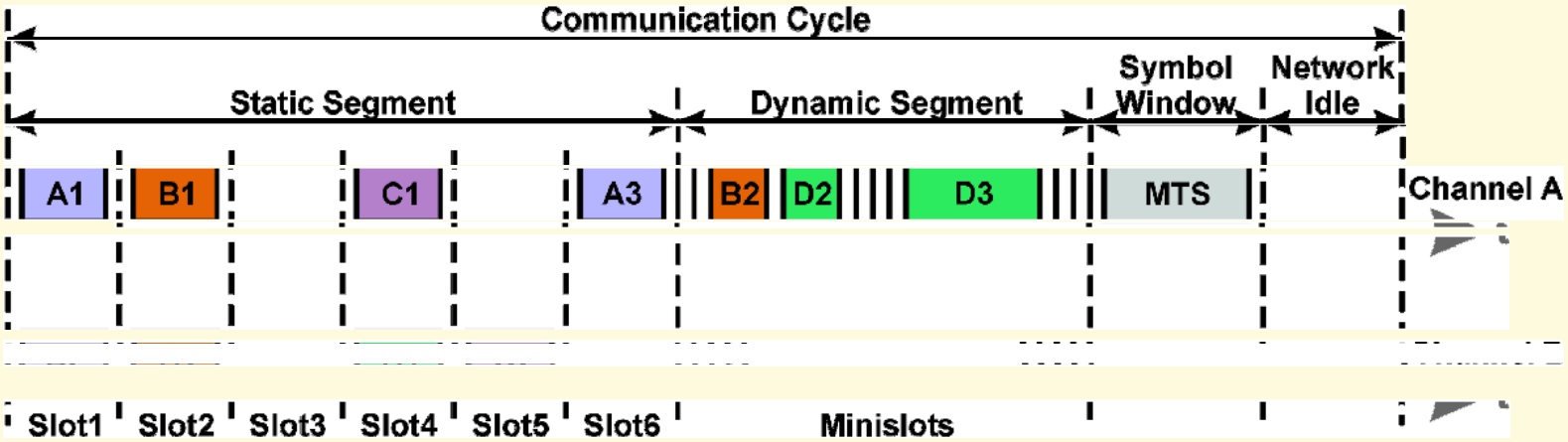


Statisches Segment:

- Ablauf auf beiden Kanälen identisch
  - Î Gesamtdauer des statischen Segments ist identisch
  - Î Zeitliche Abfolge synchron

- Zuweisung welcher Knoten in Slot sendet ist kanalunabhängig

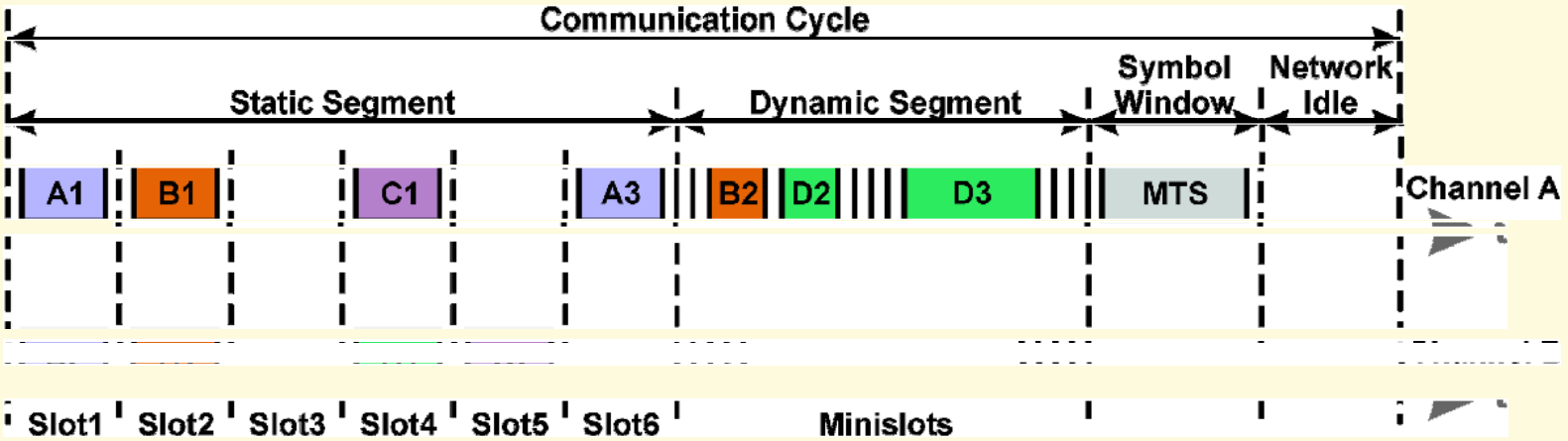
Bus Access Static Segment



Mögliche Zugriffsstrukturen im statischen Segment:

- Slot wird auf beiden Kanälen unabhängig verwendet
  - Volle Ausnutzung der Bandbreite
- Identische Daten in einem Slot auf beiden Kanälen
  - Redundante Datenübertragung
- Slots werden nicht verwendet
  - Reservierung für zukünftige Anwendungen

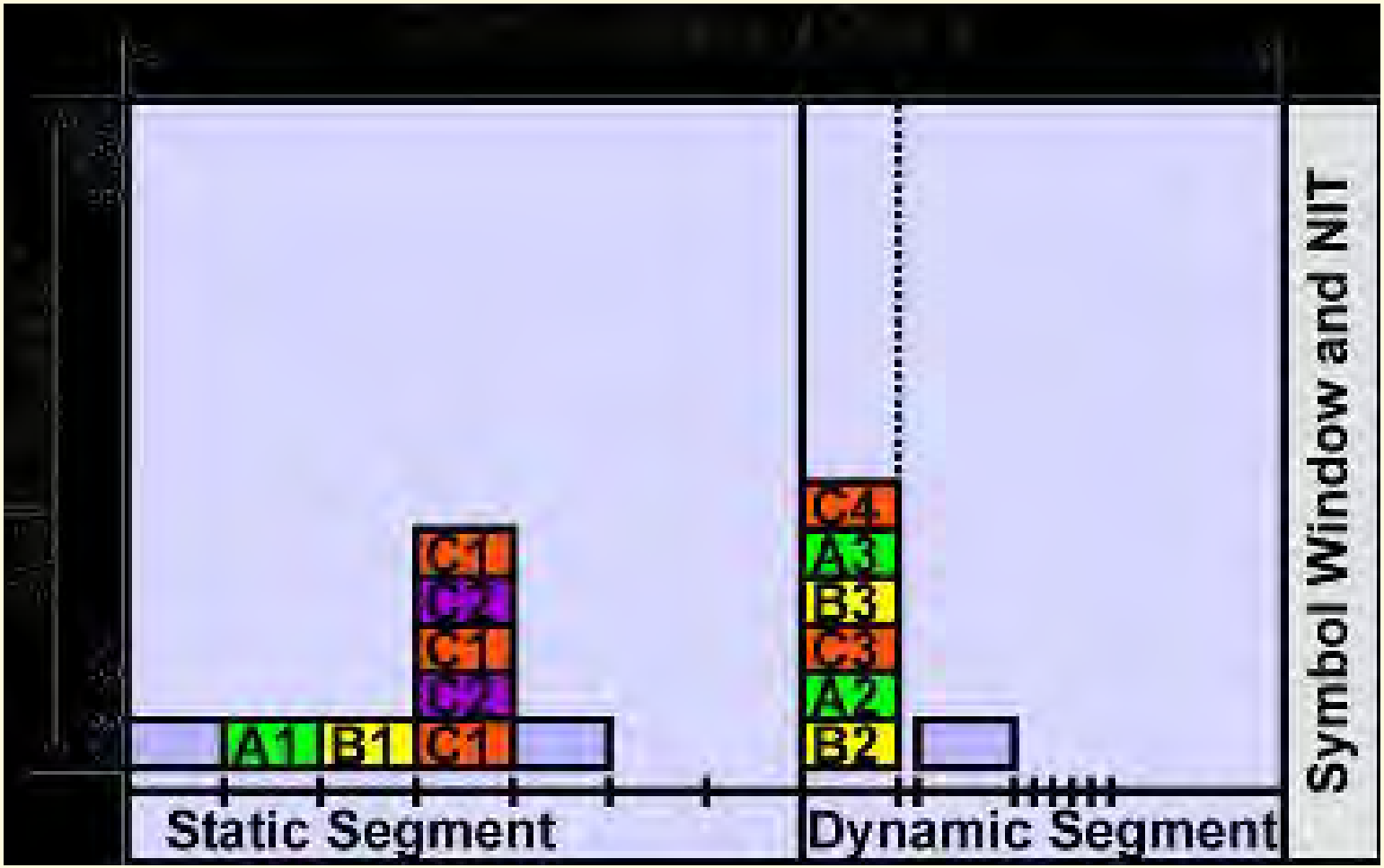
Bus Access Dynamic Segment



Dynamisches Segment:

- Gesamtlänge des Segments auf beiden Kanälen identisch
- Für beide Kanäle identische Anzahl Minislots konfiguriert
- Zeitliche Abfolge nicht synchron
  - Zugriff auf den Bus läuft auf beiden Kanälen getrennt ab

Cycle - Multiplexing



- Statisches Segment:

Dadurch, dass bei statischen Slots in jedem Cycle der Frame auf den Bus gelegt wird, kann immer nur ein Teilnehmer in einem statischen Slot senden.
Basierend auf dem Cycle Counter kann festgelegt werden, dass in Cycle x die Nachricht 1 und in Cycle y die Nachricht 2 übertragen wird. Beide Nachrichten müssen aber vom selben Teilnehmer gesendet werden.

- Dynamisches Segment.

Ein Minislot wird nur dann verwendet wenn der Communication Controller Daten zum Senden zur Verfügung hat. Wenn keine Daten vorhanden sind, so wird der Minislot nicht verwendet. Somit besteht im dynamischen Segment die Möglichkeit, dass mehrere Teilnehmer den selben dynamischen Slot verwenden.
Basierend auf dem Cycle Count kann festgelegt werden, dass in Cycle x Teilnehmer A und in Cycle y Teilnehmer B eine Nachricht auf dem Bus versendet.

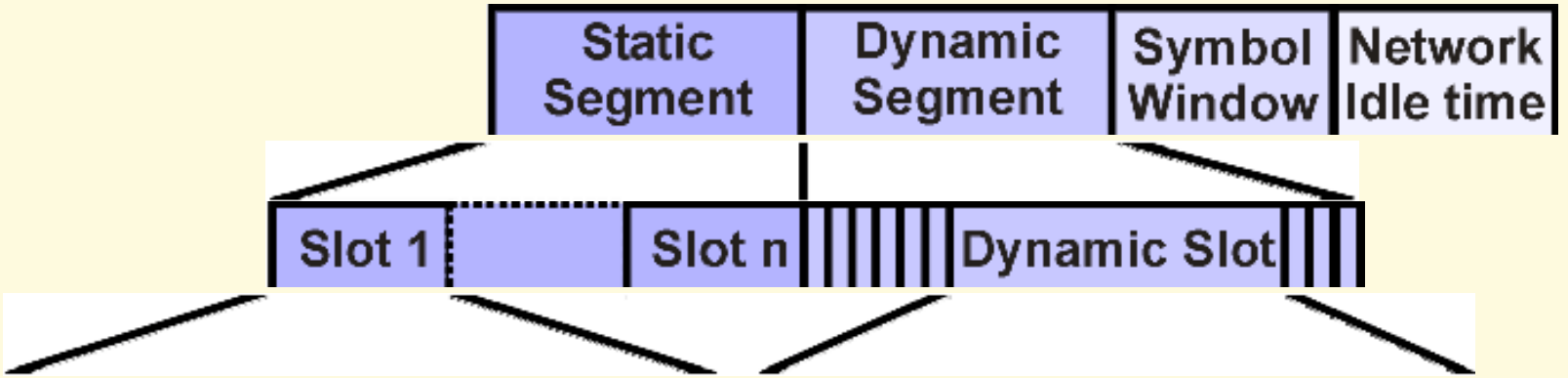
Die Communication Conrtroller unterstützen diese Funktion, indem Filter für die Nachrichten bereitgestellt werden. Somit muss die Applikation sich nicht um eine Synchronisierung kümmern, der Communication Controller versendet die Nachricht nur im entsprechenden Cycle.

Das Cycle Multiplexing im dynamischen Segment wird z.B. vom AUTOSAR Netzwerk Management verwendet.

2.4

FlexRay Frame Format

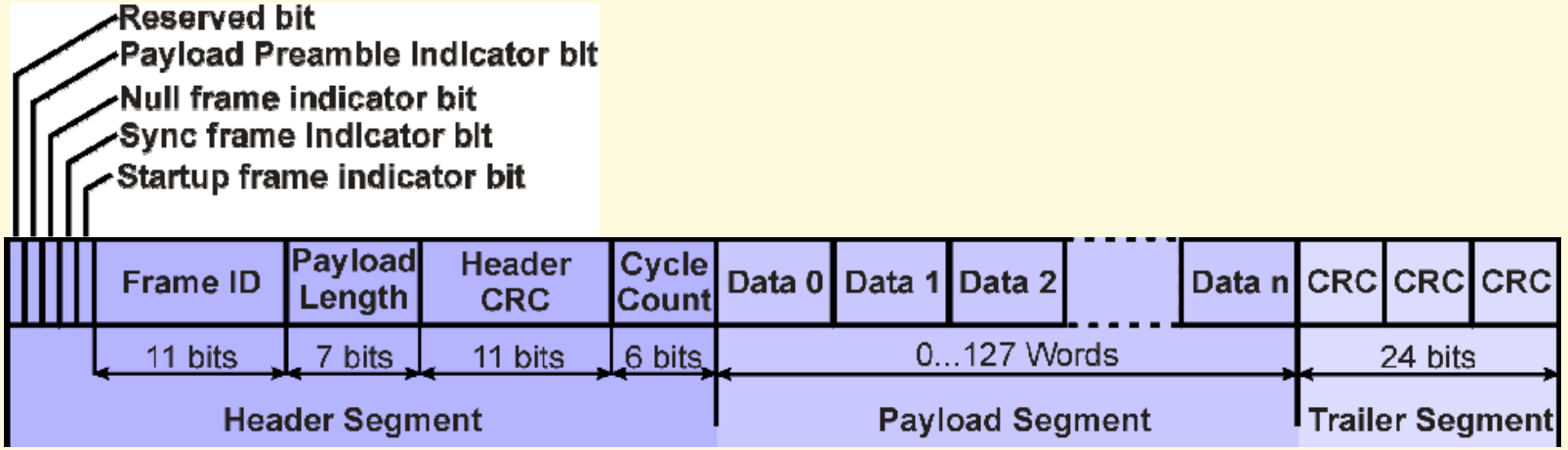
FlexRay Frame Struktur



FlexRay Frame

- Frame ist Grundstruktur für Datenübertragung
  - In jedem statischen und dynamischen Slot gilt identische Frame-Struktur.

FlexRay Frame Format



3 Basissegmente des Frames:

- Header Segment
  - Frame-Informationen
  - Synchronisationsinformationen
  - Header CRC
- Payload Segment
  - Anwenderdaten
- Trailer Segment
  - Frame CRC

2.6

FlexRay Synchronisation

Cluster Syncronisation

- **2 Synchronisationsprozesse:**

1. **Synchronisationsprozess zum Anpassen der lokalen Bit-**Abtastung
   - Ausrichten der Bit – Abtastung auf den empfangenen Datenstrom

2. Synchronisationsprozess zum Anpassen der lokalen Zeitbasis auf eine globale Zeitbasis
   - In allen Knoten ist der Beginn und die Dauer eines Cycles identisch

Cluster Synchronisation

- **Sync Frames**
  - Nur in statischem Segment
  - Sollen auf beiden Kanälen im identischen Slot übertragen werden

- **Sync Nodes**
  - Jeder Sync Node darf maximal einen Sync Frame in einem Slot übertragen (Kanal A + B)
    - Sync Frame muss in jedem Cycle im identischen Slot übertragen werden
  - In einem Cluster müssen mindestens 2 Sync Nodes vorhanden sein
    - Optimal mindestens 3 Sync Nodes
    - Maximal 15 Sync Nodes pro Cluster

Bei der Konfiguration muss angegeben werden, ob der Knoten ein SyncNode ist und in welchem Slot der SyncFrame versendet werden soll (Protokoll: “keySlot ID“). Der Communication Controller setzt dann automatisch im entsprechenden Frame das Sync Frame Indication Bit.

Ein Sync Frame kann wie ein beliebiger Frame zum Senden von Daten verwendet werden.

Die Konfiguration des Sync Frames ist unabhängig von der Bufferkonfiguration. Sobald ein Sync Frame konfiguriert ist, sendet der Communication Controller diesen Sync Frame auf den Bus, auch wenn kein Buffer für diese ID vorhanden ist.

Cluster Synchronisation

Messung der Abweichung

- Für alle vorhandenen Sync Frames wird zeitliche Abweichung ermittelt
  - Korrekturwert für Cycle Offset
  - Korrekturwert für Cycle Frequenz

Korrektur

- Anzahl der Microticks, aus denen ein Macrotick besteht, wird geändert

Der Action Point ist der zeitliche Referenzpunkt, zu dem der Communication Controller einen Frame erwartet. Um die Abweichung zu ermitteln wird die Differenz zwischen Action Point und dem Zeitpunkt gemessen, zu dem der SyncFrame real empfangen wird.
Im Beispiel wird in Slot n+1 ein SyncFrame empfangen. Die Messung der Abweichung im Cycle m ergibt einen Wert für die Offsetkorrektur. Die selbe Messung wird im nachfolgenden Cycle m+1 wiederholt und ergibt erneut einen Wert für die Offsetkorrektur. Die Differenz der beiden Korrekturwerte stellt einen Wert für die Frequenzkorrektur dar.
Diese Messung wird für jeden empfangenen SyncFrame durchgeführt. Somit erhält der Communication Controller am Ende eines Cycles eine Liste mit Abweichungen, aus denen ein Mittelwert gebildet wird (in jedem geraden Cycle eine Liste für die Offsetkorrektur, und in jedem ungeraden Cycle eine Liste für die Offsetkorrektur und für die Frequenzkorrektur).

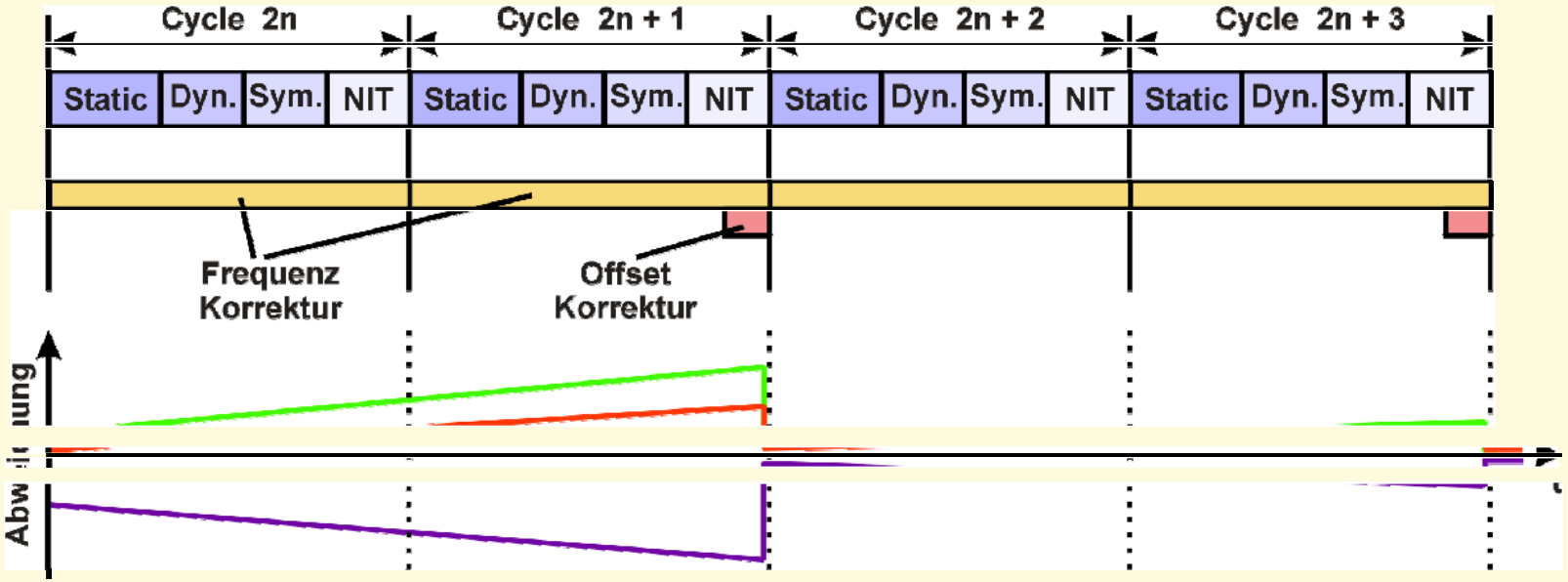
Über den “Fault-tolerant midpoint“ Algorithmus wird der Mittelwert berechnet:

- Die Werte werden der Größe nach sortiert
- Abhängig von der Gesamtanzahl werden die k größten und k kleinsten Werte verworfen

Gesamtanzahl	k
1-2	0
3-7	1
>7	2

- Aus dem überbleibenden größten und kleinsten Wert wir der Mittelwert berechnet.

Cluster Synchronisation



Frequenz-Korrektur

- Dauer von ausgesuchten Macroticks im Cycle werden geändert
- Dauer des Cycles wird angepasst

Offset-Korrektur

- Dauer der Macroticks in Network Idle Time wird gerändert
- Zeitpunkt zu dem nachfolgender Cycle beginnt, wird angepasst

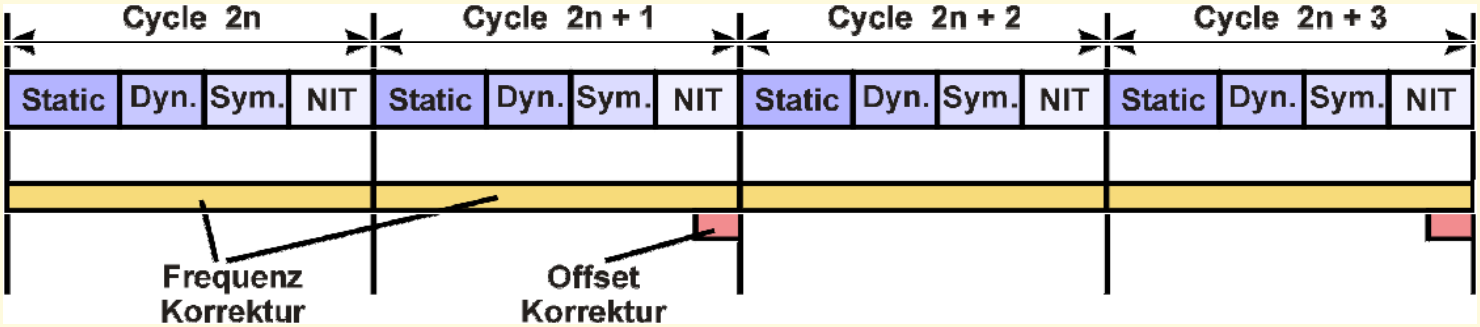
Bei der Anwendung der Korrekturwerte wird die Dauer eines ausgewählten Macroticks geändert.
Ein Macrotick besteht aus einer nominalen (konfigurierten) Anzahl von Microticks, z.B. 1MT = 40µT.
Um einen Korrekturwert von z.B. +2µT anzuwenden, wird die Dauer eines Macroticks auf 42µT erhöht.

Um die Frequenzabweichung ermitteln zu können, muss die Messung während zwei unkorrigierten (Offset) Cycles erfolgen. Aus diesem Grund wird die Offsetkorrektur nur in jedem ungeraden Cycle zugewiesen.

Bei der Offsetkorrektur wird die Dauer der Macroticks in der NIT geändert.

Der Wert für die Frequenzkorrektur gilt für 2 aufeinanderfolgende Cycles.
Das Ergebnis der Berechnung der Korrekturwerte wird gleichmäßig auf 2 Cycles verteilt.

Cluster Synchronisation



Frequenz-Korrektur

- Basiert auf Messwerten von zwei vorhergegangenen Cycles
  - Zuweisung der Korrekturwerte nur jeden geraden Cycle
  - Korrekturwert gilt für die nächsten 2 Cycles
- Für unterschiedliche Messwerte von Kanal A und Kanal B wird der Mittelwert verwendet

Offset-Korrektur

- Basiert auf Messwerten des vorhergehenden Cycles
  - Zuweisung der Korrekturwerte nur jeden ungeraden Cycle
  - Korrekturwerte während eines geraden Cycles dienen nur zur Fehlererkennung
- Für unterschiedliche Messwerte von Kanal A und Kanal B wird der kleinere Wert verwendet

Synchronisation Notes

- Jeder Knoten muss alle verfügbaren Sync Frames zur Synchronisation verwenden
- Korrekturwert darf nur innerhalb konfigurierbarer Grenzen liegen
  - Parameter maximum offset correction
    maximum rate correction
  - Überschreitung führt zu Fehlerfall
- **Externe Clock Synchronisation**
  - Host kann zusätzlich in die Synchronisation eingreifen
  - Möglichkeit mehrere unabhängige Cluster zu synchronisieren

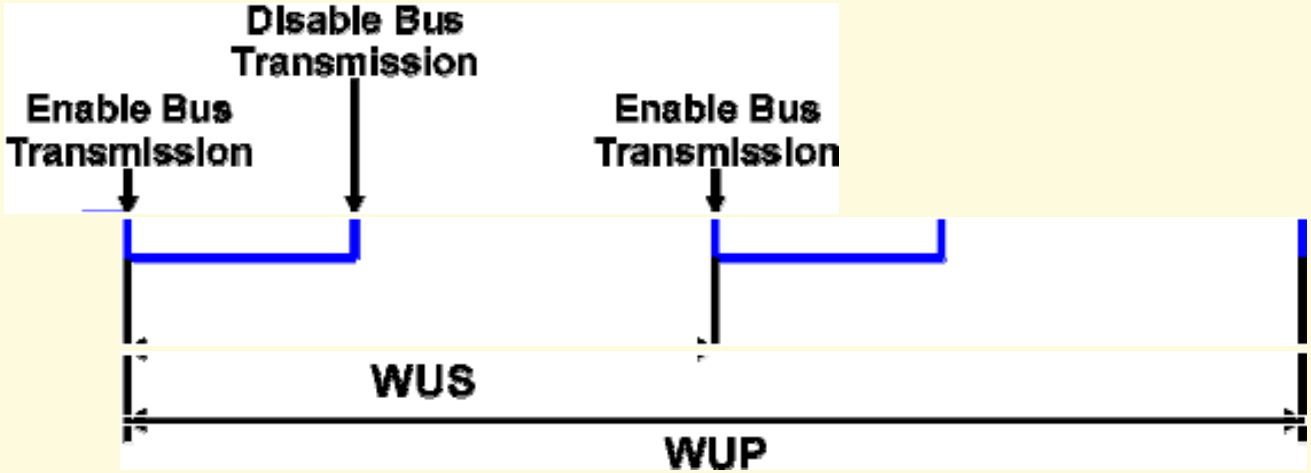
2.7

FlexRay Wakeup

FlexRay Wakeup

- **Wecken aller Teilnehmer**
  - Starten der Stromversorgungen der Teilnehmer
  - Durch Wakeup wird noch keine Kommunikation aufgebaut
- **Communication Controller unterstützt nur Power Off und Power On**
  - kein Sleep Mode
- **Remote Wakeup wird von Bus Driver erkannt**
  - Bus Driver weckt Communication Controller und Host
  - Host initialisiert Communication Controller

Wakeup Symbol



- **Wakeup Symbol WUS**
  - 15 – 60 bit ‘low‘ + 45 – 180 bit ‘idle‘
  - minimale Dauer ‘low‘ 4µs
  - minimale Dauer ‘idle‘ 4µs
  - Während Idle time wird überprüft, ob es zu Kollisionen kommt
- **Wakeup Pattern WUP**
  - Mehrere (2 – 63) Wakeup Symbols WUS

Die Bustreiber haben keine Information über die verwendete Baudrate und erkennen ein WUS anhand einer absoluten Zeit von minimal 4µs low gefolgt von min. 4µs idle.
Abhängig von der verwendeten Baudrate muss die Dauer des WUS so konfiguriert werden, dass diese absoluten Zeiten erreicht werden.

Befindet sich im System z.B. ein Aktiver Stern, so werden einige WUS “verschluckt“. Der Stern benötigt eine gewisse Zeit bis er aktiv ist und die WUS weiterleiten kann. Ein weckender Knoten muss also so lange WUS schicken und diese Zeit überbrücken, bis der Stern die WUS an die anderen Teilnehmer weiterleiten kann.

Wakeup notes

- **Es müssen immer beide Kanäle geweckt werden.**
  - Sicherstellung, dass Knoten geweckt werden die nur an einem Kanal angeschlossen sind
- **Beide Kanäle sollen nicht gleichzeitig geweckt werden**
  - Zwei Teilnehmer sollen je einen Kanal wecken
  - Vermeidung, dass ein fehlerhafter Knoten beide Kanäle stört
- **Kein Feedback, ob Teilnehmer den Wakeup erkannt haben**
  - Muss bei Bedarf durch aktive Kommunikation nach Startup vom Host realisiert werden

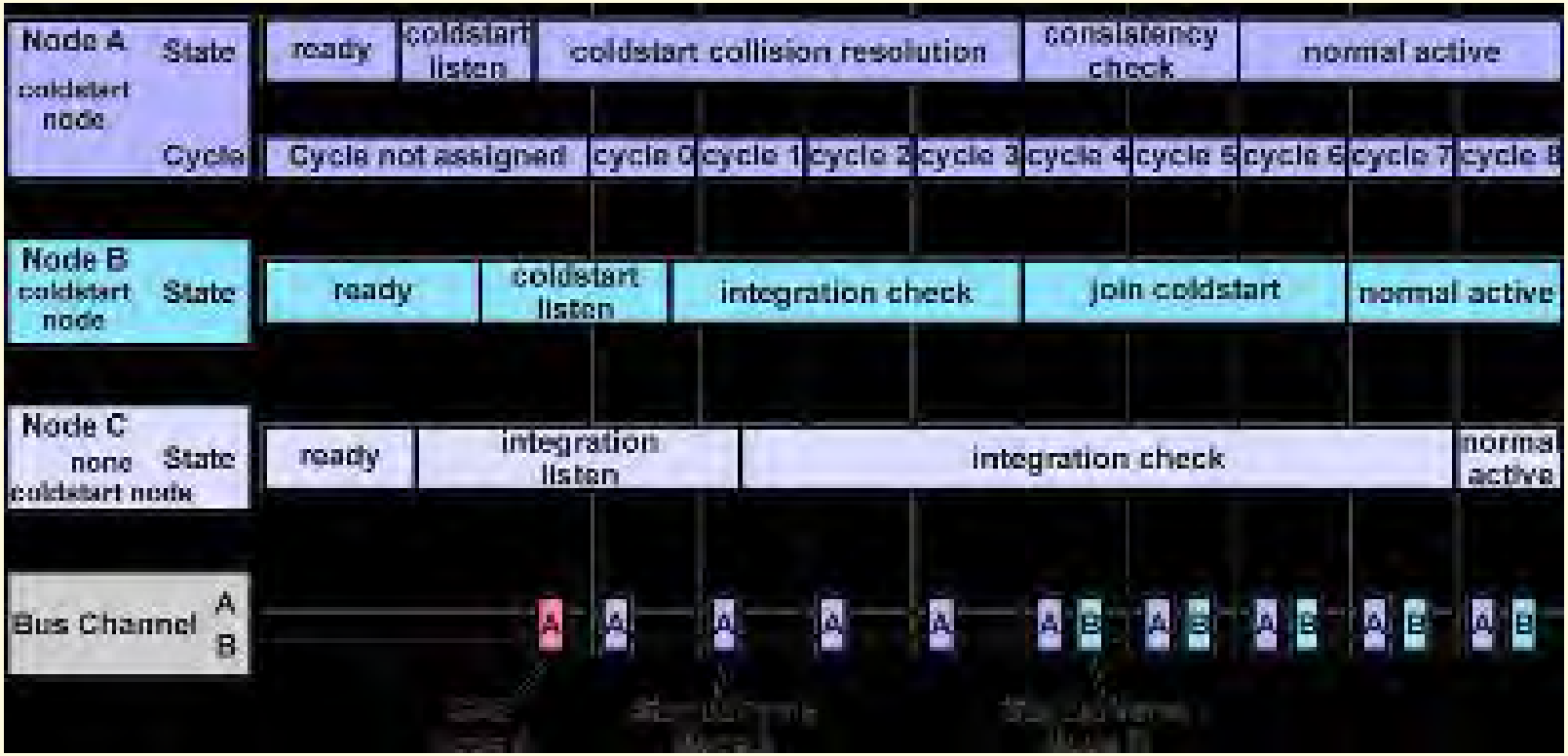
2.8

FlexRay Startup

Cluster Startup

- **Initialisierung der Kommunikation**
- **Coldstart Nodes**
  - Knoten die Synchronisation aufbauen
  - Alle Coldstart Nodes sind Sync Nodes
  - Leading Coldstart Node
    - Initiierender Synchronisationsknoten
    - Knoten beginnt Datenübertragung basierend auf eigener unkorrigierter Zeitbasis
  - Following Coldstart Node
    - Nachfolgender Synchronisationsknoten
    - Knoten versucht sich auf den Datenstrom des Leading Coldstart Node aufzusynchronisieren
  - Non Coldstart Nodes
    - Knoten synchronisieren sich nachfolgend auf den von den Coldstart Nodes aufgebauten Datenstrom auf

Startup



- Startup erfolgt synchron auf Kanal A und Kanal B
- Leading Coldstart Node wechselt nach normal operation nach minimal 6 Cycles
- Following Coldstart Node wechselt nach normal operation nach minimal 7 Cycles
- None Coldstart Node wechselt nach normal operation nach minimal 8 Cycles

- ~~Ablauf~~:

Node A erhält vom Host das Kommando, einen Coldstart durchzuführen.
Daraufhin wechselt er in “coldstart listen“ und überprüft hier, ob bereits Kommunikation vorhanden ist.
Wenn nicht, wird das CAS (collision avoidance symbol) übertragen. Damit wird angezeigt, dass ein Coldstart durchgeführt werden soll.
Node A führt den coldstart als leading coldstart node durch.
Nach dem CAS überträgt Node A in den ersten 4 Cycles seine Sync Frames alleine.
Node B hat das Host-Kommando etwas später erhalten und erkennt das CAS und die SyncFrames von Node A. Er weiß somit, dass er als following coldstart node den Startup durchführen muss.

- Die Entscheidung, wer leading- und wer following coldstart node wird, wird zur Laufzeit getroffen. Prinzip: “Wer zuerst kommt, der malt zuerst“

Wenn sich Node B erfolgreich auf den Datenstrom von Node A aufsynchronisieren kann, darf er ab Cycle 4 seine SyncFrames mit auf den Bus legen.
Diese SyncFrames fließen bei Node A in die Synchronisation mit ein. Wenn kein Fehler auftritt wechselt Node A ab Cycle 6 in normal active.
Node C als none coldstart node darf ab cycle 8 an der Kommunikation teilnehmen (vorausgesetzt, die Synchronisation war erfolgreich).

Collision Avoidance Symbol

Collision Avoidance Symbol CAS

- 30 Bit ‘low‘
- Initialisierung des Startup
- Übertragung beginnt mit Transmission Start Sequence
- Identisch zu Media Test Access Symbol (MTS)
  - Wird aber nicht in Symbol Window übertragen !

Startup Notes

- **Mehrere Knoten senden CAS gleichzeitig**
  - Knoten starten alle als Leading Coldstart Nodes
  - Jeder Knoten überwacht Bus auf gültige Frames
  - Knoten, der Startup Frame mit niedrigster ID konfiguriert hat, überträgt diesen Frame in Cycle 0 als Erster
    - dieser Frame wird von den anderen Knoten erkannt und diese wechseln in listen mode
  - Wenn mehrere Knoten als Leading Coldstart Nodes starten, bleibt der Knoten mit der niedrigsten Sync Frame ID übrig
- **Coldstart Inhibit Mode**
  - Knoten darf die Rolle des Leading Coldstart Node nicht übernehmen.
  - Knoten darf sich auf bestehende Kommunikation oder als Following Coldstart Node aufsynchronisieren
    - Möglichkeit den Startup hinauszuzögern bis alle Knoten wach und konfiguriert sind