IUMPREE Funktionsbeschreibung:

Allgemein:

Die Funktion IUMPREE dient der Abspeicherung von Größen zum Nachweis des "In-Use-Monitor Performance Ratios" (IUMPR)ins EEPROM, die in der Funktion %IUMPRKF berechnet werden. In der Funktion IUMPRKF erfolgt die Aufzeichnung von Informationen über das Laufverhalten von Diagnosen im Alltag gemäß den OBDII-Update Forderungen. Dort wird ein bestimmtes Laufverhalten pro allgemeinem Fahrzyklus gefordert. Die genauen Anforderungen an das Inkrementieren bestimmter Z¨ahler sind in den Gesetzesunterlagen der CARB spezifiziert.
CARB fordert, dass "General Denominator", "Ignition Cycle Counter", "Numerator" und "Denominator" nur im Falle einer Neuprogrammierung oder - falls die Werte im "Keep Alive Memory" (KAM) gehalten werden - bei einem Powerfail, resetiert werden dürfen. Allerdings ist für eine langfristige Sicherung der Daten die Abspeicherung im EEPROM notwendig, um nach einem Powerfail die Daten zu restaurieren. Dann bleiben die ermittelten Werte der Diagnosen in Mode$09 erhalten.
Die Abspeicherung erfolgt im SG-Nachlauf und die Restaurierung in der SG-Initialisierung durch diese Funktion.

Um eine Quasi-Doppelablage der Daten zu realisieren, werden die Daten in 2 in sich konsistenten Blöcken in den EEPROM geschrieben. Die Konsistenz wird für jeden Block durch eine Checksumme über gesamten Block überwacht.
Aufgabe der IUMPREE-Funktion ist es, bei Fehlerfall in der SG-Initialisierung die Daten aus dem neuesten und in sich konsistenten Block des EEPROMs wiederherzustellen bzw. im Nachlauf das Kopieren der Arbeitsdaten in den EEPROM-Mirror und das Berechnen der Checksummen vorzubereiten (Daten werden in den EEPROM-Mirror geschrieben, das Schreiben in den EEPROM selbst erfolgt durch eine spezielle Funktion des EEPROM-Treibers im Nachlauf.
Das Alter der Daten wird anhand von Zeitstempeln am Anfang und Ende eines Blockes bestimmt. Diese Stempel dienen gleichzeitig als schnelle Überprüfung der Blockkonsistenz (Identische Werte = Block wurde komplett geschrieben).
Im Nachlauf werden vor dem Schreiben des EEPROM-Mirrors die AD-Daten, die aktuelle Checksumme und die Zeitstempel in den EEPROM-Mirror kopiert. Danach startet der EEPROM-Treiber die normale Speicherung in den EEPROM.

Probleme und Unterschiede zur ME(D)9:

Die max. Menge der Daten und vor allen Dingen die zwingende In-sich-Konsistenz der Daten lassen eine Doppelablage der Daten wie sie bisher im EEPROM-Treiber realisiert ist, nicht mehr zu. Im Gegensatz zur ME9 kann der EEPROM-Treiber in der ME7-Welt die Doppel-Ablage nur Page-weise organisieren. D. h. beim Schreiben einer Seite im EEPROM wird sofort im Anschluss die dazugehörige Spiegelseite geschrieben, die der normalen Seite unmittelbar folgen muss. In der ME9-Welt kann der Spiegel in einem getrennten Block definiert werden und wird erst nach kompletten Schreiben des normalen Block beschrieben.

Sollte bei der ME7 der Nachlauf und damit das Schreiben des EEPROMs unterbrochen werden, würde bei bisheriger Doppelablage des Treibers ein Teil des EEPROM-Speicher aus Daten vom Zeitpunkt n+1 und der andere Teil aus Daten vom Zeitpunkt n bestehen. Die Abspeicherung und Sicherung einer einzelnen Seite wäre zwar korrekt, in der Gesamtheit der Zähler und Nenner würde aber eine Inkonsistenz vorhanden sein.

Mechanismus zur Doppelablage

Durch die oben aufgeführte Problematik muss die Doppelablage der Daten durch zusätzliche Software (IUMPREE) organisiert werden. Es werden 2 Blöcke mit einfacher Ablage sequentiell im EEPROM angelegt. Diese Daten bestehen aus einer Blockverwaltungsseite (Version, Blockchecksumme, Zeitstempel1), mehreren Seiten mit den Zählern und Nennern, sowie einer Blockende-Seite (Zeitstempel2). Beide Blöcke haben den identischen Aufbau.

Die Mirrorseiten der Blöcke werden im Nachlauf mit den aktuellen Daten beschrieben.
Sollte nun das Schreiben in den EEPROM im Nachlauf unterbrochen werden, so ist zwar der aktuelle Block inkonsistent, doch kann im Falle eines zusätzlichen Powerfail die Restaurierung der Daten aus dem anderen noch unversehrten Block erfolgen.

2.) Aufbau der Blöcke

B1:
Version
Block-Checksumme
Zeitstempel (Anfang)
Ign-C, Gen-Den, Zähler 0 Nenner 0,
.....
Zähler 60 Nenner 60
Zeitstempel (Ende)

B2:
Version
Block-Checksumme
Zeitstempel (Anfang)
Ign-C, Gen-Den, Zähler 0 Nenner 0,
.....
Zähler 60 Nenner 60
Zeitstempel (Ende)

Ein konsistenter Block muss 2 Bedingungen erfüllen:

1.) Die Zeitstempel am Anfang und Ende eines Blocks müssen identisch sein. Mit diesem schnellen Vergleich kann ein Abbruch innerhalb des Blocks erkannt werden.
2.) Die Summe aller Zähler, Nenner sowie Ign-C und Gen-Den muss mit der Checksumme identisch sein. Mit dieser Checksumme sollen korrupte oder falsche beschriebene Seiten des EEPROMs erkannt werden.

Um die Ressourcen des EEPROMs optimal zu nutzen, können auf einer EEPROM-Seite max. 7 Word-Grössen gespeichert werden. 2 Bytes einer Seite werden als Seitenchecksumme verwendet. (7x2 Byte +2 Byte CS = 16Byte)

Arbeitsdaten
Als Arbeitsdaten werden hier die Arrays imp_num[0..SY_NIMPREC], im_den[0..SY_NIMPREC] sowie die 2 Einzelgrößen cntgnden_w und cntigncy_w bezeichnet, die von der Funktion iumprkf zur Verfügung gestellt und bearbeitet werden. Eine weitere Größe stellt der Zeitstempel der Arbeitsdaten dar, der von der Funktion IUMPREE definiert und verwaltet wird.

Block-Checksumme
Die Blockchecksumme ist als Word-Größe definiert und wird im Nachlauf während des Kopierens der Arbeitsdaten in den EEPROM-Mirror errechnet und in den Block-Verwaltungsseiten abgelegt. Diese Checksumme dient dazu, Datenveränderungen innerhalb des Blocks durch Fehler oder Korrekturmaßnahmen des EEPROM-Treibers zu erkennen und den Block als inkonsistent zu erkennen.
Die Berechnung ist eine einfache Addition der beiden Zähler- und Nenner-Arrays sowie der beiden Größen cntgnden_w und cntigncy_w. Diese Blockchecksumme wird in der Initialisierungsphase wieder über die Werte des EEPROM-Mirrors berechnet, welche der EEPROM-Treiber im Normalfall aus dem EERPROM gelesen hat.

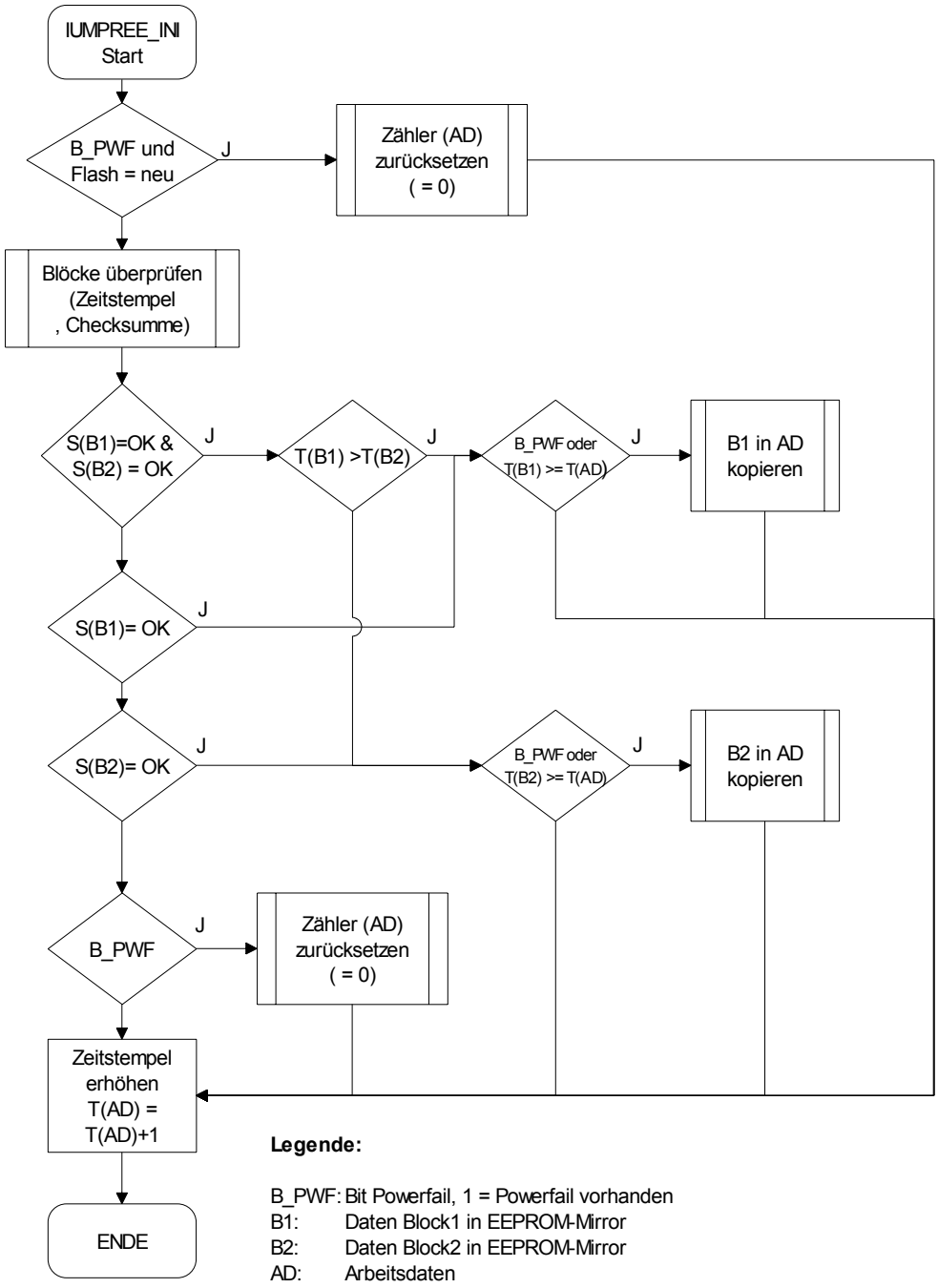
Zeitstempel
Um das „Alter“ der Daten feststellen zu können, besitzen die Blöcke jeweils 2 Long-Größen, die als Zeitstempel verwendet werden. Bei jedem Durchlaufen der Init-Funktion wird dieser Zähler in den Arbeitsdaten hochgezählt und im Nachlauf in die entsprechenden Seiten des EEPROM geschrieben. Dieser Zeitstempel ist wie schon erwähnt in jedem Block doppelt vorhanden - einmal in der Verwaltungsseite und einmal in der letzten Seite der IUMPR-Daten. Damit wird es möglich, eine Unterbrechung des Schreibprogramms im Nachlauf anhand unterschiedlicher Werte der Zeitstempel zu erkennen. Die Ausführung als Long-Größe vermeidet komplizierte Mechanismen im Überlauffall.

3.) Initialisierung oder Wiederherstellung der Daten beim SG-Hochlauf (iumpree_ini)

Bei jedem SG-Start werden die Daten des EEPROMs vom Treiber in die jeweiligen Mirror-Seiten kopiert – vorausgesetzt, der letzte Nachlauf wurde korrekt beendet.
Die Zeitstempel der Blöcke werden miteinander verglichen und falls diese übereinstimmen, die entsprechende Block-Checksumme errechnet und mit der abgelegten Checksumme verglichen. Sind diese beiden Bedingungen erfüllt, gilt ein Block als konsistent.

Nachdem bei beiden Blöcken die Konsistenz geprüft wurde, wird in Abhängig des aktuellen Betriebszustands (Powerfail, Flashprogrammierung neu, Nachlaufunterbrechung) die Arbeitsdaten wieder hergestellt.

Der ganze Ablauf wird im folgendem Diagramm verdeutlicht:



Legende:

B_PWF: Bit Powerfail, 1 = Powerfail vorhanden
B1: Daten Block1 in EEPROM-Mirror
B2: Daten Block2 in EEPROM-Mirror
AD: Arbeitsdaten
Flash: Kennung, ob Flashprogrammierung vorliegt, 1: Flashprogrammierung vorhanden, Neuinitialisierung notwendig
S(Bx): Status Block x, 1 = Stempel und Checksumme OK
T(Bx): Wert des Zeitstempels von Bx
T(AD): Wert des Zeitstempels der AD im NV-RAM

4.) Datenaufbereitung im SG-Nachlauf

Noch bevor die Daten im EEPROM-Mirror durch den Treiber in den EEPROM geschrieben werden, werden die zu sichernden IUMPR-Daten durch die Funktion iumpree_10msSwOff in der Hinsicht aufbereitet, dass zum einen die aktuellen Arbeitsdaten in die entsprechenden Seiten des EEPROM-Mirrors kopiert werden und zum anderen die neuen Blockchecksummen berechnet und zusammen mit den Zeitstempel ebenfalls in den EEPROM-Mirror abgelegt werden. Das „Befüllen“ der Mirror-Seiten findet abhängig von den Stati der beiden Blöcke beim SG-Hochlauf statt. Diese Aufbereitung der Daten wird einmalig und vor dem ersten Aufruf des EEPROM-Treibers im Nachlauf durchgeführt.

Sollte Block2 bei SG-Start als inkonsistent erkannt worden sein, werden in diesem Nachlauf die Daten des Block1 nicht überschrieben, damit falls im nächsten SG-Hochlauf zusätzlich noch ein Powerfail vorliegt, die Arbeitsdaten aus dem verbleibenden konsistenten Block1 wiederhergestellt werden können.

Ein Ausnahme bildet der Fall, dass in Block2 ein Checksummenfehler vorlag und die EEPROM-Zellen evt. physikalische Fehler vorweisen, weshalb eine zuverlässige Sicherung in diesem Block nicht mehr gegeben ist und die Arbeitsdaten dann auch zwingend in Block1 gespeichert werden müssen.

Wie zuvor schon beschrieben, werden im Nachlauf die Arbeitsdaten in den EEPROM-Mirror kopiert, die dazugehörige Block-Checksumme gebildet und der Zeitstempel der Arbeitsdaten in die entsprechenden Seiten kopiert.

5.) Ausnahmen und Besonderheiten

Bei maximaler Anzahl Zähler und Nenner (SY_NIMPREC = 61) muss für die Abspeicherung der Daten in den EEPROM (bei einer durchschnittlichen Zeit von 10 bis 15ms pro Seite) mit einer Nachlaufzeitverlängerung von ca. 400 – 600ms gerechnet werden. Im Maximalfall belegen die IUMPR-Daten 2x 20 Seiten á 16 Byte, woraus sich diese Verlängerung von ca. 400 – 600ms ergibt. Damit der Nachlauf korrekt beendet wird, sollte die Mindestnachlaufzeit für den EEPROM (2k EEPROM= 4s) eingehalten werden.
Abhängig davon , ob der Nachlauf aus Sicht des EEPROM-Treibers korrekt beendet wurde, wird im SG-Hochlauf der EEPROM-Inhalt in den dazugehörigen Mirror kopiert. Lag ein Abbruch des Nachlaufs vor und ist im nächsten SG-Hochlauf kein Powerfail vorhanden, lässt der EEPROM-Treiber die Inhalte des EEPROM-Mirrors unangetastet. Die IUMPREE-Routinen arbeiten mit dem EEPROM-Mirror, weshalb in diesem Fall nicht der wirkliche Zustand des EEPROMs wiedergegeben wird und die Prüfungen des Zeitstempel sowie der Checksumme fehlerfrei durchgeführt werden.
Die aktuellen Arbeitsdaten bleiben somit erhalten, werden bei einem zusätzlichen Powerfail aber zerstört, wodurch die Arbeitsdaten dann, falls vorhanden, auf den Stand des letzten konsistenten Blocks zurückfallen. Sollte auch im EEPROM kein konsistenter Block mehr vorhanden sein, werden die Zähler und Nenner zurückgesetzt.