

Groupe de travail sur les réseaux
Requête pour Commentaires : 1662
STD : 51
Rend obsolète : 1549
Catégorie : Standard
Traduction : Yves lescop

W. Simpson, Editor
Daydreamer
juillet 1994

Lycée la croix-rouge - Brest

PPP dans un tramage similaire à HDLC

Statut de ce document

Ce document spécifie un protocole standard d'Internet pour la communauté Internet, et ne sera éprouvé qu'après plusieurs discussions et suggestions. Merci de vous référer à l'édition courante du " Internet Official Protocol Standards " (STD1) pour l'état de standardisation et le statut de ce protocole. La distribution de ce document est illimitée.

Résumé

Le Protocole Point à Point (PPP) [1] fournit une méthode standard pour transporter des datagrammes multiprotocoles au-dessus de liens point à point.

Ce document décrit l'utilisation du tramage comme-HDLC pour les paquets encapsulés par PPP.

Table des matières

1	INTRODUCTION	3
1.1	EXIGENCES	3
1.2	TERMINOLOGIE.....	3
2	EXIGENCES DE LA COUCHE PHYSIQUE.....	4
3	LA COUCHE LIAISON DE DONNÉES.....	5
3.1	FORMAT DE TRAME	5
3.2	MODIFICATION DE LA TRAME DE BASE	7
4	TRAMAGE À OCTET DE BOURRAGE	8
4.1	DRAPEAU INDICATEUR	8
4.2	TRANSPARENCE.....	8
4.3	TRAMES INCORRECTES	9
4.4	REPLISSAGE DE TEMPS	9
4.4.1	<i>Synchrone octet.....</i>	<i>9</i>
4.4.2	<i>Asynchrone.....</i>	<i>9</i>
4.5	CONSIDÉRATIONS DE TRANSMISSION	10
4.5.1	<i>Synchrone octet.....</i>	<i>10</i>
4.5.2	<i>Asynchrone.....</i>	<i>10</i>
5	TRAMAGE À BIT DE BOURRAGE	10
5.1	DRAPEAU INDICATEUR	10
5.2	TRANSPARENCE.....	11
5.3	TRAMES INCORRECTES	11
5.4	REPLISSAGE DE TEMPS	11
5.5	CONSIDÉRATIONS DE TRANSMISSION	11
6	CONVERSION D'ASYNCHRONE À SYNCHRONE.....	12
7	OPTIONS SUPPLÉMENTAIRES DE CONFIGURATION DE LCP.....	12
7.1	TABLE DES CARACTÈRES DE COMMANDE ASYNCHRONE (ACCM).....	12
A.	OPTIONS RECOMMANDÉES DE LCP	14
B.	IDENTIFICATION AUTOMATIQUE DES TRAMES PPP	15
C.	IMPLÉMENTATION RAPIDE DE LA SÉQUENCE DE CONTRÔLE DE TRAME (FCS).....	15
C.1.	GÉNÉRATEUR DE TABLE DE FCS	15
C.2.	MÉTHODE DE CALCUL DE FCS 16 BITS	16
C.3.	MÉTHODE DE CALCUL DE FCS 32 BITS	18
	CONSIDÉRATIONS SÉCURITAIRES	20
	RÉFÉRENCES.....	20
	REMERCIEMENTS	21
	ADRESSE DU COMITÉ.....	21
	ADRESSE DE L'ÉDITEUR.....	21

1 Introduction

Cette spécification prévoit un tramage sur des liens synchrones orientés bit et orientés octet, et des liens asynchrones avec 8 bits de données et aucune parité. Ces liens DOIVENT être bidirectionnels simultanés, mais PEUVENT être dédiés ou avec commutation de circuit.

Un mécanisme d'échappement est indiqué pour permettre à des paramètres tels que XON/XOFF d'être transmis d'une manière transparente au-dessus du lien, et pour enlever les fausses données de contrôle qui peuvent être injectées dans le lien par le matériel et le logiciel intervenants.

Quelques protocoles s'attendent à une transmission sans erreur, et soit fournissent une détection des erreurs seulement sur une base conditionnelle, ou ne la fournissent pas du tout. PPP utilise la séquence de contrôle de trame d'HDLC pour la détection des erreurs. C'est généralement disponible dans des réalisations matérielles, et une implantation logicielle est fournie.

1.1 Exigences

Dans ce document, plusieurs mots sont employés pour signifier les exigences de la spécification. Ces mots sont souvent mis en majuscules.

"DOIT" Ce mot ou l'adjectif "OBLIGATOIRE" signifie que la définition est une condition absolue de cette spécification.

"NE DOIT PAS" Cette phrase signifie que la définition est absolument prohibée dans cette spécification.

"DEVRAIT" Ce mot ou l'adjectif "RECOMMANDÉ" signifie qu'il peut exister des raisons valides dans des circonstances particulières pour ignorer cet item, mais les implications complètes devront être comprises et le cas soigneusement étudié avant de choisir un chemin différent.

"PEUT" Ce mot ou l'adjectif "OPTIONNEL" signifie que cet item est un élément d'un ensemble permis de solutions de rechange. Une implantation qui n'inclut pas cette option DOIT pouvoir interopérer avec une autre implantation qui inclut l'option.

1.2 Terminologie

Ce document utilise fréquemment les termes suivants :

Datagramme : l'élément de la transmission dans la couche réseau (telle que IP). Un datagramme peut être encapsulé dans un ou plusieurs paquets passés à la couche liaison de données.

Trame : l'élément de la transmission à la couche liaison de données. Une trame peut inclure un en-tête et/ou un bas de page, avec un certain nombre d'éléments de données.

Paquet : l'élément de base de l'encapsulation, qui est passé à travers l'interface entre la couche réseau et la couche liaison de données. Un paquet est habituellement englobé dans une trame ; les exceptions sont quand une fragmentation de couche liaison de données est exécutée, ou quand des paquets multiples sont incorporés à une trame simple.

Pair : l'autre extrémité de la liaison point à point.

Rejeté silencieusement : l'implémentation rejette le paquet sans autre transformation ultérieure. L'implémentation DEVRAIT fournir la capacité d'enregistrer l'erreur, y compris le contenu du paquet silencieusement rejeté, et DEVRAIT enregistrer l'événement dans un compteur de statistiques.

2 Exigences de la Couche Physique

PPP est capable de fonctionner à travers la plupart des interfaces DTE/DCE (comme, EIA RS-232-E, EIA RS-422, et CCITT V.35). La seule condition absolue imposée par PPP est la fourniture d'un circuit bidirectionnel simultanément, dédié ou avec commutation de circuit, qui peut fonctionner en mode asynchrone (début/fin), synchrone bit, ou synchrone octet, transparent aux trames de couche liaison de données PPP.

Format d'Interface

PPP présente une interface d'octet à la couche physique. Il n'y a aucune disposition pour que des sous-octets soient fournis ou reçus.

Cadence de Transmission

PPP n'impose aucune restriction concernant la cadence de transmission, autre que celle de l'interface particulière DTE/DCE.

Signaux de Commande

PPP n'exige pas l'utilisation des signaux de commande, tels que la demande pour émettre (RTS), prêt à émettre (CTS), détection de porteuse (DCD), et terminal de données prêt (DTR).

Si disponible, l'utilisation de tels signaux peut permettre de plus grandes fonctionnalités et performances. En particulier, de tels signaux DEVRAIENT être employés pour signaler les événements montant et descendant dans l'automate de négociation d'option de LCP [1]. Quand de tels signaux ne sont pas disponibles, l'implémentation DOIT signaler l'événement montant à LCP sur l'initialisation, et NE DEVRAIT PAS signaler l'événement descendant.

Puisque la signalisation n'est pas exigée, la couche physique PEUT être découplée de la couche liaison de données, cachant les détails transitoires du transport physique. Ceci a des implications pour la mobilité dans des réseaux de radio cellulaire, et d'autres liens à commutation rapide.

En se déplaçant de cellule à cellule dans une même zone, une implémentation PEUT choisir de traiter la zone entière comme un lien unique, bien que la transmission soit commutée parmi plusieurs fréquences. Le lien est considéré comme faisant partie de l'unité centrale de traitement de la zone, plutôt que des différents émetteurs récepteurs de cellules. Cependant, le lien DEVRAIT rétablir sa configuration toutes les fois que le lien est commuté vers une gestion différente.

En raison de la nature en rafale du trafic des données, quelques réalisations ont choisi de déconnecter la couche physique pendant les périodes d'inactivité, et de reconnecter lors de la reprise du trafic, sans informer la couche liaison de données. Les réalisations robustes devraient éviter d'utiliser cette technique trop zélée, puisque le prix de la diminution du temps d'attente est la réduction de la sécurité. Les réalisations DEVRAIENT signaler l'événement descendant toutes les fois que "un temps significatif" s'est écoulé depuis que le lien a été déconnecté. La valeur pour "un temps significatif" est une question de débat considérable, et est basée sur les tarifs, les durées d'établissement de la communication, et les soucis de sécurité de l'installation.

3 La Couche liaison de données

PPP utilise les principes décrits pour la structure de trame HDLC ISO 3309-1979, plus récemment la quatrième édition 3309:1991 [2], qui indique des modifications pour permettre l'utilisation d'HDLC dans les environnements asynchrones.

Les procédures de commande de PPP utilisent les encodages des champs de commandes décrits dans les éléments des procédures HDLC ISO 4335-1979, plus récemment la quatrième édition 4335:1991 [4].

Ceci ne devrait pas être interprété pour indiquer que chaque dispositif des recommandations ci-dessus est inclus dans PPP. Chaque dispositif inclus est explicitement décrit dans les sections suivantes.

Pour rester conforme à la pratique en matière de standard d'Internet, et éviter la confusion pour les personnes chargée de la lecture des RFCs, tous les nombres binaires dans les descriptions suivantes sont dans l'ordre du bit le plus significatif (MSB) au bit le moins significatif, en lisant de gauche à droite, sauf indication contraire. Notez que c'est contraire à la pratique en matière de standard ISO et CCITT qui ordonnent les bits tels qu'ils sont transmis (ordre des bits du réseau). Gardez ceci en mémoire en comparant ce document aux documents de normes internationaux.

3.1 Format de Trame

Un récapitulatif de la structure de trame PPP comme-HDLC est montré ci-dessous. Ce schéma n'inclut pas les bits insérés pour la synchronisation (tels que les bits d'arrêt et de début et pour les liens asynchrones), ni aucuns bits ou octet inséré pour la transparence. Les champs sont transmis de gauche à droite.

Drapeau	Adresse	Commande
01111110	11111111	00000011
Protocole	Information	Remplissage
8/16 bits	*	*
FCS	Drapeau	Remplissage Inter-trame
16/32 bits	01111110	ou Adresse suivante

Les champs protocole, information et remplissage sont décrits dans l'encapsulation du protocole Point à Point [1].

Drapeau

Chaque trame commence et termine avec un drapeau indicateur, qui est la séquence binaire 01111110 (0x7e hexadécimal). Toutes les réalisations vérifient sans interruption cet indicateur, qui est utilisé pour la synchronisation de trame.

Un drapeau indicateur seulement est exigé entre deux trames. Deux drapeaux consécutifs constituent une trame vide, qui est silencieusement rejetée, et non comptée comme erreur de FCS.

Champ adresse

Le champ adresse est un simple octet, qui contient la séquence binaire 11111111 (0xff hexadécimal), l'adresse "Toutes-Stations". Des adresses de station individuelles ne sont pas assignées. L'adresse "Toutes-Stations" DOIT toujours être reconnue et reçue.

L'utilisation d'autres longueurs et valeurs d'adresse peut être définie à un temps postérieur, ou par accord antérieur. Des trames avec des adresses non reconnues DEVRAIENT être silencieusement rejetées.

Champ Commande

Le champ de commande est un simple octet, qui contient la séquence binaire 00000011 (0x03 hexadécimal), la commande "Information non numérotée" (UI) avec le bit de Poll/Final (P/F) réglé à zéro.

L'utilisation d'autres valeurs du champ de commande peut être définie à un temps postérieur, ou par accord antérieur. Des trames avec des valeurs non reconnues du champ de commande DEVRAIENT être silencieusement rejetées.

Champ de Séquence de Contrôle de Trame (FCS)

Le champ FCS est sur 16 bits (deux octets) par défaut. La FCS est transmise avec l'octet le moins significatif d'abord, qui contient le coefficient du terme le plus élevé.

Une FCS de 32 bits (quatre octets) est également définie. Son utilisation peut être négociée comme décrit dans des "extensions LCP de PPP" [5].

L'utilisation d'autres longueurs de FCS peut être définie à un temps postérieur, ou par accord antérieur.

Le champ FCS est calculé sur tous les bits des champs adresse, commande, protocole, information et remplissage, en n'incluant pas tous les bits d'arrêts et de début (asynchrones) ni n'importe quels bits (synchrones) ou octets (asynchrones ou synchrones) insérés pour la transparence. Ceci également n'inclut pas les drapeaux indicateurs ni le champ FCS lui-même.

Quand on reçoit des octets qui sont marqués dans la table des caractères de contrôle asynchrone, ils sont écartés avant de calculer la FCS.

Pour plus d'information sur la spécification de la FCS, voyez les annexes.

L'extrémité des champs information et remplissage est trouvée en localisant le drapeau indicateur et en retirant le champ séquence de contrôle de trame (FCS).

3.2 Modification de la trame de base

Le protocole de commande de lien (LCP) peut négocier des modifications à la norme de la structure de trame comme-HDLC. Cependant, les trames modifiées seront toujours clairement distinguables des trames standard.

Compression des champs Adresse et Commande

En utilisant la norme de tramage comme-HDLC, les champs adresse et commande contiennent les valeurs hexadécimales 0xff et 0x03 respectivement. Quand d'autres valeurs de champs adresse ou commande sont en service, la compression des champs Adresse et Commande NE DOIT PAS être négociée.

Pendant la transmission, les champs comprimés d'adresse et de commande sont simplement omis.

À la réception, les champs adresse et commande sont décompressés en examinant les deux premiers octets. Si elles contiennent les valeurs 0xff et 0x03, on présume qu'elles sont les champs adresse et commande. Sinon, on suppose que les champs ont été comprimés et n'ont pas été transmis.

Par définition, le premier octet d'un champ protocole de deux octets ne sera jamais 0xff (puisque'il n'est pas égal). La valeur du champ protocole 0x00ff n'est pas permise (réservée) afin d'éviter l'ambiguïté quand la compression du champ protocole est permise et que le premier octet du champ d'information est 0x03.

4 Tramage à Octet de bourrage

Ce chapitre récapitule l'utilisation du tramage comme-HDLC avec des liens asynchrones 8 bits et synchrones octet.

4.1 Drapeau indicateur

Le drapeau indicateur indique le début ou la fin d'une trame. Le flux d'octet est examiné sur une base d'octet par octet pour repérer la valeur 01111110 (0x7e hexadécimal).

4.2 Transparence

Un procédé à octet bourrant est utilisé. L'octet d'échappement de commande est défini en tant que binaire 01111101 (0x7d hexadécimal), le bit le plus significatif d'abord.

Au minimum, les implémentations d'émission DOIVENT échapper le drapeau indicateur et les octets d'échappement de commande.

Après calcul de FCS, l'émetteur examine la trame entière entre les deux drapeaux indicateurs. Chaque drapeau indicateur, octet d'échappement de commande, et n'importe quel octet qui est marqué dans la table des caractères de contrôle asynchrone d'envoi (ACCM "Async-Control-Character-Map"), est remplacé par une séquence de deux octets comprenant l'octet d'échappement de commande suivi de l'octet initial sur lequel on effectue une opération de ou exclusif avec la valeur hexadécimale 0x20.

Ceci revient à complémenter le bit 5, lorsque les positions de bits sont numérotées 76543210 (le 6ème bit comme utilisé dans la numérotation ISO : 87654321 -- PRENEZ GARDE en comparant les documents).

Les implémentations de réception DOIVENT correctement traiter toutes les séquences d'échappement de commande.

À la réception, avant le calcul de FCS, chaque octet ayant une valeur inférieure à l'hexadécimal

0x20 est contrôlé. S'il est marqué dans l'ACCM de réception, il est simplement retiré (il a pu avoir été inséré par l'équipement de liaisons informatique intervenant). Chaque octet d'échappement de commande est également retiré, et on effectue une opération de ou exclusif avec la valeur hexadécimale 0x20 sur l'octet suivant, à moins que ce soit le drapeau indicateur (qui interrompt une trame).

Quelques exemples peuvent rendre ceci plus clair. Des données échappées sont transmises sur le lien comme suit :

0x7e est encodé comme 0x7d, 0x5e. (drapeau indicateur)
0x7d est encodé comme 0x7d, 0x5d. (échappement de commande)
0x03 est encodé comme 0x7d, 0x23. (ETX)

Quelques modems avec contrôle de flux logiciel peuvent intercepter DC1 et DC3 sortants en ignorant le 8ème bit (de parité). Ces données seraient transmises sur le lien comme suit :

0x11 est encodé comme 0x7d, 0x31. (XON)
0x13 est encodé comme 0x7d, 0x33. (XOFF)
0x91 est encodé comme 0x7d, 0xb1. (XON avec la parité réglée)
0x93 est encodé comme 0x7d, 0xb3. (XOFF avec la parité réglée)

4.3 Trames Incorrectes

Les trames qui sont trop courtes (moins de 4 octets en utilisant une FCS de 16 bits), ou qui se terminent avec un octet d'échappement de commande suivi immédiatement d'un drapeau indicateur, ou dans laquelle le tramage octet est violé (en transmettant des bits d'arrêt "0" où un bit "1" est attendu), sont silencieusement rejetés, et non comptées comme erreur de FCS.

4.4 Remplissage de Temps

4.4.1 Synchrones octet

Il n'y a aucune disposition pour un remplissage de temps inter-octet.

Le drapeau indicateur DOIT être transmis pendant le remplissage de temps inter-trame.

4.4.2 Asynchrone

Le remplissage de temps Inter-octet DOIT être accompli en transmettant continuellement des bits "1" (état de repos).

Le remplissage de temps inter-trame peut être vu comme un remplissage de temps inter-octet étendu. Faire ainsi peut sauvegarder un octet pour chaque trame, diminuer le retard et augmenter la bande passante. C'est possible puisqu'un drapeau indicateur peut servir de fin de trame et de début de trame. Après avoir reçu n'importe quelle trame, un récepteur inactif sera toujours dans

l'état de début de trame.

Les émetteurs robustes devraient éviter d'utiliser cette technique trop zélée, puisque le prix de la réduction du retard est une diminution de la fiabilité. Les liens bruités peuvent provoquer la réception de caractères erronés et les interpréter en tant qu'élément d'une trame entrante. Si l'émetteur n'envoie pas un nouveau drapeau indicateur d'ouverture avant d'envoyer la prochaine trame, alors cette trame sera ajoutée aux caractères de bruit provoquant une trame incorrecte (avec une fiabilité élevée).

Il est suggéré que les implémentations réalisent les meilleurs résultats en envoyant toujours un drapeau indicateur d'ouverture si la nouvelle trame n'est pas adossée à la dernière. Les émetteurs DEVRAIENT envoyer un drapeau indicateur d'ouverture toutes les fois que le "temps appréciable" s'est écoulé après le drapeau indicateur de fermeture antérieur. La valeur maximum pour le "temps appréciable" est susceptible de n'être plus grande que la cadence de frappe d'une dactylo lente, environ 1 seconde.

4.5 Considérations de Transmission

4.5.1 Synchrones octet

La définition des divers encodages et du brouillage est de la responsabilité du matériel DTE/DCE en service, et est en dehors de la portée de cette spécification.

4.5.2 Asynchrone

Tous les octets sont transmis avec le bit le moins significatif d'abord, avec un bit de départ, huit bits de données, et un bit d'arrêt. Il n'y a aucune disposition pour des liaisons asynchrones sept bits.

5 Tramage à bit de bourrage

Ce chapitre récapitule l'utilisation du tramage comme-HDLC avec des liens synchrones orientés bit.

5.1 Drapeau indicateur

Le drapeau indicateur indique le début ou la fin d'une trame, et est utilisé pour la synchronisation de trame. Le flux binaire est examiné sur une base de bit par bit pour la séquence binaire 01111110 (0x7e hexadécimal).

Le drapeau indicateur "0111111011111110" du "mode zéro partagé" NE DEVRAIT PAS être utilisé. Si non évitable, une telle mise en place DOIT s'assurer que le premier drapeau indicateur détecté (la fin de la trame) est promptement communiqué à la couche liaison. L'utilisation du

mode zéro partagé gêne l'interopérabilité avec les convertisseurs synchrone bit à asynchrone et synchrone bit à synchrone octet.

5.2 Transparence

Après calcul de FCS, l'émetteur examine la trame entière entre les deux ordres d'indicateur. Bits "0" sont insérés après que tous les ordres de cinq un "1" contigus (5 derniers bits y compris de la FCS) pour s'assurer qu'un ordre d'indicateur n'est pas simulé.

À la réception, avant le calcul de FCS, tous les "0" a mordu qui suit directement cinq bits "1" contigus est jeté.

5.3 Trames Incorrectes

Des trames qui sont trop courtes (moins de 4 octets en utilisant la FCS de 16 bits), ou qui terminent avec une séquence de plus de six bits à "1", sont silencieusement rejetées, et non comptées comme erreur de FCS.

5.4 Remplissage de Temps

Il n'y a aucune disposition pour le remplissage de temps inter-octet.

Le drapeau indicateur DEVRAIT être transmis pendant le remplissage de temps inter-trame. Cependant, certains types de liens à commutation de circuits exigent l'utilisation de la marque repos ("1" en continu), en particulier ceux qui calculent les droits basés sur des périodes d'activité de bit. Quand la marque repos est utilisée sur un lien synchrone bit, l'implémentation DOIT assurer au moins 15 bits "1" consécutifs entre les drapeaux pendant la période de repos, et que le drapeau indicateur est toujours généré au début d'une trame après une période de repos.

Ceci diffère de la pratique dans ISO 3309, qui permet une marque de repos de 7 à 14 bits.

5.5 Considérations de Transmission

Tous les octets sont transmis avec le bit le moins significatif d'abord.

La définition des divers encodages et du brouillage est de la responsabilité du matériel DTE/DCE en service, et est en dehors de la portée de cette spécification.

Tandis que PPP fonctionnera sans se soucier de la représentation fondamentale du train binaire, le manque de normes pour la transmission gênera l'interopérabilité aussi sûrement que le manque de normes de liaison données. Aux vitesses de 56 Kbit/s à 2.0 Mbit/s, NRZ est actuellement le plus largement disponible, et sur cette base est recommandé par défaut.

Quand on permet la configuration du codage, NRZI est recommandé comme alternative, en raison de son immunité relative pour signaler des erreurs d'inversion de configuration, et d'occurrences quand il PEUT permettre la connexion sans DSU/CSU coûteux. Malheureusement, le codage de NRZI aggrave le facteur x1 manquant de la FCS 16 bits, de sorte qu'une erreur sur 2^{15} devient non détectée (au lieu d'une sur 2^{16}), et les erreurs triple ne sont pas détectées. Par conséquent, quand NRZI est en service, on recommande que la FCS 32 bits soit négociée, qui inclut le facteur x1.

À des vitesses plus élevées jusqu'à 45 Mbit/s, quelques développeurs ont choisi l'interface synchrone à grande vitesse de norme ANSI [HSSI]. Tandis que cette expérience est actuellement limitée, les développeurs sont encouragés à coopérer dans le choix du codage de transmission.

6 Conversion d'asynchrone à synchrone

Il peut y avoir une certaine utilisation des convertisseurs asynchrone à synchrones (certains construits dans des modems et des interfaces cellulaires), ayant pour résultat une implantation asynchrone de PPP sur une extrémité d'un lien et implantation synchrone de l'autre. Il est de la responsabilité du convertisseur de faire les bourrages des conversions lors du fonctionnement.

Pour permettre cette fonctionnalité, les réalisations synchrones de PPP DOIVENT toujours répondre à l'option de configuration de la table de caractères de commande asynchrone (ACCM) avec le LCP "Configuration-ACK". Cependant, l'acceptation de l'option de configuration n'implique pas que l'implémentation synchrone fera tout le mappage d'ACCM. Au lieu de cela, tout mappage d'un tel octet sera exécuté par le convertisseur asynchrone à synchrone.

7 Options Supplémentaires de Configuration de LCP

Le format d'option de configuration et les options de base sont déjà définis pour LCP [1].

Des valeurs à jour du champ type d'option de LCP sont indiquées dans le RFC "nombres assignés" le plus récent [10]. Ce document concerne les valeurs suivantes :

- 2 ACCM "Async-Control-Character-Map" table des caractères de commande asynchrone

7.1 Table des caractères de commande asynchrone (ACCM)

Description

Cette option de configuration fournit une méthode pour négocier l'utilisation de la transparence de caractère de commande sur des liens asynchrones.

Chaque extrémité du lien asynchrone met à jour deux table des caractères de commande asynchrone (ACCM). L'ACCM de réception est 32 bits, mais l'ACCM d'envoi peut être

jusqu'à 256 bits. Ceci a comme conséquence quatre ACCM distincts, deux dans chaque direction du lien.

Pour des liens asynchrones, l'ACCM reçu par défaut est 0xffffffff. L'ACCM émis par défaut est 0xffffffff, plus les caractères d'échappement de commande et de drapeau indicateur eux-mêmes, plus n'importe quels autres caractères sortants marqués (par configuration antérieure) comme devant probablement être interceptés.

Pour d'autres types de liens, la valeur par défaut est 0, puisqu'il n'y a aucun besoin de mappage.

L'inclusion par défaut de tous les octets inférieurs à l'hexadécimal 0x20 permet à tous les caractères de commande ASCII [6] à l'exception de DEL (effacement) d'être communiqués d'une manière transparente par tout équipement de liaisons informatique connu.

L'émetteur PEUT également envoyer des octets avec des valeurs dans l'intervalle 0x40 à 0xff (excepté 0x5e) par le format d'échappement de commande. Puisque ces valeurs d'octet ne sont pas négociables, ceci ne résout pas le problème des récepteurs qui ne peuvent pas manipuler tous les caractères de non-commande. En outre, puisque la technique n'affecte pas le 8ème bit, ceci ne résout pas des problèmes pour les liaisons qui peuvent envoyer seulement des caractères de 7 bits.

Notez que cette spécification diffère en détail des amendements postérieurs, tels que 3309:1991/Amendment 2 [3]. Cependant, une telle "transparence étendue" est appliquée seulement par "accord antérieur". L'utilisation des méthodes de transparence dans cette spécification constitue un accord antérieur en ce qui concerne PPP.

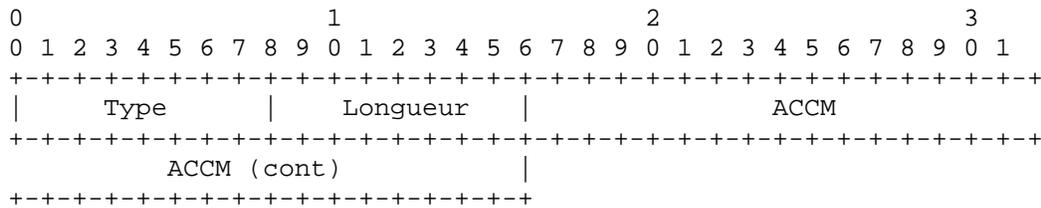
Pour la compatibilité avec 3309:1991/Amendment 2, l'émetteur PEUT échapper à DEL et à des équivalents d'ACCM avec le positionnement du 8ème bit (le plus significatif). Aucun changement n'est exigé de l'algorithme de réception.

Après négociation d'ACCM, l'émetteur DEVRAIT cesser d'échapper DEL.

Cependant, il est rarement nécessaire de tracer tous les caractères de commande, et souvent il est inutile de tracer n'importe quel caractère de commande. L'option de configuration est employée pour informer le pair quels caractères de commande DOIVENT demeurer tracés quand le pair les envoie.

Le pair PEUT toujours envoyer tous les autres octets dans le format tracé, si c'est nécessaire en raison des contraintes connues du pair. Le pair DEVRAIT renier la configuration ("Configuration-NAK") avec l'union logique des ensembles d'octets tracés, de sorte que quand de tels octets sont faussement présentés ils puissent être ignorés à la réception.

Un récapitulatif du format d'option de configuration d'ACCM est montré ci-dessous. Les champs sont transmis de gauche à droite.



Type : 2

Longueur : 6

ACCM

Le champ ACCM est de quatre octets, et indique le jeu de caractères de commande à tracer. La table est envoyée avec l'octet le plus significatif d'abord.

Chaque bit numéroté correspond à l'octet de la même valeur. Si le bit est mis à zéro, alors cet octet n'a pas besoin d'être tracé. Si le bit est mis à un, alors cet octet DOIT demeurer tracé. Par exemple, si le bit 19 est mis à zéro, alors le caractère de commande ASCII 19 (DC3, CTRL-S) PEUT être émis en transparence.

Note : Le bit le moins significatif de l'octet le moins significatif (l'octet final transmis) est numéroté bit 0, et tracerait le caractère de commande ASCII NUL.

A. Options Recommandées de LCP

Les options de configurations suivantes sont recommandées :

Liens à grande vitesse

"Magic Number"	nombre magique
"Link Quality Monitoring"	surveillance qualité du lien
"No Address and Control Field Compression"	pas de compression des champs adresse et commande
"No Protocol Field Compression"	pas de compression du champ protocole

Liens à vitesse réduite ou asynchrones

"Async Control Character Map"	Table de caractères de commande asynchrone
"Magic Number"	nombre magique
"Address and Control Field Compression"	compression des champs adresse et commande
"Protocol Field Compression"	compression du champ protocole

B. Identification automatique des trames PPP

Il est parfois souhaitable de détecter les trames PPP, par exemple pendant une séquence de connexion. Toutes les séquences d'octets suivantes commencent des trames PPP LCP valides :

```
7e FF 03 c0 21
7e FF 7d 23 c0 21
7e 7d DF 7d 23 c0 21
```

Notez que les deux premières formes ne sont pas un nom d'utilisateur valide pour Unix. Cependant, seulement la troisième forme produit une trame correctement sommée de PPP, toutes les fois que 03 et FF sont pris pour des caractères de commande ETX et DEL sans se soucier de la parité (ils sont corrects pour un lien de parité paire) et sont rejetés.

Beaucoup de réalisations traitent ceci en mettant l'interface dans le mode de paquet quand une des formes nom d'utilisateur ci-dessus est détectée pendant la procédure de connexion, sans examiner la somme de contrôle initiale de PPP. La trame entrante initiale de PPP est jetée, mais une demande de configuration est envoyée immédiatement.

C. Implémentation Rapide de la Séquence de Contrôle de Trame (FCS)

La FCS a été initialement conçue avec des réalisations matérielles à l'esprit. Un train binaire séquentiel est transmis sur le fil, la FCS est calculée sur les données séquentielles pendant leur sortie, et le complément de la FCS résultante est ajouté au flux séquentiel, suivi du drapeau indicateur.

Le récepteur n'a aucun moyen de déterminer qu'il a fini de calculer la FCS reçue jusqu'à ce qu'il détecte le drapeau indicateur. Par conséquent, la FCS a été conçue de sorte qu'une forme particulière découle quand l'opération de FCS passe au-dessus de la FCS complétée. Une bonne trame est indiquée par cette valeur de "bonne FCS".

C.1. Générateur de table de FCS

Le code suivant crée la table de consultation employée pour calculer la FCS-16.

```
/*
 * Génération d'une table FCS-16.
 *
 * Drew D. Perkins at Carnegie Mellon University.
 *
 * Code librement emprunté de Mohsen Banan and D. Hugh Redelmeier.
 */
```

```

/*
 * Le polynome générateur FCS-16 : x**0 + x**5 + x**12 + x**16.
 */
#define P      0x8408

main()
{
    register unsigned int b, v;
    register int i;

    printf("typedef unsigned short u16;\n");
    printf("static u16 fcstab[256] = {");
    for (b = 0; ; ) {
        if (b % 8 == 0)
            printf("\n");

        v = b;
        for (i = 8; i--; )
            v = v & 1 ? (v >> 1) ^ P : v >> 1;

        printf("\t0x%04x", v & 0xFFFF);
        if (++b == 256)
            break;
        printf(",");
    }
    printf("\n};\n");
}

```

C.2. Méthode de Calcul de FCS 16 bits

Le code suivant fournit une computation de table de consultation pour calculer la séquence de contrôle de trame pendant que les données arrivent à l'interface. Cette implémentation est basée sur [7], [8], et [9].

```

/*
 * u16 représente un nombre de 16 bits non signé. Ajustez le typedef à
 * votre matériel.
 */
typedef unsigned short u16;

/*
 * FCS table de consultation calculée par le générateur de table.
 */
static u16 fcstab[256] = {
    0x0000, 0x1189, 0x2312, 0x329b, 0x4624, 0x57ad, 0x6536, 0x74bf,
    0x8c48, 0x9dc1, 0xaf5a, 0xbed3, 0xca6c, 0xdbe5, 0xe97e, 0xf8f7,
    0x1081, 0x0108, 0x3393, 0x221a, 0x56a5, 0x472c, 0x75b7, 0x643e,
    0x9cc9, 0x8d40, 0xbfdb, 0xae52, 0xdaed, 0xcb64, 0xf9ff, 0xe876,
    0x2102, 0x308b, 0x0210, 0x1399, 0x6726, 0x76af, 0x4434, 0x55bd,
    0xad4a, 0xbcc3, 0x8e58, 0x9fd1, 0xeb6e, 0xfae7, 0xc87c, 0xd9f5,
    0x3183, 0x200a, 0x1291, 0x0318, 0x77a7, 0x662e, 0x54b5, 0x453c,
    0xbdcb, 0xac42, 0x9ed9, 0x8f50, 0xfbef, 0xea66, 0xd8fd, 0xc974,
    0x4204, 0x538d, 0x6116, 0x709f, 0x0420, 0x15a9, 0x2732, 0x36bb,
    0xce4c, 0xdfc5, 0xed5e, 0xfcd7, 0x8868, 0x99e1, 0xab7a, 0xbaf3,

```

```

0x5285, 0x430c, 0x7197, 0x601e, 0x14a1, 0x0528, 0x37b3, 0x263a,
0xdecd, 0xcf44, 0xfddf, 0xec56, 0x98e9, 0x8960, 0xbbfb, 0xaa72,
0x6306, 0x728f, 0x4014, 0x519d, 0x2522, 0x34ab, 0x0630, 0x17b9,
0xef4e, 0xfec7, 0xcc5c, 0xddd5, 0xa96a, 0xb8e3, 0x8a78, 0x9bf1,
0x7387, 0x620e, 0x5095, 0x411c, 0x35a3, 0x242a, 0x16b1, 0x0738,
0xffcf, 0xee46, 0xdcdd, 0xcd54, 0xb9eb, 0xa862, 0x9af9, 0x8b70,
0x8408, 0x9581, 0xa71a, 0xb693, 0xc22c, 0xd3a5, 0xe13e, 0xf0b7,
0x0840, 0x19c9, 0x2b52, 0x3adb, 0x4e64, 0x5fed, 0x6d76, 0x7cff,
0x9489, 0x8500, 0xb79b, 0xa612, 0xd2ad, 0xc324, 0xf1bf, 0xe036,
0x18c1, 0x0948, 0x3bd3, 0x2a5a, 0x5ee5, 0x4f6c, 0x7df7, 0x6c7e,
0xa50a, 0xb483, 0x8618, 0x9791, 0xe32e, 0xf2a7, 0xc03c, 0xd1b5,
0x2942, 0x38cb, 0x0a50, 0x1bd9, 0x6f66, 0x7eef, 0x4c74, 0x5dfd,
0xb58b, 0xa402, 0x9699, 0x8710, 0xf3af, 0xe226, 0xd0bd, 0xc134,
0x39c3, 0x284a, 0x1ad1, 0x0b58, 0x7fe7, 0x6e6e, 0x5cf5, 0x4d7c,
0xc60c, 0xd785, 0xe51e, 0xf497, 0x8028, 0x91a1, 0xa33a, 0xb2b3,
0x4a44, 0x5bcd, 0x6956, 0x78df, 0x0c60, 0x1de9, 0x2f72, 0x3efb,
0xd68d, 0xc704, 0xf59f, 0xe416, 0x90a9, 0x8120, 0xb3bb, 0xa232,
0x5ac5, 0x4b4c, 0x79d7, 0x685e, 0x1ce1, 0x0d68, 0x3ff3, 0x2e7a,
0xe70e, 0xf687, 0xc41c, 0xd595, 0xa12a, 0xb0a3, 0x8238, 0x93b1,
0x6b46, 0x7acf, 0x4854, 0x59dd, 0x2d62, 0x3ceb, 0x0e70, 0x1ff9,
0xf78f, 0xe606, 0xd49d, 0xc514, 0xblab, 0xa022, 0x92b9, 0x8330,
0x7bc7, 0x6a4e, 0x58d5, 0x495c, 0x3de3, 0x2c6a, 0x1ef1, 0x0f78
};

#define PPPINITFCS16    0xffff /* valeur FCS Initiale */
#define PPPGOODFCS16   0xf0b8 /* bonne valeur FCS finale */

/*
 * Calcul d'une nouvelle fcs donnant la fcs courante et la nouvelle donnée.
 */
u16 pppfcs16(fcs, cp, len)
    register u16 fcs;
    register unsigned char *cp;
    register int len;
{
    ASSERT(sizeof (u16) == 2);
    ASSERT(((u16) -1) > 0);
    while (len--)
        fcs = (fcs >> 8) ^ fcstab[(fcs ^ *cp++) & 0xff];

    return (fcs);
}

/*
 * Comment utiliser la fcs
 */
tryfcs16(cp, len)
    register unsigned char *cp;
    register int len;
{
    u16 trialfcs;

    /* ajout sur la sortie */
    trialfcs = pppfcs16( PPPINITFCS16, cp, len );
    trialfcs ^= 0xffff; /* complement */
    cp[len] = (trialfcs & 0x00ff); /* octet le moins significatif d'abord */
    cp[len+1] = ((trialfcs >> 8) & 0x00ff);
}

```

```

/* test sur l'entrée */
trialfcs = pppfcs16( PPPINITFCS16, cp, len + 2 );
if ( trialfcs == PPPGOODFCS16 )
    printf("Good FCS\n");
}

```

C.3. Méthode de Calcul de FCS 32 bits

Le code suivant fournit une computation de table de consultation pour calculer la séquence de contrôle de trame 32 bits pendant que les données arrivent à l'interface.

```

/*
 * Le polynome générateur de FCS-32 :  $x^{*0} + x^{*1} + x^{*2} + x^{*4} + x^{*5}$ 
 *                                     +  $x^{*7} + x^{*8} + x^{*10} + x^{*11} + x^{*12} + x^{*16}$ 
 *                                     +  $x^{*22} + x^{*23} + x^{*26} + x^{*32}$ .
 */

/*
 * u32 représente un nombre de 32 bits non signé. Ajustez le typedef à
 * votre matériel.
 */
typedef unsigned long u32;

static u32 fcstab_32[256] =
{
    0x00000000, 0x77073096, 0xee0e612c, 0x990951ba,
    0x076dc419, 0x706af48f, 0xe963a535, 0x9e6495a3,
    0x0edb8832, 0x79dcb8a4, 0xe0d5e91e, 0x97d2d988,
    0x09b64c2b, 0x7eb17cbd, 0xe7b82d07, 0x90bf1d91,
    0x1db71064, 0x6ab020f2, 0xf3b97148, 0x84be41de,
    0x1adad47d, 0x6ddde4eb, 0xf4d4b551, 0x83d385c7,
    0x136c9856, 0x646ba8c0, 0xfd62f97a, 0x8a65c9ec,
    0x14015c4f, 0x63066cd9, 0xfa0f3d63, 0x8d080df5,
    0x3b6e20c8, 0x4c69105e, 0xd56041e4, 0xa2677172,
    0x3c03e4d1, 0x4b04d447, 0xd20d85fd, 0xa50ab56b,
    0x35b5a8fa, 0x42b2986c, 0xdbbbc9d6, 0xacbcf940,
    0x32d86ce3, 0x45df5c75, 0xdcd60dcf, 0xabd13d59,
    0x26d930ac, 0x51de003a, 0xc8d75180, 0xbf061116,
    0x21b4f4b5, 0x56b3c423, 0xcfba9599, 0xb8bda50f,
    0x2802b89e, 0x5f058808, 0xc60cd9b2, 0xb10be924,
    0x2f6f7c87, 0x58684c11, 0xc1611dab, 0xb6662d3d,
    0x76dc4190, 0x01db7106, 0x98d220bc, 0xefd5102a,
    0x71b18589, 0x06b6b51f, 0x9fbfe4a5, 0xe8b8d433,
    0x7807c9a2, 0x0f00f934, 0x9609a88e, 0xe10e9818,
    0x7f6a0dbb, 0x086d3d2d, 0x91646c97, 0xe6635c01,
    0x6b6b51f4, 0x1c6c6162, 0x856530d8, 0xf262004e,
    0x6c0695ed, 0x1b01a57b, 0x8208f4c1, 0xf50fc457,
    0x65b0d9c6, 0x12b7e950, 0x8bbeb8ea, 0xfcb9887c,
    0x62dd1ddf, 0x15da2d49, 0x8cd37cf3, 0xfbd44c65,
    0x4db26158, 0x3ab551ce, 0xa3bc0074, 0xd4bb30e2,
    0x4adfa541, 0x3dd895d7, 0xa4d1c46d, 0xd3d6f4fb,
    0x4369e96a, 0x346ed9fc, 0xad678846, 0xda60b8d0,
    0x44042d73, 0x33031de5, 0xaa0a4c5f, 0xdd0d7cc9,

```

```

0x5005713c, 0x270241aa, 0xbe0b1010, 0xc90c2086,
0x5768b525, 0x206f85b3, 0xb966d409, 0xce61e49f,
0x5edef90e, 0x29d9c998, 0xb0d09822, 0xc7d7a8b4,
0x59b33d17, 0x2eb40d81, 0xb7bd5c3b, 0xc0ba6cad,
0xedb88320, 0x9abfb3b6, 0x03b6e20c, 0x74b1d29a,
0xead54739, 0x9dd277af, 0x04db2615, 0x73dc1683,
0xe3630b12, 0x94643b84, 0x0d6d6a3e, 0x7a6a5aa8,
0xe40ecf0b, 0x9309ff9d, 0x0a00ae27, 0x7d079eb1,
0xf00f9344, 0x8708a3d2, 0x1e01f268, 0x6906c2fe,
0xf762575d, 0x806567cb, 0x196c3671, 0x6e6b06e7,
0xfed41b76, 0x89d32be0, 0x10da7a5a, 0x67dd4acc,
0xf9b9df6f, 0x8ebee9f9, 0x17b7be43, 0x60b08ed5,
0xd6d6a3e8, 0xa1d1937e, 0x38d8c2c4, 0x4fdff252,
0xd1bb67f1, 0xa6bc5767, 0x3fb506dd, 0x48b2364b,
0xd80d2bda, 0xaf0a1b4c, 0x36034af6, 0x41047a60,
0xdf60efc3, 0xa867df55, 0x316e8eef, 0x4669be79,
0xcb61b38c, 0xbc66831a, 0x256fd2a0, 0x5268e236,
0xcc0c7795, 0xbb0b4703, 0x220216b9, 0x5505262f,
0xc5ba3bbe, 0xb2bd0b28, 0x2bb45a92, 0x5cb36a04,
0xc2d7ffa7, 0xb5d0cf31, 0x2cd99e8b, 0x5bdeae1d,
0x9b64c2b0, 0xec63f226, 0x756aa39c, 0x026d930a,
0x9c0906a9, 0xeb0e363f, 0x72076785, 0x05005713,
0x95bf4a82, 0xe2b87a14, 0x7bb12bae, 0x0cb61b38,
0x92d28e9b, 0xe5d5be0d, 0x7cdcefb7, 0x0bdbdf21,
0x86d3d2d4, 0xf1d4e242, 0x68ddb3f8, 0x1fda836e,
0x81be16cd, 0xf6b9265b, 0x6fb077e1, 0x18b74777,
0x88085ae6, 0xff0f6a70, 0x66063bca, 0x11010b5c,
0x8f659eff, 0xf862ae69, 0x616bffd3, 0x166ccf45,
0xa00ae278, 0xd70dd2ee, 0x4e048354, 0x3903b3c2,
0xa7672661, 0xd06016f7, 0x4969474d, 0x3e6e77db,
0xaed16a4a, 0xd9d65adc, 0x40df0b66, 0x37d83bf0,
0xa9bcae53, 0xdeb9ec5, 0x47b2cf7f, 0x30b5ffe9,
0xbdbdf21c, 0xcabac28a, 0x53b39330, 0x24b4a3a6,
0xbad03605, 0xcdd70693, 0x54de5729, 0x23d967bf,
0xb3667a2e, 0xc4614ab8, 0x5d681b02, 0x2a6f2b94,
0xb40bbe37, 0xc30c8ea1, 0x5a05df1b, 0x2d02ef8d
};

```

```

#define PPPINITFCS32 0xffffffff /* valeur FCS Initiale */
#define PPPGOODFCS32 0xdebb20e3 /* bonne valeur FCS finale */

```

```

/*
 * Calcul d'une nouvelle fcs donnant la fcs courante et la nouvelle donnée.
 */
u32 pppfcs32(fcs, cp, len)
    register u32 fcs;
    register unsigned char *cp;
    register int len;
    {
    ASSERT(sizeof (u32) == 4);
    ASSERT(((u32) -1) > 0);
    while (len--)
        fcs = (((fcs) >> 8) ^ fcstab_32[((fcs) ^ (*cp++)) & 0xff]);

    return (fcs);
    }

```

```
/*
 * Comment utiliser la fcs
 */
tryfcs32(cp, len)
    register unsigned char *cp;
    register int len;
{
    u32 trialfcs;

    /* ajout sur la sortie */
    trialfcs = pppfcs32( PPPINITFCS32, cp, len );
    trialfcs ^= 0xffffffff; /* complément */
    cp[len] = (trialfcs & 0x00ff); /* octet le moins significatif d'abord */
    cp[len+1] = ((trialfcs >>= 8) & 0x00ff);
    cp[len+2] = ((trialfcs >>= 8) & 0x00ff);
    cp[len+3] = ((trialfcs >> 8) & 0x00ff);

    /* test sur l'entrée */
    trialfcs = pppfcs32( PPPINITFCS32, cp, len + 4 );
    if ( trialfcs == PPPGOODFCS32 )
        printf("Good FCS\n");
}
```

Considérations Sécuritaires

Comme indiqué dans la section des exigences de la couche physique, la couche liaison ne pourrait pas être au courant quand l'état connecté de la couche physique a changé. Ceci a comme conséquence des lacunes de sécurité possibles dues à la sur-dépendance dans l'intégrité et la sécurité des systèmes et des gestions de commutation. Une attaque par insertion pourrait être non détectée. Un attaquant qui est capable de parodier la même identité appelante pourrait être capable d'éviter l'authentification du lien.

Références

- [1] Simpson, W., Editor, "The Point-to-Point Protocol (PPP)", STD 50, RFC 1661, Daydreamer, July 1994.
- [2] ISO/IEC 3309:1991(E), "Information Technology - Telecommunications and information exchange between systems - High-level data link control (HDLC) procedures - Frame structure", International Organization For Standardization, Fourth edition 1991-06-01.
- [3] ISO/IEC 3309:1991/Amd.2:1992(E), "Information Technology - Telecommunications and information exchange between systems - High-level data link control (HDLC) procedures - Frame structure - Amendment 2: Extended transparency options for start/stop transmission", International Organization For Standardization, 1992-01-15.
- [4] ISO/IEC 4335:1991(E), "Information Technology - Telecommunications and information

exchange between systems - High-level data link control (HDLC) procedures - Elements of procedures", International Organization For Standardization, Fourth edition 1991-09-15.

- [5] Simpson, W., Editor, "PPP LCP Extensions", RFC 1570, Daydreamer, January 1994.
- [6] ANSI X3.4-1977, "American National Standard Code for Information Interchange", American National Standards Institute, 1977.
- [7] Perez, "Byte-wise CRC Calculations", IEEE Micro, June 1983.
- [8] Morse, G., "Calculating CRC's by Bits and Bytes", Byte, September 1986.
- [9] LeVan, J., "A Fast CRC", Byte, November 1987.
- [10] Reynolds, J., and J. Postel, "Assigned Numbers", STD 2, RFC 1340, USC/Information Sciences Institute, July 1992.

Remerciements

Ce document est le produit du groupe de travail du Protocole Point à Point du Internet Engineering Task Force (IETF). Les commentaires devraient être soumis à la liste de diffusion ietf-ppp@merit.edu.

Cette spécification est basée sur des RFC précédents, où beaucoup de contributions ont été reconnues.

Le code d'exemple de FCS 32 bits a été fourni par Karl Fox (Morning Star Technologies).

Remerciements spéciaux à "Morning Star Technologies" pour la fourniture de ressources informatiques et d'accès de réseau pour écrire cette spécification.

Adresse du comité

Le groupe de travail peut être contacté par l'intermédiaire du siège actuel :

Fred Baker
Advanced Computer Communications
315 Bollay Drive
Santa Barbara, California 93117

fbaker@acc.com

Adresse de l'Editeur

Des questions au sujet de cette note peuvent également être dirigées vers :

William Allen Simpson
Daydreamer
Computer Systems Consulting Services
1384 Fontaine
Madison Heights, Michigan 48071

Bill.Simpson@um.cc.umich.edu
bsimpson@MorningStar.com