

## 7.2.4 Le MIDI Time Code

Durant l'année 1986, certains membres de la MMA (dont Chris Meyer et Evan Brooks de Sequential Circuits, mais aussi Gerry Lester d'Adams-Smith) décidèrent d'intégrer à la norme MIDI un nouveau code de synchronisation, proposant une ébauche de ce qui allait devenir en avril 87 le MIDI Time Code (MTC).

Si le processus de synchronisation SMPTE/MIDI permet d'asservir un séquenceur à une machine à bande (ATR, VTR) grâce à la conversion d'une référence temporelle absolue (l'heure SMPTE) en une référence temporelle relative (l'horloge MIDI), l'idée première du MIDI Time Code consiste à traduire directement sous forme de messages MIDI le code SMPTE provenant d'une machine maître, afin que l'intégralité d'un réseau puisse bénéficier d'une seule et unique référence horaire, sans passer par une conversion en messages d'horloge.

Une configuration MTC sera donc constituée d'une machine génératrice de code SMPTE (magnétophone, magnétoscope, synchroniseur), d'un convertisseur SMPTE/MTC et de différentes machines esclaves au standard MTC (séquenceurs, boîtes à rythmes, magnétophones...). Nous verrons par la suite que les applications du MIDI Time Code vont bien au-delà d'une simple synchronisation. Commençons tout d'abord par en étudier le langage de base.

### La traduction SMPTE/MTC : les messages temporels

Pour traduire en temps réel les informations SMPTE en informations MIDI, trois messages sont utilisés. Le premier se nomme *MIDI Time Code quarter frame* (quart d'image). Il est constitué d'un octet de statut (F1H de la catégorie system common messages), suivi d'un octet de donnée. En voici le format :

Format :            11110001 (F1H) 0nnndddd

Type :              system common message

Le message MTC quarter frame est une traduction simultanée (en temps réel) de certaines des informations contenues dans un message SMPTE : les heures, minutes, secondes et images par seconde. La conversion en binaire de la valeur de chacune de ces quatre catégories d'informations SMPTE occupe donc respectivement 5 bits (24 valeurs de 0 à 23 pour les heures), deux fois 6 bits (60 valeurs de 0 à 59 pour les minutes et les secondes) et 5 bits (de 24 à 30 valeurs pour les images, en fonction du standard du code). En conséquence, la traduction d'un seul et unique message SMPTE nécessite plusieurs messages MTC quarter frame. Pour maintenir un format homogène, chacune de ces quatre catégories d'informations

SMPTE est codée sur 8 bits (deux demi-octets), même si certains d'entre eux sont inutilisés. Or, un seul octet de donnée (constitué par définition de 7 bits utiles) ne suffit pas à transmettre l'une de ces quatre catégories (8 bits).

L'octet de donnée d'un message quarter frame (constituée, comme toute donnée MIDI, de 7 bits utiles) décomposé en deux groupes de bits : nnn et dddd. Les bits 0 à 3 (dddd) représentent la moitié de l'une des quatre catégories d'informations SMPTE (heures, minutes, secondes ou images par seconde), tandis que les bits 4 à 6 (nnn) précisent de quelle catégorie et de quelle moitié il s'agit.

nnn = 000      poids faible de l'indicateur d'images par seconde (Frame Count LS Nibble)

nnn = 001      poids fort de l'indicateur d'images par seconde (Frame Count MS Nibble)

nnn = 010      poids faible de l'indicateur de secondes (Seconds Count LS Nibble)

nnn = 011      poids fort de l'indicateur de secondes (Seconds Count MS Nibble)

nnn = 100      poids faible de l'indicateur de minutes (Minutes Count LS Nibble)

nnn = 101      poids fort de l'indicateur de minutes (Minutes Count MS Nibble)

nnn = 110      poids faible de l'indicateur d'heures (Hours Count LS Nibble)

nnn = 111      poids fort de l'indicateur d'heures et type de code SMPTE (Hours Count MS Nibble and SMPTE Type)

LS Nibble      demi-octet de poids faible (Least Significant)

MS Nibble      demi-octet de poids fort (Most Significant)

En assemblant les demi-octets de poids fort et de poids faible, nous obtenons bien huit bits significatifs par catégorie d'informations SMPTE :

### *Images par seconde*

xxx yyyyy (0001xxxxy + 0000yyyyy)

xxx      réservé à un usage ultérieur. L'unité émettrice doit forcer ce bit à zéro, et l'unité réceptrice l'ignorer.

yyyyy images de 00000 (0) à 11101 (29)

### Secondes

xx yyyyyy (0011xxyy + 0010yyyy)

xx réservé à un usage ultérieur. L'unité émettrice doit forcer ces bits à zéro, et l'unité réceptrice les ignorer.

yyyyyy secondes de 000000 (0) à 111011 (59)

### Minutes

xx yyyyyy (0101xxyy + 0100yyyy)

xx réservé à un usage ultérieur. L'unité émettrice doit forcer ces bits à zéro, et l'unité réceptrice les ignorer.

yyyyyy minutes de 000000 (0) à 111011 (59)

### Heures et type de code SMPTE

x yy zzzzz (0111xyyz + 0110zzzz)

x réservé à un usage ultérieur. L'unité émettrice doit forcer ces bits à zéro, et l'unité réceptrice les ignorer.

yy type de code SMPTE

00 24 images/seconde

01 25 images/seconde

10 30 images/seconde (Drop Frame)

11 30 images/seconde (Non Drop Frame)

zzzzz heures de 00000 (0) à 10111 (23)

Par exemple, le message SMPTE "08 :51 :21 :12" en 25 images par seconde sera représenté de la manière suivante :

Pour les heures zzzzz = 01000 (08), yy = 01 (25 images/seconde), x = 0

MS dddd + LS dddd = xyyzzzzz = 00101000

MS dddd = 0010

LS dddd = 1000

Hours Count MS Nibble and SMPTE  
Type = 01110010

Hours Count LS Nibble and SMPTE  
Type = 01101000

*Pour les minutes*      yyyyyy = 110011 (51), xx = 00

MS dddd + LS dddd = xxyyyyyy =  
00110011

MS dddd = 0011

LS dddd = 0011

Minutes Count MS Nibble : 01010011

Minutes Count LS Nibble : 01000011

*Pour les secondes*      yyyyyy = 010101 (21), xx = 00

MS dddd + LS dddd = xxyyyyyy =  
00010101

MS dddd = 0001

LS dddd = 0101

Seconds Count MS Nibble : 00110001

Seconds Count LS Nibble : 00100101

*Pour les images*      yyyyyy = 01100 (12), xxx = 000

MS dddd + LS dddd = xxxyyyyy =  
00001100

MS dddd = 0000

LS dddd = 1100

Frame Count MS Nibble : 00010000

Frame Count LS Nibble : 00001100

D'où les huit messages MTC quarter frame suivants :

```
F1H 00001100 (0CH) : image/demi-octet de poids faible
F1H 00010000 (10H) : image/demi-octet de poids fort
F1H 00100101 (25H) : seconde/demi-octet de poids faible
F1H 00110001 (31H) : seconde/demi-octet de poids fort
F1H 01000011 (43H) : minute/demi-octet de poids faible
F1H 01010011 (53H) : minute/demi-octet de poids fort
```

F1H 01101000 (68H) : heure/demi-octet de poids faible  
 F1H 01110010 (72H) : heure/demi-octet de poids fort + type de code SMPTE

Lorsque le code SMPTE est lu dans le sens de la marche, ces huit messages sont envoyés à intervalles réguliers :

F1H 0XH  
 F1H 1XH  
 F1H 2XH  
 F1H 3XH  
 F1H 4XH  
 F1H 5XH  
 F1H 6XH  
 F1H 7XH

En marche arrière, l'ordre est inversé :

F1H 7XH  
 F1H 6XH  
 F1H 5XH  
 F1H 4XH  
 F1H 3XH  
 F1H 2XH  
 F1H 1XH  
 F1H 0XH

En ce qui concerne la fréquence d'émission des messages MTC quarter frame, sachant que huit de ces messages sont nécessaires à l'encodage d'une image SMPTE, nous devrions théoriquement obtenir une densité de 200 messages par seconde (8 x 25) pour un code à 25 images. En réalité, elle est réduite de moitié (100, c'est-à-dire un toutes les dix millisecondes à 25 images/seconde) puisque, comme son nom l'indique, le MTC quarter frame est envoyé tous les quarts, et non tous les huitièmes d'image. En conséquence, deux images SMPTE s'écouleront pendant que le MTC en transmettra une. En 24 et 30 images par seconde, le MIDI Time Code représente les images paires, tandis qu'en 25 images/seconde, il alterne chaque seconde entre images paires et images impaires. Ainsi dans l'exemple suivant, les images 08 :51 :21 :13 et 08 :51 :21 :15 ne sont pas traduites en messages MTC :

SMPTE MTC quarter frame

08 :51 :21 :12 et 0 centième d'image F1H 0CH

08 :51 :21 :12 et 25 centièmes d'image F1H 10H (0CH = 12 images)

08 :51 :21 :12 et 50 centièmes d'image F1H 25H

08 :51 :21 :12 et 75 centièmes d'image F1H 31H (15H = 21 secondes)

08 :51 :21 :13 et 0 centième d'image F1H 43H

08 :51 :21 :13 et 25 centièmes d'image F1H 53H (33H = 51 minutes)

08 :51 :21 :13 et 50 centièmes d'image F1H 68H

08 :51 :21 :13 et 75 centièmes d'image F1H 72H (28H = 8 heures, 25 images par secondes)

08 :51 :21 :14 et 0 centième d'image F1H 0EH

08 :51 :21 :14 et 25 centièmes d'image F1H 10H (0EH = 14 images)

08 :51 :21 :14 et 50 centièmes d'image F1H 25H

08 :51 :21 :14 et 75 centièmes d'image F1H 31H (15H = 21 secondes)

08 :51 :21 :15 et 0 centième d'image F1H 43H

08 :51 :21 :15 et 25 centièmes d'image F1H 53H (33H = 51 minutes)

08 :51 :21 :15 et 50 centièmes d'image F1H 68H

08 :51 :21 :15 et 75 centièmes d'image F1H 72H (28H = 8 heures, 25 images par seconde)

Dans tous les cas, les messages F1H 0XH et F1H 4XH correspondent à un changement d'image SMPTE. Pour qu'une unité MIDI reliée à la sortie d'un convertisseur SMPTE/MTC soit en mesure d'afficher la totalité d'un message SMPTE, elle doit lire une séquence complète de huit messages MTC. Le premier message MTC quarter frame (F1H 0XH), tout comme pour le premier bit des 80 subframes du code SMPTE, tombe physiquement à une frontière de début d'image. Signalons qu'à l'instant précis où le convertisseur SMPTE/MTC reçoit le dernier des huit messages de quart d'image, l'information traduite en MTC a deux images de retard (compensées par le convertisseur) par rapport à la position SMPTE de la bande. Techniquement, une unité esclave synchronisera donc la fréquence de son horloge interne d'après les messages MTC quater frame reçus, en procédant à une interpolation entre chacun d'eux

Côté encombrement du réseau, le MIDI Time Code est assez peu gourmand. Sa densité maximale correspond à la traduction d'un code SMPTE à 30 images/seconde, pour un total de 240 octets par seconde (30 divisé par 2 et multiplié par 8 x 2 octets), soit moins de 8 % de la bande passante MIDI. D'autre part, cette limite de capacité de transmission explique qu'il soit impossible d'envoyer des messages de quart d'image en phase d'embobinage ou de rembobinage rapide.

D'où l'utilité du second message temporel MTC, appelé *full message* (message entier), destiné à encoder d'un seul coup une

heure SMPTE. Il s'utilise dans tout autre mode que le mode de lecture (embobinage, rembobinage, locate, etc.), et appartient à la catégorie real time des messages exclusifs. Son rôle apparaîtra plus clairement lors de l'étude de l'interface MTC-1 Fostex.

*Format :* F0H 7FH <channel> 01H <sub-ID#2>  
hrH mnH scH frH F7H

*Type :* system exclusive real time

F0H statut de message exclusif

7FH catégorie real time

<channel> = 7FH indique que le message est à prendre en compte par l'ensemble du réseau

<sub-ID#1> = 01H identificateur de message MTC

<sub-ID#2> = 01H identificateur du type de message MTC (full message)

hrH heure et type de code SMPTE (0 yy  
zzzz)

yy = 00 : 24 images/seconde

yy = 01 : 25 images/seconde

yy = 10 : 30 images/seconde (Drop  
Frame)

yy = 11 : 30 images/seconde (Non  
Drop Frame)

zzzz = heures

mnH minutes

scH secondes

frH images par seconde

F7H EOX

Le troisième message temporel MTC, ou *user bits*, a pour rôle de traduire les 32 bits SMPTE dont dispose l'utilisateur. Rarement utilisés, ils servent à coder des informations telles qu'un numéro de bande, une date d'enregistrement, etc.

*Format :* F0H 7FH <channel> 01H <sub-ID#2>  
u1H u2H u3H u4H u5H u6H u7H u8H  
u9H F7H

*Type :* system exclusive

F0H	statut de message exclusif
7FH	catégorie real time
<channel> = 7FH	indique que le message est à prendre en compte par l'ensemble du réseau
<sub-ID#1> = 01H	identificateur de message MTC Cueing
<sub-ID#2> = 02H	identificateur du type de message MTC (user bits)
	u1H = 0000aaaa
	u2H = 0000bbbb
	u3H = 0000cccc
	u4H = 0000dddd
	u5H = 0000eeee
	u6H = 0000ffff
	u7H = 0000gggg
	u8H = 0000hhhh
	u9H = 000000ji
F7H	EOX

Les demi-octets u1H à u8H ainsi que les bits j et i servent respectivement à encoder les groupes binaires 1 à 8 ainsi que les bits 59 et 43 d'une image SMPTE. Lorsque qu'il s'agira de représenter des informations sur un octet, les huit groupes binaires seront lus dans l'ordre suivant :

hhhhgggg ffffeeee ddddcccc bbbbaaaa

Lorsqu'il s'agira de représenter une heure d'après le format DCB (Décimal Codé Binaire), le groupe huit correspondra aux chiffres des unités du nombre d'images, le groupe sept aux chiffre des dizaines..., et le groupe un au chiffre des dizaines du nombre d'heures.

Voici la façon dont procède le DCB pour coder les chiffres 0 à 9 :

Binaire	Symbole DCB
0000	0
0001	1

0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	inutilisé
1011	inutilisé
1100	inutilisé
1101	inutilisé
1110	inutilisé
1111	inutilisé

Ainsi, l'heure SMPTE 01.21.12.23 se traduira par :

hhhh = 0000

gggg = 0001

ffff = 0010

eeee = 0001

dddd = 0001

cccc = 0010

bbbb = 0010

aaaa = 0011

Les trois messages ci-dessus assurent une fidèle traduction de la norme SMPTE et enrichissent le MIDI d'un code de synchronisation basé sur une métrique temporelle absolue. Outre l'asservissement d'un environnement MIDI, audio et vidéo à une référence horaire commune, les applications prévues par le MIDI Time Code vont bien plus loin. Ce dernier est en effet capable d'automatiser un certain nombre de procédures en envoyant une séquence d'ordres à des unités esclaves intelligentes compatibles MTC (séquenceurs, magnétophones,

etc.), chaque ordre de cette séquence étant accompagné de son heure SMPTE d'exécution.

### Les messages de set-up (MTC cueing)

Reprenons la configuration de départ, à l'intérieur de laquelle une machine maître distribue le code SMPTE à un convertisseur SMPTE/MTC, qui lui-même distribue le code MTC à une série d'unités esclaves. Admettons qu'il existe trois unités esclaves A, B et C, et ajoutons un ordinateur (D), capable de programmer et d'envoyer via MIDI une séquence d'ordres à ces trois unités. Nous pourrions, par exemple, décider de programmer sur cet ordinateur D l'envoi d'ordres x1, x2, y et z à exécuter respectivement par les unités A, A, B et C aux heures suivantes :

```
x1 : 01.10.17.06
x2 : 01.10.19.04
y  : 01.10.03.22
z  : 01.11.00.00
```

Ces ordres sont envoyés de D vers les mémoires de A, B et C, préalablement à la lecture du code MTC. Lorsque ce code est envoyé, les machines A, B et C comparent en permanence l'heure reçue à l'heure d'exécution de chaque ordre. En cas de concordance, l'ordre en question est immédiatement exécuté. L'ensemble des ordres mémorisés par un périphérique MTC se nomme liste d'insertion (*cue list*). Ces ordres sont matérialisés par des messages MTC de set-up (mise en oeuvre), envoyés dans notre exemple de D vers A, B et C. Ils appartiennent à la catégorie non real time des messages system exclusive.

*Format :* F0H 7EH <channel> 04H <sub-ID#2> hrH  
mnH scH frH ffH slH smH <add. info> F7H

*Type :* system exclusive non real time

F0H statut de message exclusif

7EH catégorie non real time

<channel> numéro de canal

<sub-ID#1> 04H : identificateur de message MTC

<sub-ID#2> type de set-up (nature de l'ordre)

00H = special

01H = punch in point

02H = punch out point

03H = delete punch in point

	04H = delete punch out point
	05H = event start point
	06H = event stop point
	07H = event start point with additional info
	08H = event stop point with additional info
	09H = delete event start point
	0AH = delete event stop point
	0BH = cue point
	0CH = cue point with additional info
	0DH = delete cue point
	0EH = event name in additional info
hrH	heure et type de code SMPTE (0 yy zzzzz)
	yy = 00 : 24 images/seconde
	yy = 01 : 25 images/seconde
	yy = 10 : 30 images/seconde (Drop Frame)
	yy = 11 : 30 images/seconde (Non Drop Frame)
	zzzzz = heures
mnH	minutes
scH	secondes
frH	images par seconde
ffH	fractions d'image (00-99)
slH	numéro d'événement (event number) LSB
smH	numéro d'événement (event number) MSB
<add. info>	informations additionnelles
F7H	EOX

Parmi les 128 messages de set-up théoriquement utilisables (<sub-ID#2>), seuls quinze sont actuellement définis par la norme. En voici la signification détaillée.

## **00H special**

Par opposition aux quatorze messages suivants, qui s'adressent à une portion spécifique de l'unité réceptrice (piste, etc.), le message "special" concerne l'unité en question dans son ensemble. Six de ces messages sont actuellement définis, leur numéro étant spécifié par le numéro d'événement (slH/smH). Notons que les messages 01H 00H à 04H 00H ne tiennent pas compte de la référence horaire (octets hrH, mnH, scH, frH, ffH).

### **slH/smH = 00H 00H : time code offset**

En supposant, par exemple, que nous devons synchroniser une bande audio lue par une machine esclave MTC (démarrant à 01.06.58.07) avec une image lue par une machine maître SMPTE (démarrant à 01.02.54.03), il s'avérera nécessaire de décaler la synchronisation de la machine esclave de 4 minutes, 4 secondes et 4 images (01.02.54.03 + 00.04.04.04 = 01.06.58.07). Ce décalage (appelé offset) est encodé dans les octets hrH, mnH, scH et frH.

### **slH/smH = 01H 00H : enable event list**

Ce message ordonne à l'unité réceptrice d'exécuter les événements de la liste d'insertion à réception de l'heure MTC correspondante. Il s'agit en quelque sorte d'une validation de la liste.

### **slH/smH = 02H 00H : disable event list**

Ce message ordonne à l'unité réceptrice d'ignorer les événements de la liste d'insertion, sans pour autant les effacer de sa mémoire.

### **slH/smH = 03H 00H : clear event list**

Ce message ordonne à l'unité réceptrice d'effacer de sa mémoire les événements de la liste d'insertion.

### **slH/smH = 04H 00H : system stop**

Ce message ordonne à l'unité réceptrice de s'arrêter.

### **slH/smH = 05H 00H : event list request**

Lorsque l'unité maître transmet ce message sur un canal identique à celui d'une unité réceptrice, cette dernière envoie sa liste d'insertion à l'unité maître, à partir de l'événement dont l'heure est spécifiée par les octets hrH, mnH, scH, frH et ffH.

**01H/02H : punch-in/punch-out**

Ces messages ordonnent à l'unité réceptrice d'enclencher ou de désenclencher l'enregistrement de l'une des pistes à l'heure spécifiée par les octets hrH, mnH, scH, frH et ffH. Le numéro de piste est codé sur les octets slH/smH (event number). Plusieurs messages de ce type peuvent être envoyés à des heures différentes.

**03H/04H : delete punch-in/delete punch-out**

Ces messages ordonnent à l'unité réceptrice d'effacer de sa mémoire l'un des deux messages précédents, à condition que l'heure spécifiée par les octets hrH, mnH, scH, frH et ffH ainsi que le numéro de piste codé sur les octets slH/smH coïncident.

**05H/06H : event start/event stop**

Ces messages ordonnent à l'unité réceptrice d'activer ou de désactiver un événement continu (il pourra s'agir de la lecture d'un échantillon, d'un mouvement de potentiomètre, etc.) à l'heure spécifiée par les octets hrH, mnH, scH, frH et ffH. Le numéro d'événement est codé sur les octets slH/smH (event number). Plusieurs messages de ce type peuvent être envoyés à des heures différentes.

**07H/08H : event start/event stop with additional informations**

Ces messages sont identiques aux messages précédents, à ceci près qu'ils intègrent une description complémentaire d'événement correspondant à un nombre variable d'octets contenus dans la rubrique <add. info.>.

**09H/0AH : delete event start/delete event stop**

Ces messages ordonnent à l'unité réceptrice d'effacer de sa mémoire l'un des quatre messages précédents, à condition que l'heure spécifiée par les octets hrH, mnH, scH, frH et ffH ainsi que le numéro d'événement codé sur les octets slH/smH coïncident.

**0BH : cue point**

Par opposition aux event start/event stop, ce message ordonne à l'unité réceptrice d'activer un événement ponctuel à l'heure spécifiée par les octets hrH, mnH, scH, frH et ffH. Le numéro d'événement est codé sur les octets slH/smH (event number). Plusieurs messages de ce type peuvent être envoyés à des heures différentes.

## **0CH : cue point with additional informations**

Ce message est identique au message précédent, à ceci près qu'il intègre une description complémentaire d'événement correspondant à un nombre variable d'octets contenus dans la rubrique <add. info.>.

## **0DH : delete cue point**

Ces messages ordonnent à l'unité réceptrice d'effacer de sa mémoire l'un des deux types de messages précédents (0BH, 0CH), à condition que l'heure spécifiée par les octets hrH, mnH, scH, frH et ffH ainsi que le numéro d'événement codé sur les octets slH/smH coïncident.

## **0EH : event name in additional information**

Ce message permet de nommer un événement en ASCII à l'aide des octets d'informations additionnelles.

Les octets contenus dans les informations additionnelles représentent les messages MIDI et les codes ASCII sous forme de demi-octets. Ainsi, pour coder une information MIDI sur trois octets, comme par exemple :

```
9FH 64H 00H (Mi 6 relâché)
```

nous utiliserons trois groupes de deux octets d'informations additionnelles, sous la forme LSB/MSB :

```
0FH 09H 04H 06H 00H 00H
```

Pour la plupart, les messages de set-up que nous venons d'étudier existent sous une seconde forme, dans la catégorie system exclusive real time. De ce fait, un appareil les recevant se doit d'exécuter immédiatement les ordres qu'ils véhiculent (punch-in/out à la volée...). C'est ce qui explique que l'heure ait disparue, ainsi que les types de set-up suivants : 00H (special), avec slH/smH = 00H 00H (time code offset), 01H 00H (enable event list), 02H 00H (disable event list), 03H 00H (clear event list) et 05H 00H (event list request), mais aussi 03H (delete punch in point), 04H (delete punch out point), 09H (delete event start point), 0AH (delete event stop point) et 0DH (delete cue point).

Format :            F0H 7FH <channel> 05 <sub-ID#2> slH smH  
                         <add. info> F7H

Type :             system exclusive real time

```

F0H      statut de message exclusif
7FH      catégorie real time
<channel> numéro de canal
<sub-ID#1> 05H : identificateur de message MTC cueing
<sub-ID#2> type de set-up (nature de l'ordre)
          00H = special (seul le numéro d'évènement 04H 00H est
              exploité)
          01H = punch in point
          02H = punch out point
          03H = réservé
          04H = réservé
          05H = event start point
          06H = event stop point
          07H = event start point with additional info
          08H = event stop point with additional info
          09H = réservé
          0AH = réservé
          0BH = cue point
          0CH = cue point with additional info
          0DH = réservé
          0EH = event name in additional info
slH      numéro d'évènement (event number) LSB
smH      numéro d'évènement (event number) MSB
<add. info> informations additionnelles
[F7H      EOF

```

Pour résumer, voici le fonctionnement des différents modes de synchronisation (messages temporels) pris en compte par une configuration MTC :

- Une unité maître (magnétophone, magnétoscope, séquenceur...) émet un code SMPTE vers le convertisseur SMPTE/MTC.
- Le convertisseur SMPTE/MTC traduit le code SMPTE en messages temporels (quarter frame, full message) à destination de périphériques MTC (séquenceurs...) dans le but de les synchroniser. Le principe est identique à celui des messages timing clock + SPP, à ceci près que le MTC travaille en référence temporelle absolue et non plus relative, avec tous les avantages que cela comporte.
- Lorsque l'unité maître est en mode de lecture, le convertisseur transmet des messages MTC quarter frame.
- Lorsque l'unité maître est en mode d'avance ou de retour rapide (fast forward, rewind, shuttle, etc.), le convertisseur envoie des informations de type full message, indiquant grossièrement la position de la bande. Ces informations proviennent généralement des signaux de tachymétrie et de direction (voir rubrique suivante) dès l'instant où la piste de time code devient illisible (c'est-à-dire lorsque la bande n'est plus en contact avec les têtes).
- Lorsque l'unité maître est en mode d'édition (cue mode/la bande est tournée à la main dans un sens ou dans l'autre), le convertisseur envoie des messages de quart d'image.
- En réception du code MTC, et à condition que les périphériques concernés soient intelligents (qu'ils acceptent une liste d'insertion), les actions correspondant aux messages de set-up non real time préalablement programmés ou transmis via MIDI par une unité

spécifique se déclencheront à l'heure dite, tandis qu'avec les messages de set-up real time, elles seront immédiatement exécutées.

Actuellement, les périphériques MTC intelligents (capables d'interpréter les messages de set-up) sont encore peu nombreux. Par contre, certains séquenceurs reçoivent et émettent les messages temporels de synchronisation (quarter frame, full message).

*Figure 7.9 : Les réglages de synchronisation du séquenceur Cubase.*

Pour conclure l'étude de ces différents modes de synchronisation, voici la manière dont le séquenceur Cubase gère la synchronisation. A la base, pour l'ensemble de ses opérations, il utilise une référence temporelle absolue : un time code. Celle-ci peut être générée de manière interne par l'ordinateur, provenir de la prise MIDI-in sous forme de MIDI Time Code, ou d'un convertisseur dédié (SMP24, SMP2, Timelock, MIDEX+) sous forme SMPTE. De son côté, indépendamment de ces trois réglages, le tempo peut être généré en interne ou provenir du mode human sync ou de la prise MIDI-in (timing clock + SPP). En ce qui concerne l'émission, même s'il est synchronisé sur un format différent, Cubase est capable de transmettre les deux types de synchronisation MIDI (timing clock + SPP, MIDI Time Code) et de générer les quatre types de code SMPTE par l'intermédiaire des convertisseurs précités. Une implantation tout à fait exhaustive.