Application Note DK9222-0213-0063

Oversampling

Reading SMPTE timecode information

This application example provides basics on the SMPTE timecode and contains a TwinCAT sample code for extracting the time information sent by a SMPTE master. Using an EL3702 EtherCAT oversampling terminal it is possible to "listen" to the analog signal the SMPTE master broadcasts to the slaves. For the sample code see DK9222-0213-0063_TwinCAT_sample.zip.

Preamble

Please keep in mind that in this document the term **frame** is used in a double meaning:

– Part or unit of a protocol.
– Snippet or single picture as unit of a movie.

What is SMPTE-Timecode?

Defined by the **S**ociety of **M**otion **P**icture and **T**elevision **E**ngineers, the SMPTE timecode is a standard to provide an absolute time reference for media metadata. Mainly it is used to synchronise different layers in media productions. A SMPTE frame consists of 80 bits and is always broadcast from a SMPTE master to the slaves. Via an analog signal ($\pm1…10$ V), the SMPTE timecode provides time information in a binary coded decimal identification on an absolute 24-hour time base (hh:mm:ss:ff). Depending on the signal coding, there are different rates of frames per second (fps); the most common values are 24, 25, and 30 fps.
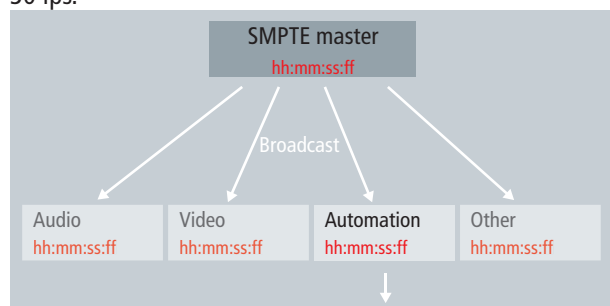

Fig. 1　SMPTE timecode: Broadcasting an absolute time reference to the slaves

Application Note DK9222-0213-0063
Oversampling

Protocol details | Duration of a single bit

The first step is to calculate the duration of a single bit in relation to the frame rate. Because of the different picture frame rates (24, 25, 30), the duration of a single bit differs and is calculated as follows:

Frames per second	Duration of one bit	Duration of a frame
30	416⅔ µs	80 x 416⅔ µs = 33⅓ ms
25	500 µs	80 x 500 µs = 40 ms
24	520 µs	80 x 520 µs = 41⅔ ms

Table 1 Duration of one bit

Protocol details | High or low bit

The second step is to count the transitions within this duration. The bits are encoded as biphase: A „0" bit has a single transition at the start of the bit period. A „1" bit has two transitions, at the beginning and middle of the period. So by counting the transitions within a defined period of time, it can be determined whether the bit is high or low. When the signal is stable during 417 µs at a rate of 30 fps, it is 0. When there are two constant signals of ~208 µs detected, it is 1.
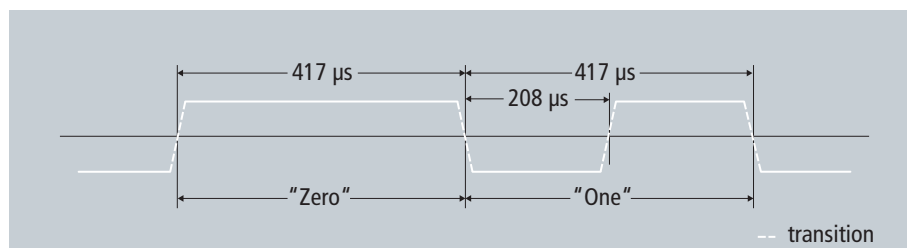

Fig. 2   Sample at 30 fps

Protocol details | Frame setup

Aa a timecode frame always consists of 80 bits, it is 10 byte long. To mark the ending of a frame, the last two bytes follow a defined sequence called **SyncWord**: 0011.1111.1111.1101. In table 2 you can see the setup of a complete timecode frame. In the first row an overview of the time segmentation is given. To enable a fast overview of time-relevant information, the bits containing other information are marked „x".

BECKHOFF New Automation Technology

Application Note DK9222-0213-0063
Oversampling

	Frame units	Frame tens	Second units	Second tens	Minute units	Minute tens	Hour units	Hour tens	Sync word (fixed sequence of bits)	
Bit 0	1	10	1	10	1	10	1	10	binary 0	binary 1
Bit 1	2	20	2	20	2	20	2	20	binary 0	binary 1
Bit 2	4	x	4	40	4	40	4	x	binary 1	binary 1
Bit 3	8	x	8	x	8	x	8	x	binary 1	binary 1
Bit 4	x	x	x	x	x	x	x	x	binary 1	binary 1
Bit 5	x	x	x	x	x	x	x	x	binary 1	binary 1
Bit 6	x	x	x	x	x	x	x	x	binary 1	binary 0
Bit 7	x	x	x	x	x	x	x	x	binary 1	binary 1
	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8	Byte 9	Byte 10

Table 2 Binary coded time information in the SMPTE frame. Bits containing other information than timecode are marked x.

Analysing a frame

In the following figures two sections of a frame are decoded as an example. Figure 3 shows the complete SMPTE frame, figure 4 focusses on a frame detail (the frame units and the frame tens) to analyse the binary coded time information and transfer it into the decimal based count of frames.
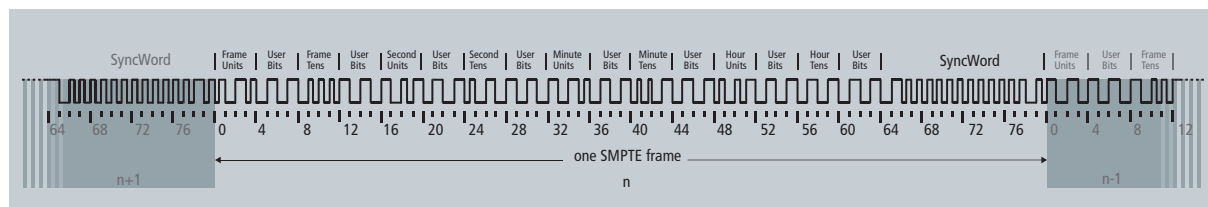

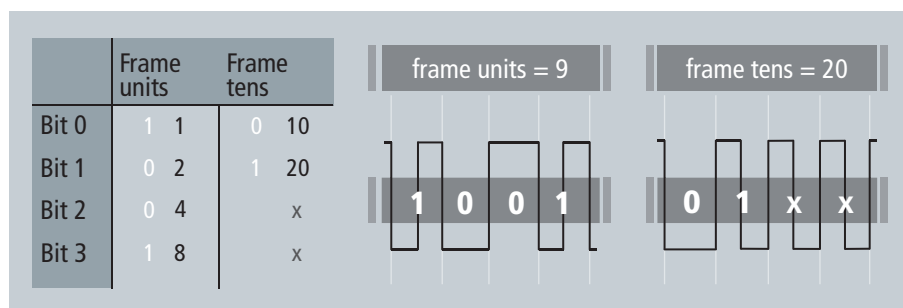
Fig. 3   A single SMPTE sample frame

Fig. 4   Bit sequence 1001 | frame unit = 9; bit sequence 01xx | frame tens = 20, as both last bits do not give information about time

To obtain the complete frame value, both values are added: 20 + 9 = 29. It is the 29th frame in this second. Extracting further time data out of the complete sample frame shown in fig. 3 results in: 05:38:14:29. Having understood the coding of the frame,

BECKHOFF New Automation Technology

Application Note DK9222-0213-0063
Oversampling

you can now focus on a suitable hardware setup to use in conjunction with the TwinCAT PLC sample code. The setup includes an external SMPTE timecode master and an EL3702 EtherCAT terminal.

TwinCAT Sample Program

The file TwinCAT sample.zip contains a program sample to extract time information out of a SMPTE frame. It was programmed for the following hardware setup:

– EL3702 | 2-channel analog input terminal ±10 V with oversampling
– EK1100 | EtherCAT Coupler
– CX5010 | Embedded PC series with Intel® Atom™ processor; or any Beckhoff IPC with a higher performance processor, see chapter **Limitations**
– Video Binloop | External SMTPE device from Alcorn McBride Inc.

EL3702
1 | Input1+
2 | Input1-
3 | GND

SMPTE output
9 | SMPTEOUT+
8 | SMPTEOUT-
5 | GND

CX5010
EL3702
SMPTE output
(D-sub 9-pin, female)
EK1100
Video Binloop
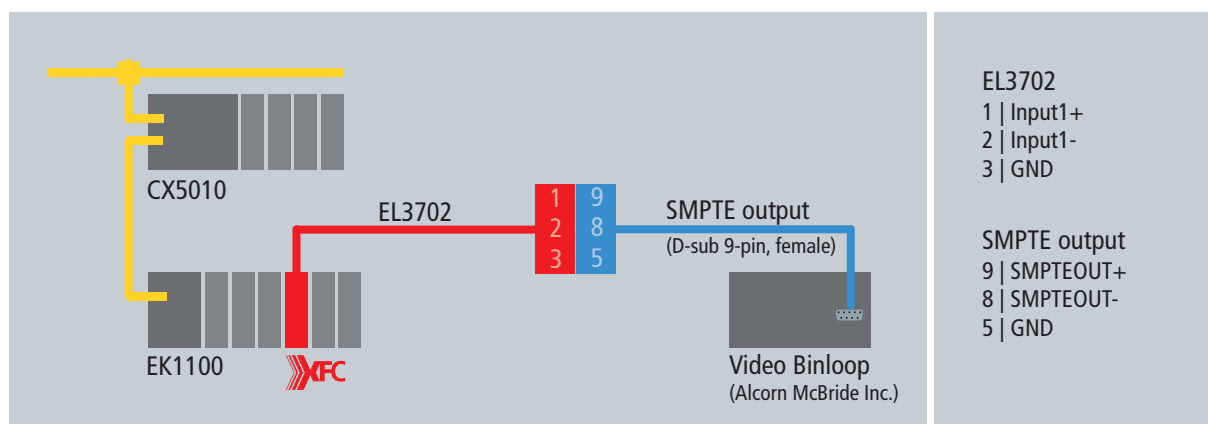(Alcorn McBride Inc.)

Fig. 5   Hardware setup and wiring to connect the timecode master to the EL3702

The SMPTE connector provides both an input and output for the SMPTE interface of the Binloop. The balanced input is 600 ohm transformer-isolated and will accept levels between 1…5 V_{PP}. The balanced output level defaults to 0 dBV into 150 ohms, but can be adjusted between 0…4 dBV using the front-panel menu of the Binloop.

Application Note DK9222-0213-0063
Oversampling

The following table gives an overview about the settings of the Video Binloop, which are not configured in TwinCAT but in the external device.

Setting	Description	Value
SMPTE	Mode Determines whether the SMPTE Interface will generate timecode, or read external timecode.	Gen
Frame rate	The frame rate in which SMPTE is generated.	23,976
Preroll Time	Initial time of SMPTE generator when it is first enabled.	00:00:00.00
Start Time	Time that SMPTE generator returns to after it reaches the end time (when looping)	00:00:00.00
End Time	Time when SMPTE generator either stops or loops back to start time.	00:00:00.00
Loop Mode	Determines whether SMPTE generator loops continuously from start to end time.	enabled
Powerup Mode	Determines whether SMPTE is enabled or disabled when Binloop powers up.	Start
Restart Mode	Determines how SMPTE generator responds when it is told to start again. It can either restart or ignore the start command.	enabled
Idle Mode	Determines if SMPTE output is muted or active when the SMPTE Interface is idle.	SMTPE Active
Self Gen Mode	Determines how long SMPTE reader will free-wheel when external SMPTE is lost.	disabled
Output Level	Configures the voltage level (in dB) of the SMPTE output.	4 dBV

Table 3 Binloop parameters (from manual at Alcorn McBride Inc. website) are not configured in TwinCAT

Note:

Theoretically the information is readable at all frame rates, but the sample program is designed for 24 fps and has not been tested for other frame rates. Furthermore only the time information is read out, not the user bits, the flags and other information transmitted with the frame.

BECKHOFF New Automation Technology

Application Note DK9222-0213-0063

Oversampling

TwinCAT sample

In the file DK9222-0213-0063_TwinCAT_sample.zip, the TwinCAT sample program is given as **SMTPE_Timecode.pro**.

With the code of the program sample, six steps are provided to extract the time information out of the SMPTE frame:

- – Conditioning the EL3702
- – Detecting the transitions
- – Analysing the transitions
- – Identifying high and low bits
- – Analysing the bitstream
- – Extracting the bits

1 | Conditioning the EL3702

The EtherCAT Oversampling Terminal EL3702 listens to the SMPTE output signal and provides an array of values to the TwinCAT PLC. By default the cycle time of the PLC is set to 1 ms and the oversampling rate of the EL3702 is set to 100. Therefore the signal is split into samples with a size of 10 µs, and 40…50 samples for each timecode bit can be analysed.

2 | Detecting the transitions

In the array **Tab_EL3702_Doublecycles** the two last cycles are stored: the samples of the current cycle (n) and the cycle before that (n-1). In each PLC cycle, the program edits the current cycle samples (n). The samples of n-1 are used to compare the current state of the timecode input with the past state. In that way it is possible to check whether a change of sign has occurred in the periods n-1 to n. When the sign is changing, a transition is detected.

IF   Tab_EL3702_Doublecycles[i-1] * Tab_EL3702_Doublecycles [i] < 0

     (* The analog signal changes sign --> there is an edge *)

OR   Tab_EL3702_Doublecycles [i-1] <> 0 AND Tab_EL3702_Doublecycles [i] = 0 THEN

3 | Analysing the transitions

Tab_EL3702_DoublecyclesTrig is an array that shows only the transitions of the timecode signal. Only the transitions of the last cycle samples (n) are managed/handled. **Tab_EL3702_Doublecycles** contains the samples, generated by the EL3702, of the last 2 PLC cycles. Only transitions of the past cycle are checked, marked in red in the figure below.

Application Note DK9222-0213-0063
Oversampling
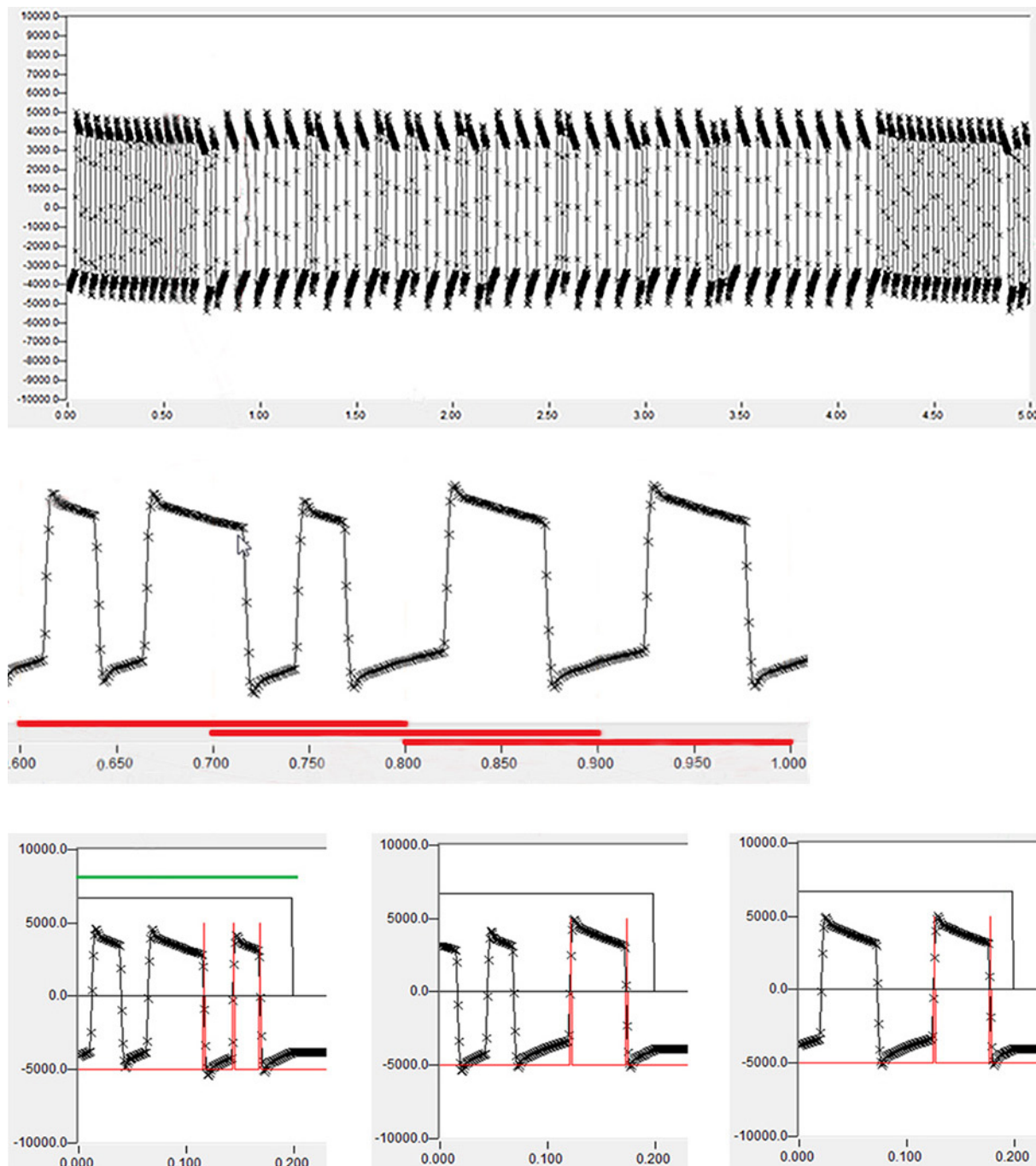
In the pointer **Index_Tab_EL3702**, the positions of the transitions within the last cycle are stored. For the next PLC cycle, this pointer is shifted in order to search for new transitions starting at the position of the last one detected. If there is no further transition detected, an error occurs in **Index_Tab_EL3702_Error**.

BECKHOFF New Automation Technology

Application Note DK9222-0213-0063

Oversampling

4 | Identifying high and low bits

The duration between two transitions indicates the bit state (high = 1 or low = 0). When transitions (changes in the sign of the signal's value) are detected, the time between the transitions is measured and named **ConstantTime**. The value of **ConstantTime** is compared to 330 µs to identify the bit status.

The lower duration limit between two transitions is 417 µs at 30 fps, the upper limit of a half duration is 260 µs at 24 fps (0.5 x 520 µs). This results in a reference of < 330 µs as this fixed limit matches all frame rates when searching for high and low bits:

– If **ConstantTime** > 330 µs, the transitions include a low bit (0). ("duration long")
– If **ConstantTime** < 330 µs, the two transitions measured could be part of a triple identifying a high bit (1). If the next period is < 330 µs, too, the high bit (1) is identified. ("duration short")

5 | Analysing the bitstream

To identify a bit as 1, the **Even** variable is set to **TRUE** at the first detection of "duration short". If **Even** still is **TRUE**, as a "duration long" is detected, an error occurs (**FrameError**), because the number of consecutive ocurrences of "duration short" must always be even. Therefore a high bit (1) can only be identified, if a low bit (0) has occurred before. With **First_BitZero_Reach**, the detection of a high bit (1) is started only, if a low bit (0) has already been detected before.

The detected bits are stored to the array **Tab_TimeCodeFrame** which contains the sequence of bits detected in the timecode signal.

6 | Extracting the bits

As the timecode frame consists of 80 bits, the next step to extract a valid frame is going through the array **Tab_TimeCodeFrame** to find the bit sequence of the SyncWord: If the SyncWord sequence is detected behind position 80 in the array, a valid frame can be extracted to **LastTimeCodeFrame**. From here the information can be decoded to **TimeCodeFrame** with table 2 to read the information inside.

To prevent overflow of the array **TimeCodeFrame**, an error is set in **Index_Tab_TimeCodeFrame_Error** if more than 300 timecode bits are detected without a SyncWord sequence.

Application Note DK9222-0213-0063
Oversampling

Limitations

The complete program causes quite a heavy load on a CPU as a lot of data is copied and calculations are executed. Running the program on a CX5010 at a PLC cycle time of **T#1ms** occupies 50% of the CPU capacity. When expanding the cycle time to **T#2ms** at an oversampling rate of 100, the CPU load will still be 25%. Under these conditions it is however possible to manage 20…25 samples per timecode bit which is quite acceptable.

Please note: When the PLC cycle time is changed, other constants in the program have to be adjusted, too:
- **10** stands for a 10 µs oversampling rate (cycle time/oversampling rate: 1 ms/1000 = 10 µs). If the cycle time is expanded to 2 ms, the oversampling rate should be set to **20**.
- If the oversampling rate in the source code is changed, the factor has to be adapted in the System Manager, too.

Fig. 6   Further constants needs to be adjusted if the cycle time is changed.

With a PLC at **T#1ms** and an oversampling rate of 50, the CPU load is still 40%. A further decrease is not possible as then the number of sample would not be sufficient to generate the signal correctly. **Therefore the use of a CX5020 or a higher** performance **device is recommended.**

Cycle time	Oversampling rate	CPU load
1 ms	50	40 %
1 ms	100	50 %
2 ms	100	25 %

Table 4 Performance limitations with a CX5010

Application Note DK9222-0213-0063
Oversampling

– ± 10 V analog input terminal with oversampling **www.beckhoff.com/EL3702**

– Download the TwinCAT sample

http://download.beckhoff.com/download/document/Application_Notes/DK9222-0213-0063_TwinCAT_sample.zip

– see also Application Example for DMX protocol

http://download.beckhoff.com/download/document/Application_Notes/DK9222-0311-0029.pdf

– PC-based Control for Stage and Show Equipment **www.beckhoff.com/stage**

– PLC and Motion Control on the PC **www.beckhoff.com/TwinCAT**

– SMPTE time code master 'Video Binloop' from Alcorn McBride Inc. **www.alcorn.com**