Simple E1 Software Synchronization Algorithm Based on the Windows PIPE Concept.

Yuriy Dorozhovets, Mykhailo Klymash

Abstract – **This work demonstrate algorithm invented for software realization at Windows PC, and demonstrate principles of E1 stream synchronization handling particular for fractional E1 to.**

Keywords – **E1, G.704, ITU-T, algorithm, PIPE, synchronization.**

I. Introduction

Every telecommunication system requires some low level data transmitting protocol. In Microwave radio relay, Satellite system is very common used E1 Frame to convey any transmitted data. E1 can handle with stream up to 2048kbps and it's still a base for GSM Abis interface, HDLC and DCME links. E1 is based in few concepts:

- Frame alignment signal(FAS)
- 8000 fps
- Timeslot

Data traffic can be conveyed inside E1 in any order but maximum throughput is up to 1984kbps.

During projecting some receive equipment for E1 streams, it is very important to implement simple and efficient E1 synchronization algorithm, and synchronization is only a half of all story. Most important to receiver site is a second part – resynchronization. This term mean recovering of the Frame alignment signal in various points of the time. To provide this possibility in real-time mode it is required some cycle buffer. Almost all modern OS has very convenient way to do this – PIPE. There is two kinds of this objects – named and anonymous. Named pipe is an extension to the traditional pipe concept on UNIX and Unix-like systems, and is one of the methods of inter-process communication. The concept is also found in Microsoft Windows, although the semantics differ substantially. A traditional pipe is "unnamed" because it exists anonymously and persists only for as long as the process is running. A named pipe is system-persistent and exists beyond the life of the process and must be deleted once it is no longer being used. Processes generally attach to the named pipes (usually appearing as a file) to perform inter-process communication (IPC). As you see from this definition, the main purpose of this concept is to organize multithread and multiprocessors interaction, but also you can use PIPEs as very efficient FIFO buffers managed by OS. In this article we will demonstrate how to use it in this mode to provide software E1 synchronization and resynchronization module.

II. E1 Base Synchronization Principles

In accordance to ITU-T G.704(E1) bits 1 to 8 of the both frame types (frame containing the frame alignment signal - FAS, and frame not containing the frame alignment signal - NFAS) are used for sending some signaling/management information, normally not used for any content data transmission. Synchronization signal present in the bits 2…8 in the each FAS frame, format of this signal is in binary 0011011(LSB).

Yuriy Dororzhovets, Mykhailo Klymash - Lviv Polytechnic National University , S. Bandery Str., 12, Lviv, 79013, UKRAINE, E-mail: ydor@ukr.net

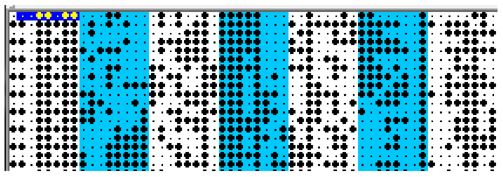
Fig.1 Binary FAS in the real E1 stream.

All other features described in the G.704, like MFAS CRC4 or 4kbprs are not the part of this work, because it's optional E1 features.

So base purpose of the realized software algorithm is FAS search/monitoring. There is wary important notation that proposed technic is designed to work in very hard condition, where synchronization of the signal can be lost many times during receiving signal, and it's very important to understand that output of such kind system will be very complex to future information recovery. But without such algorithms there will be no any possibility to provide any data receiving.

III. Windows PIPE Concept

For simple FIFO realization it is enough anonymous PIPE. It's requiring less overhead than named pipes and easily to use. To create anonymous PIPE call:

```
BOOL WINAPI CreatePipe(
  __out    PHANDLE hReadPipe,
  __out    PHANDLE hWritePipe,
  __in_opt LPSECURITY_ATTRIBUTES lpPipeAttributes,
  __in     DWORD nSize);
```

hReadPipe and hWritePipe – HANDLE for FIFO read/write functions. nSize – FIFO buffer size. lpPipeAttributes – to set binary/massage mode, by default – binary.

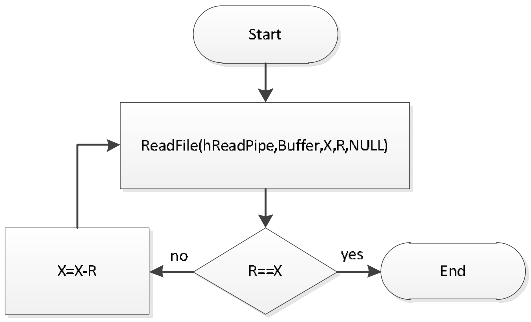

Fig.2 Base PIPE read algorithm.

. When working with this concept bandwidthit's very important to know that starting from Windows XP it is required to control PIPE read side, because Microsoft make some undocumented changes in to PIPE logic and according to this changes all read function must control read size by algorithm.

Another PIPE feature – blocking (by default), require read and write counters to avoid thread blocking during read from empty PIPE.

IV. Algorithm Description

Signal monitoring requires two threads, one realize write to FIFO function, another read and provide search/monitoring logic. When we have some E1 input, read write thread will call function what provide write frames to our FIFO buffer, size of this buffer must be enough to compensate time wasted by search logic. In the first cycle read thread provide read 256 bytes (8 frames (4 FAS frames)) from FIFO using base PIPE read algorithm and search for 3 times FAS signal code sequence much in period from 2 to 32 bytes by step in 2 bytes, this way is very important to avoid wrong sequence detection in fractional E1. After this program will obtain E1 period and copy all good at least 6 good frames to output FIFO buffer. If period not detected drop all data, read new portion, and start from the beginning.

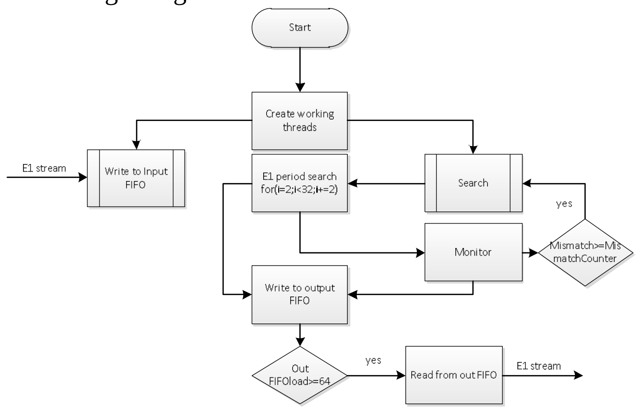

Fig. 3 E1 synchronization algorithm.

Next depending to the input parameter (MismatchCounter) program will monitor stream for detection of sequent MismatchCounter loss of the FAS. To perform better algorithm flexibility proposed to use byte weight table, to avoid wrong starting of the FAS search module. If detected loss of the FAS then call search module to detect new position of the FAS sequence and loop monitoring again. All good frames during this process must be copied to output FIFO and read from it each 64 bytes (one FAS + one NFAS frame). MismatchCounter in accordance to ITU-T recommendation is from 2 to 4, but it can be lager for some types of traffic.

V. Algorithm Performance Research.

It's performed a research of output E1 speed limit according to input stream synchronization loss. Research consist of the two parts, first linear sync loss and second – random sync loss. All research provided at Windows PC based on Intel Core i3 3GHz processor.

The obtained results are shown on the Fig. 4 and Fig. 5:

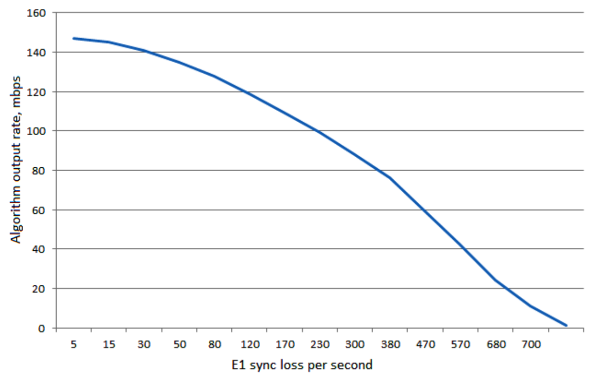

Fig. 4 Output rate in different sync loss ps level.

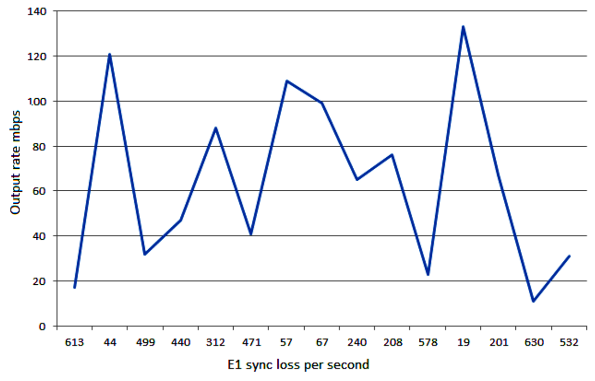

Fig.5 Random sync loss ps level.

All research where provide in correct limits. This mean that to high level of the sync loss per second is uninteresting, because data recovery at these conditions can't take place.

VI. Conclusion

This work described a simple E1 synchronization algorithm designed for software implementation at Windows PC using PIPE concept. After offered results of the performance research and demonstrated no linear dependence of the algorithm performance and FAS loos frequency. Output stream rate, and quality various directly proportional.

Maximum algorithm performance in condition from 1 to 5 FAS loos per second is more than 140Mbps, and this is equivalent to 70 parallel E1. So, proposed algorithm provides opportunity to create software processing units with good performance.

VII. Reference

[1] ITU-T G.704 Synchronous frame structures used at 1544, 6312, 2048, 8448 and 44 736 kbit/s hierarchical levels 10.1998. [Electronic resource] http://www.itu.int/rec/T-REC-G.704/en

[2] Microsoft developers network [Electronic resource] http://msdn.microsoft.com

[3] ITU-T G.706 frame alignment and cyclicre dundancy check (CRC) procedures relating to basic frame structures