Design and Implementation of a Multi-programs Transport Stream Multiplexer

Lei Zhang, Xiaofeng Huang, Yangang Cai
School of Electronic and Computer Engineering
Peking University Shenzhen Graduate School
Shenzhen, China
{lzhang, xfhuang, ygcai}@jdl.ac.cn

Zhuo Li, Huizhu Jia, Xiaodong Xie
National Engineering Laboratory of Video Technology
Peking University
Beijing, China
{zli, hzjia, xdxie}@jdl.ac.cn

***Abstract*—This paper describes design and implementation of a multi-programs transport stream multiplexer which is based on AVS system standard. We have developed an AVS TS compliant multiplexer with special considerations for multi-programs multiplexing. In particular, we construct a monitoring structure which can imitate the behavioral model of T-STD and then use the monitoring information as the key factors for scheduling. We have made efficient verification of AVS and MPEG-2 compliance by RTL simulation.**

***Keywords*—Multi-programs, AVS system standard, Monitoring structure, Behavioral model, Scheduling.**

I. INTRODUCTION

Multiplexer is one of the key installations in the digital television system. We have developed a multi-programs TS multiplexer which adopts AVS transport stream as multiplexing scheme. The multiplexer can multiplex at most 4 programs into a single transport stream. Every program contains one video signal and one audio signal.

AVS system standard [1] expounds how to combine one or multiple audio, video and other elementary data streams into a single transport stream for storage and transmission. The transport stream is mainly for environments where significant errors may occur. System coding must follow the grammar and semantic rules specified by AVS system standard. The standard also provides a method to guarantee the timing and synchronization of video and audio signal.

AVS system standard specifies a "System Target Decoder" (STD) for analyzing the relationship between timing and buffers. For transport streams, it is called "Transport Stream System Target Decoder" (T-STD). The factors of T-STD depend on particular elementary streams. TS multiplexer should generate transport stream which must satisfy the constraints of T-STD.

This paper describes design requirements, operating principles and the RTL architecture of our design. The key point of our design is using T-STD monitors to control TS output for avoiding overflow and underflow in decoder buffers.

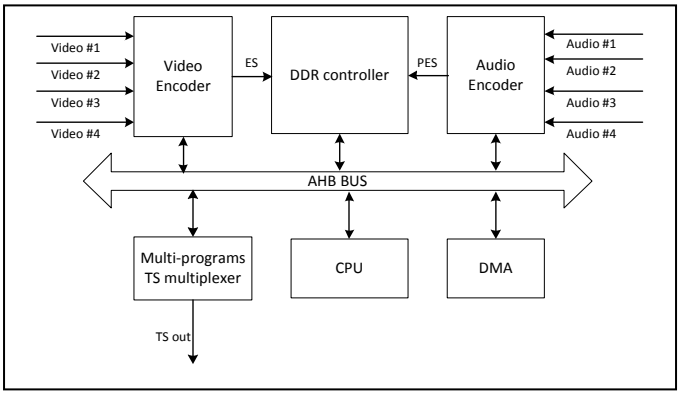
This paper is organized as follows. In section 2，we show the functional requirements and design issue in multi-programs TS multiplexer design. Section 3 presents the hardware architecture and implementation. The process of verification is show in Section 4. Finally, we conclude this paper in section 5.

This work is partially supported by grants from the Chinese National Natural Science Foundation under contract No.61171139 and No. 61035001, and National High Technology Research and Development Program of China (863 Program) under contract No.2012AA011703.

II. FUNCTIONAL REQUIREMENTS AND DESIGN ISSUE

Our multi-programs multiplexer is designed as a hardware module in SOC. It is linked on the AMBA AHB bus as a slave. The architecture of the entire system is show in figure 1.

Fig. 1. Architecture of the Encoder and Multiplexer entire system



There are 4 video signals and 4 audio signals input. The video and audio encoder compress the video and audio signal into ES (Elementary Stream) and PES (Packetized Elementary Stream), then store them in DDR. DMA pushes the ES and PES to TS multiplexer.

The firmware run on CPU calculates the PES header for video. One frame of video is packetized to one PES packet. The firmware also generates PSI&SI for multiplexer.

So, the function of our multiplexer is multiplexing 4 video ES and 4 audio PES into 1 single TS.

Commonly, decoder design must refer to the T-STD. It means that TS must satisfy the constraint of T-STD; otherwise, it will not be decoded correctly. The structure of the T-STD is shown in figure 2.

For all data streams, there are TBs (transport buffer) for them. TB is for storing relevant TS packets. The size of TB is 512 bytes. For video, it is drained by the speed of Rx_n.

$Rx_n = 1.2 \times R_{max}$[profile, level], R_{max}[profile, level] is related to the profile and level of video signal.

For audio, TB is drained by the speed of 2×106bit/s. For system data, TB is drained by the speed of 1×106bit/s.

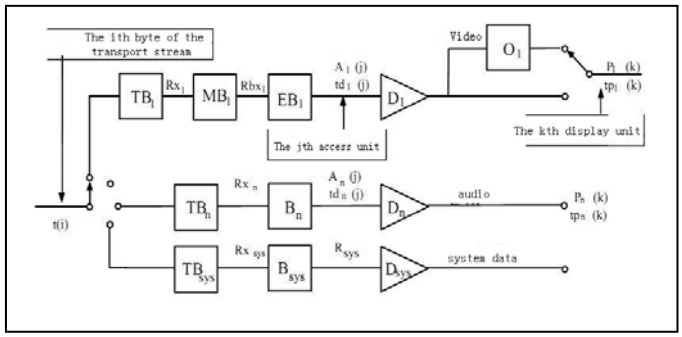


Fig. 2. T-STD structure

For video, there are 2 buffers follow TB, called MB (multiplexing buffer) and EB (elementary buffer). MB is for PES and EB is for ES. Sometimes, the MB and EB can be taken as one buffer. We can call it MB&EB. The data in MB&EB is taken out by frame.

For audio, there is only one buffer follow TB, called B (main buffer). It is for PES. The data in B is taken out by frame.

The TB shall never overflow. The MB&EB and B shall neither overflow nor underflow [2]. Overflow will cause data loss and underflow will cause jitter in display terminal.

To avoid overflow and underflow of the buffers in decoder, the video TS packets, audio TS packets and NULL packets shall be evenly distributed in the transport stream.

The scheduling strategy is the key point of multi-programs TS multiplexer design.

III. HARDWARE ARCHITECTURE AND IMPLEMENTATION

The architecture of our design is shown in figure 3. The functions of the modules are as follows:

AHB slave: this module is the bridge between AHB bus and multiplexer;

Video TS generator: packetize video ES to TS packets and monitor the state of video buffers in T-STD;

Audio TS generator: packetize audio PES to TS packets and monitor the state of audio buffers in T-STD;

PSI/SI generator: packetize PSI and SI to TS packets;

PCR generator: generate PCR and packetize it to TS packets; provide STC to Video TS generator and Audio TS generator;

NULL generator: generate NULL packets;

Scheduler: schedule the TS output and control the output bitrate.

Because there is no buffer in Scheduler module, it does not need to overwrite the PCR in the output path.

The Video TS generator modules and Audio TS generator modules connect to AHB slave module with asynchronous FIFO and connect to Scheduler module with synchronous FIFO. There is an asynchronous block RAM in the PSI/SI generator module for store PSI and SI. Firmware writes this RAM through AHB slave module.

The PSI/SI generator module, PCR generator module, NULL generator module connect to the Scheduler module with FIFO interface but not FIFO. This can simplify the design of Scheduler module and lower the coupling of entire multiplexer.

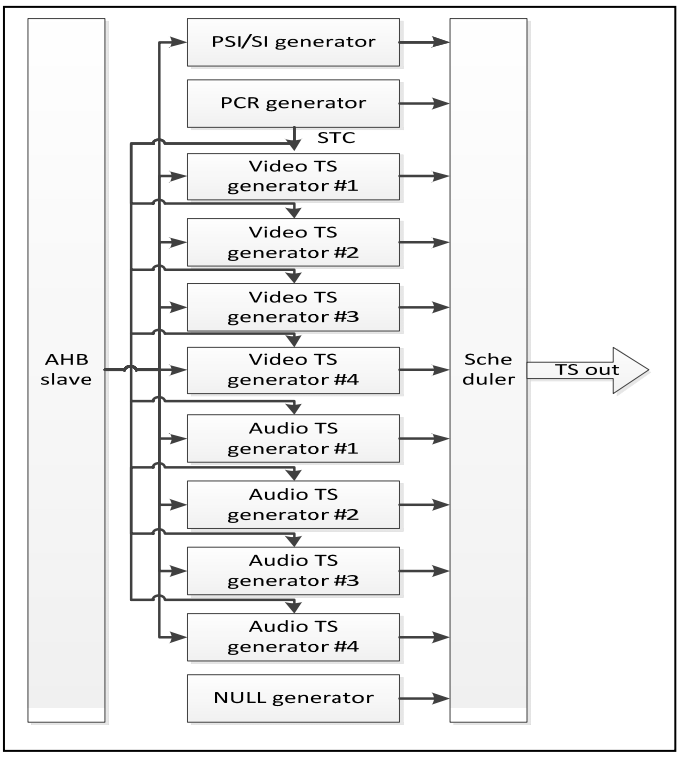


Fig. 3. Top architecture of the multiplexer

In order to solve the problem mentioned in Section 2，we have designed a schedule strategy which is based on the state information of T-STD buffers. We have inserted T-STD monitors in Video TS generators and Audio TS generators. The T-STD monitors can imitate the behavior of the buffers in T-STD and then figure out the data size of the T-STD buffers. The architectures of the Video TS generator and Audio TS generator are shown in figure 4 and figure 5, respectively.

For video signals, firmware calculates ES length, PES header length, DTS and PES header, then push them into FIFOs in Video TS generators. Because CPU and multiplexer work in different clock domains, we use asynchronous FIFO to transfer data.

Video controller controls other sub modules in Video TS generator to packetize ES to TS.

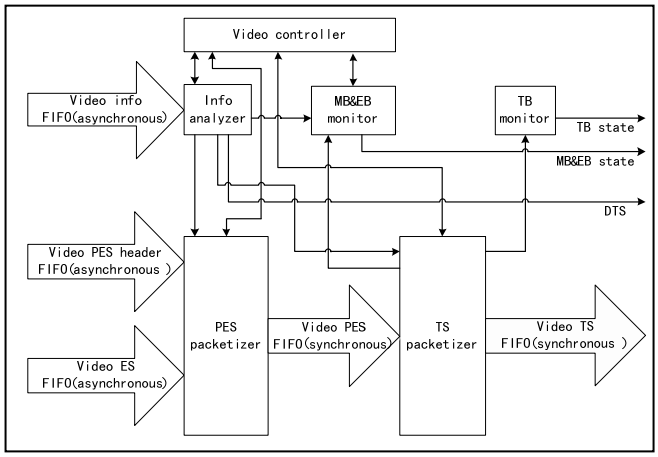


Fig. 4. Structure of Video TS generator;

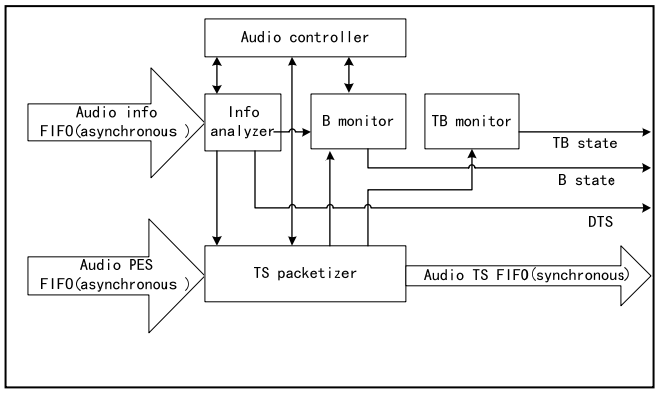


Fig. 5. Structure of Audio TS generator

There are 2 T-STD monitors in Video TS generator: MB&EB monitor and TB monitor. The MB&EB monitor imitates the behavior of video MB&EB in T-STD and the TB monitor imitates the behavior of video TB in T-STD.

The structure of MB&EB monitor is shown in figure 6. The Info analyzer provides DTS and PES length to MB&EB monitor. The req signal means that the DTS and PES length are effective. It is sent by Video controller. When Info receiver received req, it buffered DTS and PES length to DTS&len FIFO, and then returns ack to Video controller.

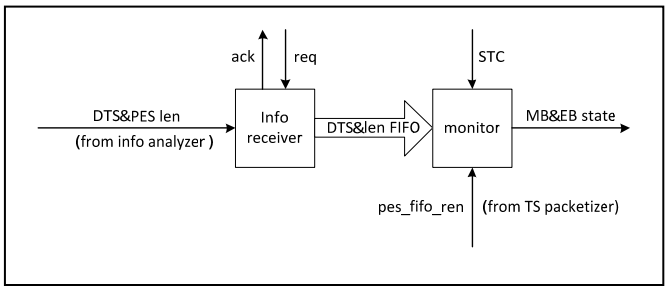


Fig. 6. Architecture of MB&EB monitor

The monitor utilizes the DTS and PES length to imitate the behavior of MB&EB in T-STD.

The pes_fifo_ren is the read enable signal of Video PES FIFO. MB&EB state equal to the data size of the MB&EB in T-STD.

There is a counter in the monitor. The value of this counter indicates the data size of the video MB&EB in T-STD. We call this value VMEB_DATA_SIZE.

The FSM of the monitor is shown in figure 7.

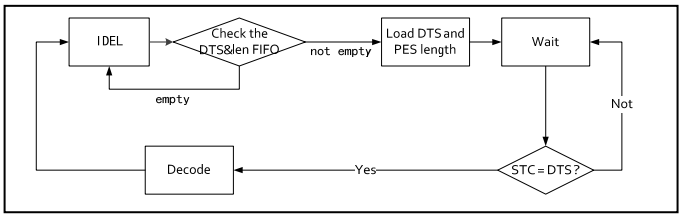


Fig. 7. FSM of monitor

The DTS, PES length and pes_fifo_ren signal determines the MEB_SIZE value by the way as follow:

```
if((pes_fifo_ren == 1)  && (state == Decode))
VMEB_DATA_SIZE=VMEB_DATA_SIZE+1
-PES_length;
else if(pes_fifo_ren == 1)
VMEB_DATA_SIZE = VMEB_DATA_SIZE + 1;
else if(state == Decode)
VMEB_DATA_SIZE=VMEB_DATA_SIZE–PES_length;
else
VMEB_DATA_SIZE = VMEB_DATA_SIZE;
```

(pes_fifo_ren == 1) means the TS packetizer is reading the Video PES FIFO.

The TB monitor use simple up-down counter to check the fullness of the corresponding TB in T-STD [3].

The structure of Audio TS generator is fundamentally same as Video TS generator. The difference is that Audio TS generator package PES into TS without any need to generate PES.

By the structure described above, Video TS generators and Audio TS generators can provide important information to scheduler for scheduling.

Video TS generator provides:

1) Data size of video MB&EB in T-STD (hereafter called VMEB_DATA_SIZE);

2) Data size of video TB in T-STD (hereafter called VTB_DATA_SIZE);

3) DTS of the frame which is being packetized now (hereafter called VIDEO_DTS);

4) Data size of the Video TS FIFO (hereafter called VFIFO_DATA_SIZE).

Similarly, Audio TS generator provides:

1) Data size of audio B in T-STD (hereafter called AB_DATA_SIZE);

2) Data size of audio TB in T-STD (hereafter called ATB_DATA_SIZE);

3) DTS of the frame which is being packetized now (hereafter called AUDIO_DTS);

4) Data size of the Audio TS FIFO (hereafter called AFIFO_DATA_SIZE).

The factors listed above are all necessary for our multiplexing strategy.

Our multiplexing strategy used by Scheduler module is shown in figure 8. Before that, we first give an illustration of the meanings of the variable in figure 8.

PCR and PSI/SI need to be output at regular time intervals. For PCR, AVS and MPEG-2 system standard specifies the maximum time interval, and DVB recommends that the time interval between two successive occurrences of PCRs for each program should not exceed 40 ms [4]. DVB also specifies minimum repetition rates for PSI/SI tables.

In our design, the 4 programs share the same PCR and we use counters based on STC to control the repetition rates for PCR and PSI/SI. In our multiplexing strategy, PCR has the highest priority, PSI/SI take the second place.

For video and audio signals, there are some requirements for output as follows:

VMEB_DATA_SIZE < T1. If this requirement is not met, it means that the MB&EB for this video signal in T-STD is almost full. As a result, we should not output TS packets of this video signal until requirements were met.

VTB_DATA_SIZE < T2. If this requirement is not met, it means that the TB for this video signal in T-STD is almost full. As a result, we should not output TS packets of this video signal until requirements were met.

VFIFO_DATA_SIZE > T3. If this requirement is not met, it means that the TS data in Video TS FIFO is not enough for a TS packet. So we cannot output TS packets of this video signal until requirements were met.

AB_DATA_SIZE < T4. If this requirement is not met, it means that the B for this audio signal in T-STD is almost full. As a result, we should not output TS packets of this audio signal until requirements were met.

ATB_DATA_SIZE < T5. If this requirement is not met, it means that the TB for this audio signal in T-STD is almost full. As a result, we should not output TS packets of this audio signal until requirements were met.

AFIFO_DATA_SIZE > T6. If this requirement is not met, it means that the TS data in Audio TS FIFO is not enough for a TS packet. So we cannot output TS packets of this audio signal until requirements were met.

T1, T2, T3, T4, T5, T6 are threshold values. In our design, these threshold values can be adjusted as needed. They should be configured before multiplexer start by firmware.

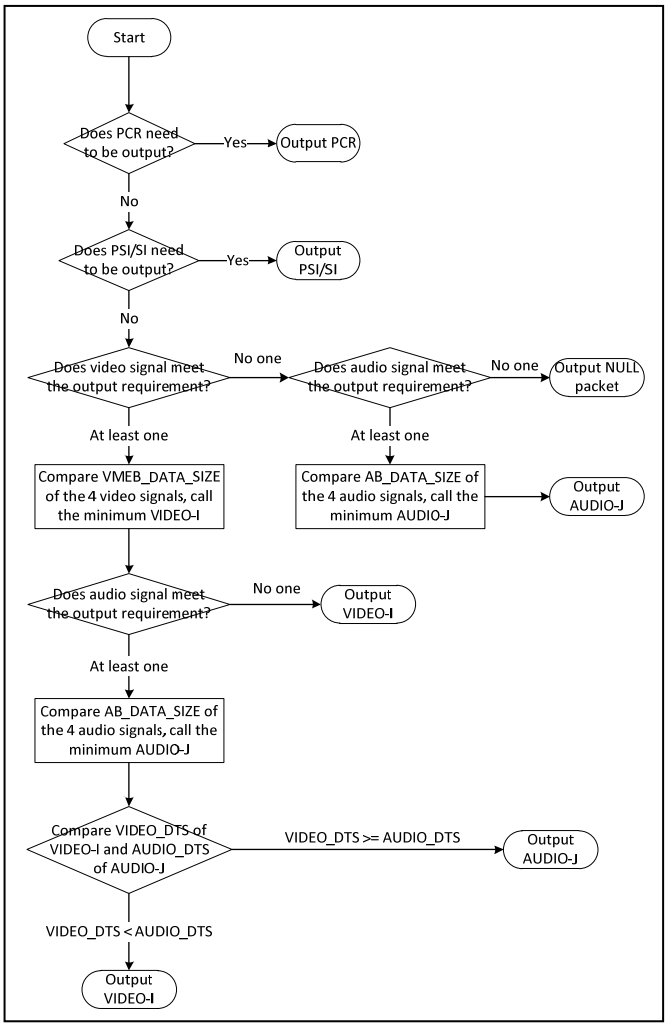


Fig. 8. Multiplexing strategy

The scheduler module use this strategy to decide which kind of TS packet (include video, audio, PCR, PSI/SI and NULL) should be output in next TS packet period. The choice will be made in current TS packet period.

IV. RTL SIMULATION

We have verified the function of our design by RTL simulation.

This multiplexer is designed as a hardware module linked on AHB bus in SOC. The input signals are controlled by firmware. So we have designed a test bench imitating the process of firmware. The corresponding data is generated in advance by software and stored in files.

The process of the test bench is shown in figure 9.

In our process, we assume that frame rates of the 4 video signals are same and frame rates of the 4 audio signals are same.

We save the result of RTL simulation as TS file. The TS file can be decoded and displayed normally.

And we have synthesized our design by ISE XST. We assume the target device is xc6vlx760-2ff1760. The device utilization summary is shown in table 1.

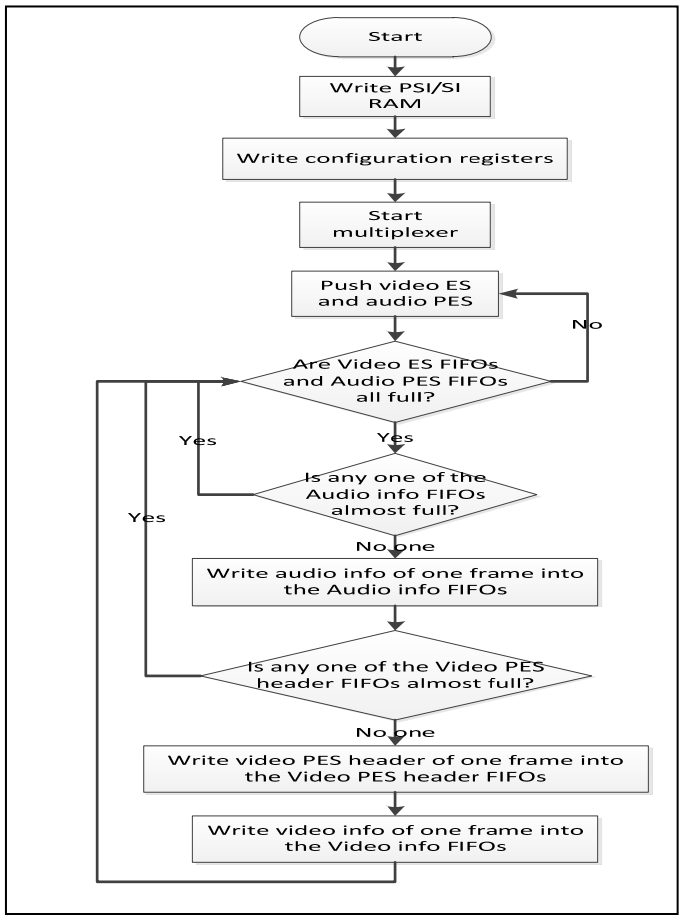


Fig. 9. Process of test bench

V. CONCLUSION

Our design can support 4 programs, multiplexing them into one single transport stream. And the architecture can be easily expanded to support more programs.

The key point of our design is using T-STD monitors to control TS output for avoiding overflow and underflow of buffers in decoder.

This design has AHB slave interface, so it can be easily transplant to other SOC systems. Meanwhile, we can also take the asynchronous FIFO as input channel directly.

And we have synthesized our design by ISE XST. We assume the target device is xc6vlx760-2ff1760. The device utilization summary is shown in table 1.

TABLE I. SYNTHESIZE RESULT

Device Utilization Summary(estimated values)			
Logic Utilization	***Used***	***Available***	***Utilization***
Number of slice registers	6265	948480	0%
Number of slice LUTs	11482	474240	0%
Number of fully used LUT-FF pairs	4524	13223	34%
Number of bonded IOBs	101	1200	8%
Number of BUFG/BUFGCTRLs	3	32	9%

REFERENCES

[1] GB/T 20090.1-2006. “Information Technology, Advanced Audio and Video Standard, Part 1: System ”

[2] Simin HE, Wei ZHAO, Wen GAO. “Even Multiplexing Problem − Abstraction, Algorithm, and Applications”.

[3] Jae-Gon Kim, Hankyu Lee, Jinwoong Kim and Joo-Hong Jeong. “Design and implementation of an MPEG-2 transport stream multiplexer for HDTV satellite broadcasting”. IEEE Transactions on Consumer Electronics, Vol. 44, No. 3, pp. 676, August 1998.

[4] ETSI TR 101 154, “Digital Video Broadcasting (DVB); Implementation guidelines for the use of MPEG-2 Systems, Video and Audio in satellite, cable and terrestrial broadcasting applications” V1.4.1, July 2000.

[5] Si J. Kim, Jong-Seog Koh. “An implementation of MPEG-2 transport stream multiplexer”. Signal Processing Systems, 1999.

[6] Hee-Beom Kang, Choon-Sik Jung, Hyoung-Gil Kim, Sang-Keun Lee, Cheul-Hee Hahm. “MPEG-2 transport stream multiplexer for recoding”. Consumer Electronics, 2005. ICCE. 2005 Digest of Technical Papers.

[7] David K. Fibush. “Timing and synchronization using MPEG-2 transport streams”. SMPTE Journal, pp. 395-400, July 1996.