

---

# SPI-4.2 인터페이스 코어의 설계

손승일\*

## A Design of SPI-4.2 Interface Core

Seung-Il Sonh\*

---

This work is supported by Hanshin University Research Grants in 2004

---

### 요 약

시스템 패킷 인터페이스 4레벨 2단계(System Packet Interface Level4 Phase 2)는 10Gbps 이더넷 응용 뿐만아니라, OC-192 대역폭의 ATM 및 POS를 통한 패킷 또는 셀 전송을 위한 물리계층과 링크 계층 소자간의 인터페이스이다.

SPI-4.2 코어는 전송 인터페이스 블록과 수신 인터페이스 블록으로 구성되어 있으며, 전이중 통신을 지원한다. 전송부는 사용자 인터페이스로부터 64비트의 데이터와 14비트의 헤더 정보를 비동기 FIFO에 쓰고, PL4 인터페이스를 통해 DDR 데이터를 전송한다. 그리고 수신부의 동작은 전송부와 역으로 동작한다. 전송부와 수신부는 캘린더 메모리를 컨피규레이션함으로서 최대 256개의 채널 지원이 가능하고, 대역폭 할당을 제어할 수 있도록 설계하였다. DIP-4 및 DIP-2 패리티 생성 및 체크를 자동적으로 수행하도록 구현하였다. 설계된 코어는 자일링스 ISE 5.1i 툴을 이용하여 VHDL언어를 사용하여 Model\_SIM 5.6a를 이용하여 시뮬레이션 하였다. 설계된 코어는 라인당 720Mbps의 데이터 율로 동작하였다. 따라서 총 11.52Gbps의 대역폭을 지원할 수 있다. SPI-4.2 인터페이스 코어는 기가비트/테라비트 라우터, 광학 크로스바 스위치 및 SONET/SDH 기반의 전송 시스템에서 라인카드 로 사용할 경우 적합할 것으로 사료된다.

### ABSTRACT

System Packet Interface Level 4 Phase 2(SPI-4.2) is an interface for packet and cell transfer between a physical layer(PHY) device and a link layer device, for aggregate bandwidths of OC-192 ATM and Packet Over Sonet/SDH(POS), as well as 10Gbps Ethernet applications. SPI-4.2 core consists of Tx and Rx modules and supports full duplex communication. Tx module of SPI-4.2 core writes 64-bit data word and 14-bit header information from the user interface into asynchronous FIFO and transmits DDR(Double Data Rate) data over PL4 interface. Rx module of SPI-4.2 core operates in vice versa. Tx and Rx modules of SPI-4.2 core are designed to support maximum 256-channel and control the bandwidth allocation by configuring the calendar memory. Automatic DIP-4 and DIP-2 parity generation and checking are implemented within the designed core. The designed core uses Xilinx ISE 5.1i tool and is described in VHDL Language and is simulated by Model\_SIM 5.6a. The designed core operates at 720Mbps data rate per line, which provides an aggregate bandwidth of 11.52Gbps. SPI-4.2 interface core is suited for line cards in gigabit/terabit routers, and optical cross-connect switches, and SONET/SDH-based transmission systems.

### 키워드

SPI-4.2, OC-192 ATM, 10Gbps Ethernet, Interface Protocol, Hardware, VHDL

1. 서론

최근 인터넷의 확산으로 더욱 더 빠른 네트워크망이 요구되고 있다. 네트워크망의 발전과정을 보면, 기존의 전화선망, 전용선망을 거쳐 이제는 빛의 속도로 데이터를 전송할 수 있는 광통신망이 요구되고 있으며, 실제 많은 네트워크망이 광통신망으로 바뀌고 있다. 네트워크망의 발전 속도를 현재 구축된 인터페이스 디바이스들은 이를 충족시키지 못하고, 네트워크망 내부에서 병목현상을 일으키고 있다. 이를 해결하기 위해 고속의 대역폭을 지원 하는 인터페이스 기술이 발표되고 있다[1]-[4]. 시스템 패킷 인터페이스 4레벨 2단계는 물리층 소자와 링크층 소자간의 10Gb/s 이상의 대역폭을 위한 인터페이스로 광 인터페이스 포럼에서 권고한 규격이다[3]. SPI-4.2 프로토콜 코어나 이를 내장한 네트워크 프로세서 제품을 살펴보면, Paxonet 커뮤니케이션사의 CC401 코어, Xilinx사의 SPI-4.2 코어, 인텔사의 IXP2800 네트워크 프로세서, Altera사의 MegaCore, Lattice 반도체사의 ORSPI4 등이 있다[5]-[10].

본 연구에서는 OC-192 ATM과 POS (Packet Over SONET/SDH), 그리고 10Gb/s 이더넷을 통한 패킷 또는 셀 전송을 위한 물리계층소자와 링크계층소자 간의 인터페이스인 시스템 패킷 인터페이스(SPI-4.2)에 대한 연구와 SPI-4.2 인터페이스 코어를 VHDL언어를 사용하여 설계하였고, Model\_Sim 5.6a를 이용하여 시뮬레이션 하였다.

II. SPI-4.2의 개요

SPI-4.2는 물리층 소자와 링크층 소자간의 데이터 전달을 위한 시스템 패킷 인터페이스이다. 그림 1은 이와 관련된 시스템 참조 모델을 보여주고 있다. 한편, SFI (SERDES Framer Interface)는 PHY 계층의 SONET/SDH 프레임과 고속 Parallel-to-Serial/SerIal-to-Parallel(SERDES) 로직과의 전기적 인터페이스를 정의한다[11].

SPI-4.2는 데이터와 제어 및 상태 신호들의 전달 방향에 따라 "Transmit"과 "Receive"로 구분된다. 그림 2는 SPI-4.2의 인터페이스 신호를 보여주고 있다.

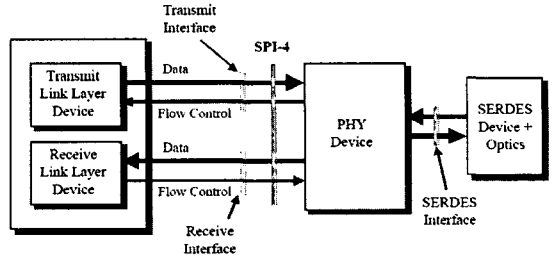


그림 1. 시스템 참조 모델  
Fig. 1 System reference model

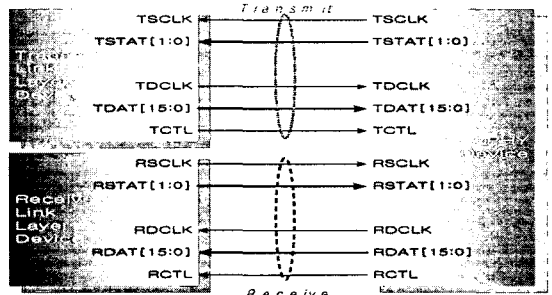


그림 2. SPI-4.2의 인터페이스 신호  
Fig. 2 Interface signals for SPI-4.2

16비트의 데이터 버스를 가지고 있으며 각 데이터 동기를 위하여 TDCLK신호를 각 모듈의 수신단에게 전송하게 된다. 데이터를 전송하기 위해서 SPI-4.2모듈은 2비트의 수신단 FIFO의 상태정보를 수신하고 수신된 상태정보를 확인하여 데이터를 전송하게 된다. 상태신호의 동기를 위하여 TSCLK 신호를 입력받게 된다[3].

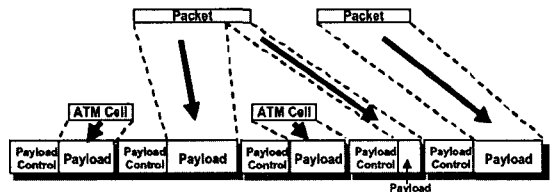


그림 3. 페이로드스트림상으로 패킷과 ATM 셀의 맵핑  
Fig. 3 Mapping of packets and ATM cells onto payload stream

데이터의 전송은 제어가 가능한 최대 길이로 연속(Burst)해서 전달될 수 있으며 EOP(End of Packet) 신호에 의해 전달이 중지된다. 매 전달과 관련된 정보들은 16비트 제어워드 형태로 전달된다. 그림 3은 데이터 흐름에 ATM 셀과 가변길이의 패킷이 맵핑되는 것을 보여주고 있다[3].

그림 4는 SPI-4.2의 Tx모듈로 입력되는 데이터와 헤더 정보를 보여주고 있다. 64비트의 데이터와 14비트의 헤더 정보를 가지고 있다. Sop는 패킷의 시작을, Eop는 패킷의 끝을, P는 현재 데이터의 에러 정보를, VByte는 데이터의 유효 데이터의 바이트 수를 나타내고, Port Address는 목적지 포트주소를 가지고 있다[3].

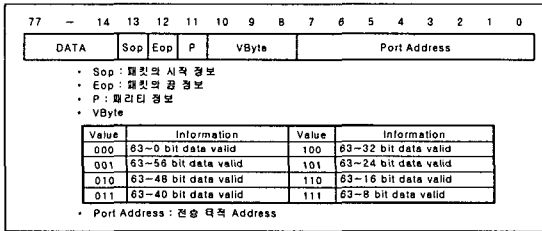


그림 4. 입력데이터 형식  
Fig. 4 Input data format

그림 5는 Tx모듈에서 전송되고, Rx모듈에서 수신하는 컨트롤 워드를 보여주고 있다. 컨트롤 워드는 16비트의 크기를 가지며, Tp는 컨트롤 워드가 Idle 컨트롤인지 Payload 컨트롤인지를, Eop는 마지막 패킷을, Sop는 패킷의 시작을, Port Address는 목적지 주소를, DIP(Diagonal Interleaved Parity)-4는 에러 정보를 나타낸다[3].

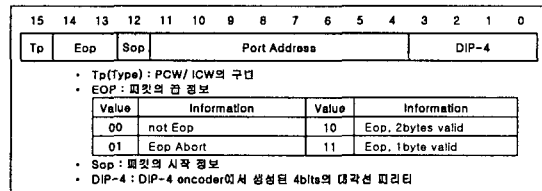


그림 5. 컨트롤 워드 형식  
Fig. 5 Control word format

SPI-4.2 인터페이스는 광통신에 적용하기 위한 인터페이스로 정확한 데이터 전송을 위해 데이터 라인에서 발생할 수 있는 비트 도달시간을 조정하여 올바른 신호를 수신할 수 있도록 데이터 전송 타이밍을 조정할 수 있는 기회를 제공한다. 이와 같은 처리를 트레이닝 시퀀스라 하며, 트레이닝 시퀀스는 별도로 설정된 주기와 연속적인 에러가 발생하였을 때 수행하게 된다[3][5].

### III. SPI-4.2 모듈의 설계

SPI-4.2 인터페이스 모듈은 512 워드 크기의

FIFO를 사용할 경우 랜덤 유니폼 트래픽에서는 97%까지, 버스트 길이 32를 갖는 버스트 트래픽에서는 94%까지의 offered load에 대해 적용이 가능하다. 그리고 원활한 패킷 처리를 위해 PCW(Payload Control Word)와 다음 PCW와는 16바이트의 이상의 간격으로 생성해야 하는데, 16바이트 미만의 작은 크기의 패킷이 입력되었을 때 많은 오버헤드로 인한 성능 저하가 발생한다[12][13]. 본 설계에서는 작은 크기의 패킷이 입력되었을 때 생성되는 ICW(Idle Control Word)패킷의 생성을 제한하여 작은 크기 패킷의 오버 헤드를 줄이는 방안을 모색하여 설계하였다.

#### 3.1 Tx 모듈의 설계

##### 3.1.1 Tx 모듈의 아키텍처

유입되어온 64비트의 데이터 패킷과 14비트의 헤더 정보를 조합하여 FIFO에 저장하고, FIFO에 데이터가 존재할 경우 Transmit FSM은 데이터 요구 신호를 발생하여 데이터를 읽어들이고 후 헤더 정보를 확인하여 Header Encoding 모듈과 Control Word Generation 모듈에서 새로운 컨트롤 워드를 생성하여 Prefetch 레지스터에 저장하고 64비트의 데이터 정보를 4개의 워드로 나누어 Data Path 모듈에 전송하여 데이터를 전송가능한 데이터로 변환하고 Prefetch 레지스터에 저장하게 된다. DIP-4 Cal 모듈에서는 전송되어질 데이터를 DIP-4 계산하여 다음에 보내질 컨트롤 워드에 추가한다. 출력 인터페이스는 DDR Interface 모듈을 사용하여 데이터를 전송하게 된다. 초기에 패킷수신단과 싱크를 맞추기 위해서 동기화기가 초기에 활성화되어 트레이닝 시퀀스를 수행하게 된다. 성공적으로 트레이닝 시퀀스를 마친 후에 수신부에서 패킷 수신이 가능함을 확인하고, 데이터를 전송하게 된다.

Calendar Memory는 FIFO 채널 정보를 가지고 있는 블록으로 TSClk와 동기하여 채널 정보를 읽어 가게 된다. Calendar Memory와 동기하여 입력되는 FIFO 상태 정보가 어느 채널의 정보인지 확인하고 Status Bank Memory 모듈에 업데이트하여 채널 정보를 관리한다. Channel FIFO Information Buffer Register 모듈은 FIFO 상태 정보가 변경된 FIFO 상태 बैं크와 업데이트된 채널 주소를 상위 계층으로 전송하는 역할을 한다. 그림 6은 Tx 블록

의 상세 데이터처리 블록도를 보여주고 있다.

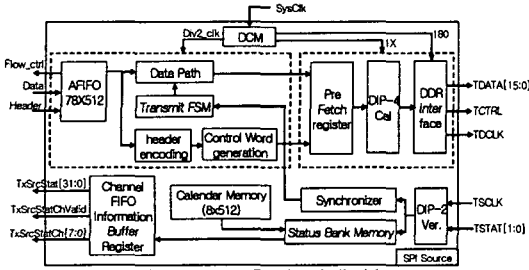


그림 6. Tx 모듈의 상세 블록도  
Fig. 6 Block diagram for Tx module

### 3.1.2 Tx 모듈의 데이터 패스

고속으로 인터페이스 되며, 인터페이스로서 병목현상을 방지하기 위하여 데이터 패스를 파이프라인 방식으로 구성하였다. 총 8단의 파이프라인을 구성하여 FIFO에서 데이터를 읽어들이 후 다음상태를 확인하여 연속적인 데이터 전송을 실행한다. 각 단계를 지나면서 데이터 패킷의 컨트롤 정보를 생성한다. 패킷의 전송이 요구되는 경우에는 PCW를 생성하며, 패킷이 없을 때에는 ICW를 생성하게 된다. 추가적인 외부의 ICW/ 트레이닝 워드 생성을 요구하는 신호를 받았을 때는 데이터 처리를 중지시키는 Freeze신호를 발생시킨다. 이 외에도 PCW와 PCW와의 간격의 16바이트이상이 되어야 데이터 처리가 가능하므로, 8바이트 패킷인 SEOP 패킷의 처리에 있어서 모듈은 Freeze 신호를 발생하게 된다. 그리고, 데이터 패킷의 정보를 가지고 있는 PCW와 ICW의 생성으로 오버헤드가 발생하게 되는데 이러한 오버헤드의 처리를 위하여 Freeze신호를 발생시킨다. 이와 같이 3개의 Freeze 신호를 통하여 파이프라인을 중재하여 데이터 패스를 구현하였다. 그림 7은 파이프라인의 방식의 데이터 패스를 보여주고 있다.

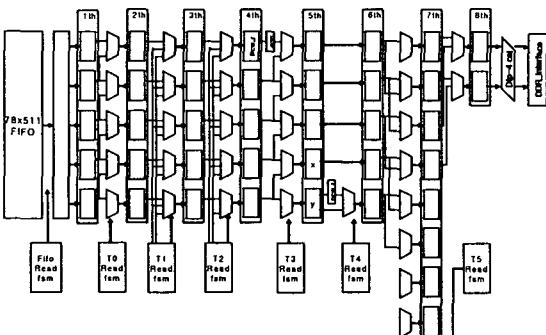


그림 7. 파이프 라인방식의 데이터 패스  
Fig. 7 Data path using pipeline method

### 3.1.3 Tx 모듈의 외부 인터페이스

Tx 모듈의 외부 인터페이스는 DDR(Double Data Rate) 인터페이스를 사용하여 구현 하였다. 16비트의 데이터와 동기를 위한 클럭, 전송 패킷의 정보를 전송하게 된다. 그림 8은 데이터 패스부의 일곱 번째 레지스터의 상위 4개의 레지스터를 이용하여 데이터를 전송하게 된다. 4개의 워드 레지스터를 2배 클럭 속도의 DDR로 1번에 출력하게 된다.

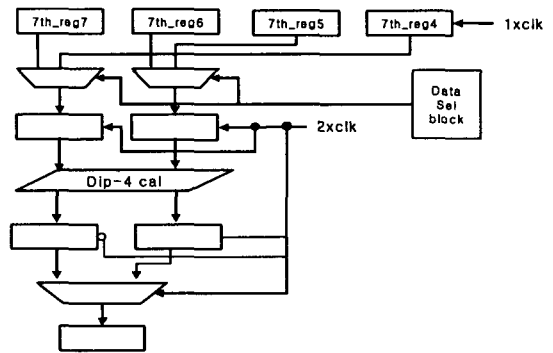


그림 8. SPI-4 Tx 모듈의 외부 인터페이스  
Fig. 8 External interface for SPI-4 Tx module

### 3.1.4 Tx 모듈의 싱크 설정

연속적인 Dip2\_Error가 발생하지 않을 경우 모듈은 항상 Sync를 유지한다. 그리고 미리 설정되어 있는 TxNumDip2Errors의 값 만큼의 연속적인 Dip2\_Error가 발생하면 모듈은 Sync를 잃게 되고 싱크를 맞추기위한 작업을 수행한다. 그림 9는 싱크 생성 상태머신을 보여주고 있다.

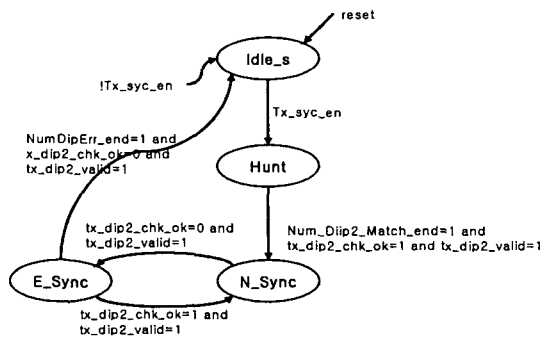


그림 9. Tx 모듈의 싱크 생성 상태머신  
Fig. 9 Synchronization FSM for Tx module

싱크 상태머신의 N\_Sync 상태와 E\_Sync 상태에서 Tx 모듈의 싱크가 활성화 된다. Tx 모듈의 데이터 전송부와 상태 수신부는 서로 동기 클럭이 다르기 때문에 비동기 FIFO를 이용하여 싱크신호를 Tx 모듈의 데이터 전송부에서 사용되는 동기 클럭으로 맞추어 활성화한다.

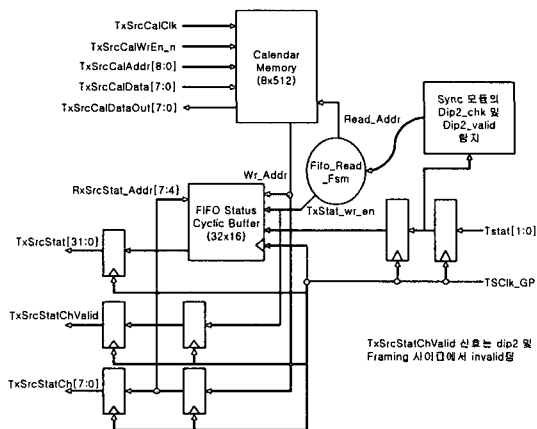


그림 10. 채널 상태정보의 수신 블록도  
Fig. 10 Receive block diagram for channel status information

### 3.1.5 Tx 모듈의 채널상태 정보의 수신

FIFO 상태는 수신단으로부터 2비트를 입력받게 된다. 입력받은 상태비트를 통해서 Sync를 활성화하고, DIP-2 에러 정보를 이용하여 특정 회수만큼의 에러 검출시 Sync를 비활성화한다. 캘린더 정보를 확인하여 현재 입력되어온 FIFO 정보가 어떤 포트의 정보인지를 확인하여 흐름제어를 한다. 입력되어온 FIFO 정보는 뱅크 구조의 메모리 블록에 항상 업데이트 하게 된다. 그림 10은 FIFO 상태 정보 수신후 업데이트 하고, 상태 정보를 출력하는 블록도를 보여주고 있다.

### 3.1.6 Tx 모듈의 에러 검출 및 생성

그림 11은 DIP4의 계산 블록도이다. Tx모듈의 데이터 전송부에서 계산하게 되던 전송되는 데이터를 DIP-4를 계산하여 다음에 전송될 컨트롤 정보에 삽입하여 전송하게 된다. DDR 인터페이스에 의해 2개의 데이터가 한 클럭에 전송되게 되므로 2개의 레지스터를 두어 DIP-4값을 계산하여 관리한다.

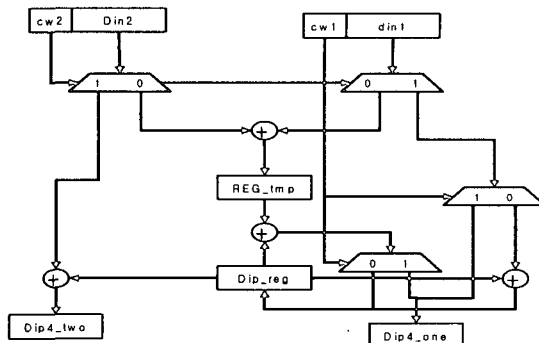


그림 11. DIP-4 생성 블록도  
Fig. 11 Block diagram for DIP-4 generation

Tx 모듈은 수신단의 채널 정보값 2비트를 입력받게 된다. 2비트의 FIFO 상태정보는 설정되어있는 Calendar\_M, Calendar\_Len을 곱한만큼 수신하게 되고, 그때까지의 수신된 상태정보를 DIP-2 연산후 상태정보에 이어서 수신하게 된다. DIP-2정보를 수신한 후 Frame신호 "11"을 수신하게 된다. 이 과정을 계속 반복하며 수신부의 FIFO정보를 업데이트 한다. 그림 12는 전송부에서 FIFO의 상태 정보와 DIP-2를 수신한 후 DIP-2의 에러 유무를 판별하는 블록도이다.

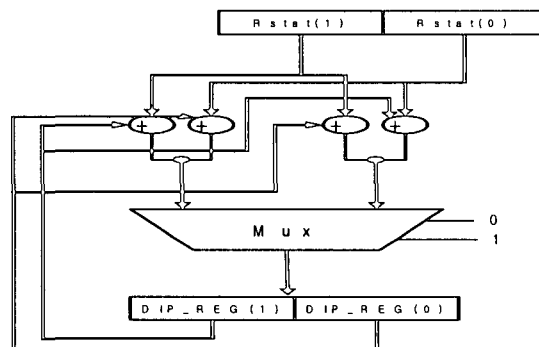


그림 12. DIP-2 에러 검출 블록도  
Fig. 12 DIP-2 error detection block diagram

## 3.2 SPI-4.2 Rx 모듈의 설계

### 3.2.1 Rx 모듈의 아키텍처

유입되어온 16비트의 PDW(Payload Data Word), PCW, ICW등의 데이터 패킷과 각각의 종류를 확인할 수 있는 컨트롤 비트와 동기클럭을 입력받는다. DDR Input Interface 모듈에서 RData와 Rdata의 정보인 RCtrl을 싱글 에지로 동작하는 레지스터에 저장하여 트레이닝 워드일 경우 TP Verification & Synchronizer 모듈에서 받아 동기

를 확인하고, 그 외의 정보는 DIP-4 Verification 모듈에서 DIP-4 에러를 검출하게 된다. Data Align 모듈은 입력 스트림에서 PCW를 검출하여 Header Decoding 모듈로 전송하고 PDW를 재정렬하여 Data Path 모듈에 전송하게 된다. Data Path 모듈에서 재조립된 64비트의 데이터 정보와 Header Decoding 모듈에서 생성된 14비트의 헤더 정보를 Prefetch Register에 저장하여 FIFO Write Control 모듈의 제어신호에 의해 FIFO에 저장하게 된다.

Calendar Memory는 Rx 모듈이 인에이블 되기 전에 설정되게 되고 Rx 모듈의 인에이블은 Calendar Memory의 값이 모두 입력되었을 때 가능하다. 하위 계층에서 입력받은 뱅크 구조의 32비트의 FIFO 상태 정보를 입력받아 Status Bank Memory 블록에 업데이트 하게 된다. Transmit Status Select 모듈은 Calendar Memory의 채널 주기 정보를 확인하여 Frame 정보, FIFO 상태 정보, DIP-2 계산값들 중에 하나를 선택하여 하위 계층으로 전송하는 역할을 수행한다.

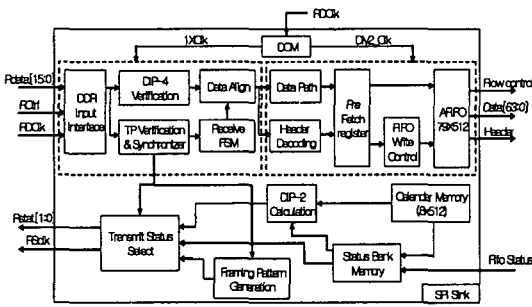


그림 13. Rx 모듈의 상세 블록도  
Fig. 13 Block diagram of Rx module

3.2.2 Rx 모듈의 데이터 패스

DDR 인터페이스를 통해 입력되는 패킷을 재조립하여 상위계층으로 전송하기 위해 파이프라인 방식의 데이터 패스를 구현하였다. 유입되는 PCW와 ICW 패킷을 이용하여 14비트의 헤더 정보를 생성하고, PDW를 이용하여 64비트의 데이터 정보를 생성한다. 데이터 생성에는 4개의 PDW가 필요하게 된다. 데이터의 처리에 있어서 컨트롤 워드는 따로 분류하여 처리하게 되는데 이때 발생하는 데이터 패킷의 순서가 변화하게 된다. 이것을 막기 위해 데이터를 정렬하는 모듈이 삽입되어 두개의 데이터 쌍으로 맞추게 된다. 그림14는 Rx 모듈의 파이프라인방식의 데이터 패스를 보여주고 있다.

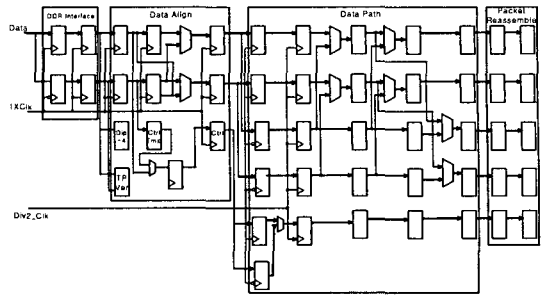


그림 14. Rx 모듈의 데이터 패스  
Fig. 14 Data path of Rx module

DDR 인터페이스를 통한 데이터에 대해 PCW의 위치를 확인하여 PCW를 컨트롤 레지스터로 전송해 주고 PDW 데이터를 재정렬 해준다. 그림 15는 PCW의 위치에 따른 PDW의 재정렬을 보여주고 있다.

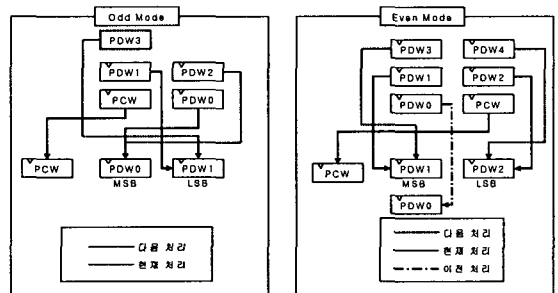


그림 15. PCW의 위치에 따른 PDW 재정렬  
Fig. 15 PDW realignment according to PCW position

Status Bank Memory는 2x16크기의 메모리로 16개를 두어, 256개의 채널의 정보를 상시 업데이트해주며, 사용하지 않는 포트에 대해서는 RxSnk Mask를 이용하여 마스크 처리하고 정보를 업데이트 한다. 그림 16은 Rx FIFO status 전송부의 블록도를 보여주고 있다.

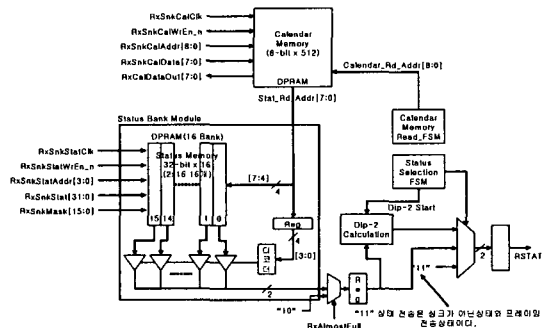


그림 16. Rx 모듈의 채널 정보 전송 블록도  
Fig. 16 Block diagram of Channel status information in Rx module

#### IV. 설계 모듈의 시뮬레이션

##### 4.1 Tx 모듈의 파형분석

그림 17은 Tx 모듈의 파형을 보여주고 있다. 현재 파형은 트레이닝 시퀀스를 전송하고, Rx모듈로부터 Sync가 활성화되어 NumDip2Matches에 설정된 횟수만큼의 DIP-2 매치를 확인하여 TxSync가 활성화된다. TxSync신호를 확인하고 SrcFFAddr, SrcFFData, SrcFFMod, SrcFFSop, SrcFFEop, SrcFFErr, 정보가 SrcFFWrEn\_n 신호가 활성화되었을 때 데이터를 입력받게된다. 입력받은 데이터를 FIFO에 저장하여 FIFO가 비어있지 않을때 데이터를 읽어들이어 SrcFFData를 4개의 워드로 나누고, 부가정보를 이용하여 PCW정보를 생성하여 파이프라인방식으로 처리하면서 최종적으로 TData신호와 TCtrl 신호에 데이터를 전송하게되고 FIFO에 데이터가 없을 경우 ICW를 생성하여 전송하게 된다.

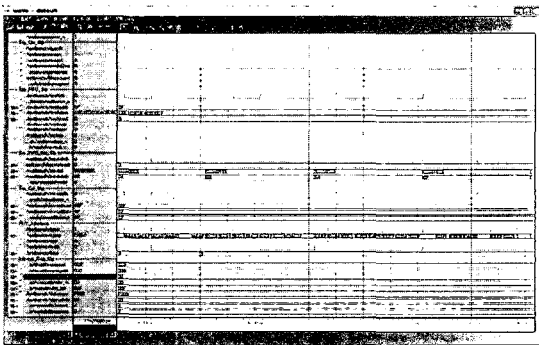


그림 17. Tx모듈의 파형  
Fig. 17 Waveform of Tx module

##### 4.2 Rx 모듈의 파형분석

그림 18는 Rx 모듈의 파형을 보여주고 있다. 현재 파형은 하위 계층으로부터 트레이닝 시퀀스를 입력받아 NumTrainSequences 만큼의 트레이닝 시퀀스 매치를 확인하여 RxSync 신호를 인에이블하게 된다. 그리고 상대 전송부에 프레임 패턴전송을 중지하고 상위계층으로부터 입력받은 FIFO 상태정보를 전송하게 된다. RxSync 신호가 활성화된 후에도 연속적으로 트레이닝 데이터가 들어올 경우 SnkTrainValid 신호가 활성화되어 유효하지 않은 데이터 입력됨을 알리게 된다. SnkTrainValid 신호가 비활성 되었을때 RData와 RCtrl 신호에 데이터 유입되는 것으로 판별하여 데이터를 재조립하게 된다. 재조립된 데이터는 FIFO에 저장되어 있다. SnkFFEmpty\_n 신호가 활성화되면 상위 계층에서는

재조립이 완료된 데이터가 존재함을 확인하여 SnkFFRdEn\_n 신호를 활성화 하여 SnkFFAddr, SnkFFData, SnkFFMod, SnkFFSop, SnkFFEop, SnkFFEr에 데이터가 전송되게 된다. 현재 전송되고 있는 데이터가 유효한 데이터임을 알리는 SnkFFValid가 활성화게 된다. 그리고 패킷 재조립 과정에서 발생된 에러 정보인 SnkFFPayloadErr, SnkFFDIP4Err, SnkBusErr정보를 전송하여 현재 전송되는 패킷의 에러 유무를 전송한다. 상위 계층으로부터 SnkStat 신호를 통해 유입된 बैं크구조의 상위 FIFO 상태정보를 입력받아 상태 बैं크 메모리를 업데이트하고 캘런더 메모리와 같은 동기로 기 설정된 Calendar\_Len을 Calendar\_M 번의 주기만큼 RStat 신호를 통해서 전송하게 된다.

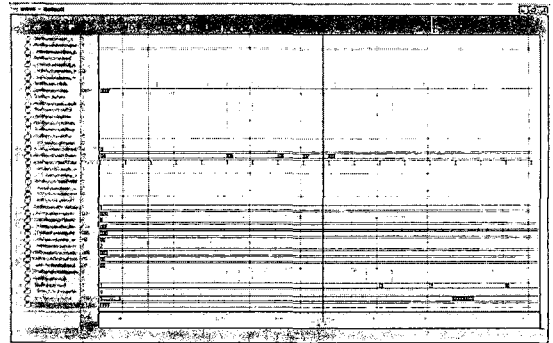


그림 18. Rx 모듈의 파형  
Fig. 18 Waveform of Rx module

#### V. 설계 결과

설계된 SPI-4.2모듈은 사용자 인터페이스에 64비트의 데이터와 PHY 계층과의 인터페이스에 16비트의 데이터 전송라인을 요구하며, OC-192 (10Gb/s)의 대역폭을 지원하기위해 라인당 최소한 622Mbps를 지원해야한다. Xilinx PL4 v5.2와 Paxonet CC401 모듈은 라인당 700Mbps를 지원하여 11.2Gb/s전송율을 지원한다. 설계한 모듈은 라인당 720Mbps를 지원하여 총 대역폭이 11.52Gb/s의 전송율을 나타내어 더욱 안정적으로 패킷을 인터페이스 할 수 있다. 설계 면적 또한 PL4 v5.2보다 적은면적으로 설계 되었다. 표 1은 설계모듈과 기존에 설계된 모듈에 대한 성능 비교를 표로 보여주고 있다.

표1. SPI-4 모듈에 대한 성능 비교  
Table 1. Performance comparison for SPI-4 modules

비교모델 비교 값	표준안 요구 사항	Xilinx PL4 V5.2	Paxonet CC401	Our design
User Interface	.	175Mhz	175Mhz	180Mhz
PL4 Interface	622Mbps	700Mbps	700Mbps	720Mbps
Slices	n/a	3150	2735	2720

### VI. 결 론

SPI-4.2는 10Gb/s 이상의 대역폭을 위한 패킷 인터페이스를 지원한다. 정의된 인터페이스는 라인당 최소 622MHZ 이상의 전송율을 가지는 16비트의 데이터 패스로 구성되며, OC-192 SONET/SDH를 통한 셀 또는 패킷의 전송, 그리고 10Gb/s 이더넷을 지원한다.

본 논문에서는 시스템 패킷 인터페이스 4레벨 2단계(SPI-4.2)에 대한 연구와 성능평가 결과를 기반으로 하여 짧은 길이의 패킷이 입력되었을 때, 적은 오버헤드를 발생할 수 있게 설계하였다. 그리고 가변길이 패킷에 대한 처리도 가능하다. SPI-4모듈의 설계는 Xilinx ISE 5.1i 툴을 이용하여 VHDL 언어로 설계하였고, 설계모듈의 검증은 ModelSim 5.6a를 이용하여 시뮬레이션 하였다. 짧은 길이 패킷에 대해 적은 수의 오버헤드를 확인할 수 있었다. 추가적으로 10Gb/s의 지원을 위해 설계모듈은 라인당 622Mbps를 지원해야 한다. 설계 모듈은 라인당 720Mbps를 지원하여 관련 응용분야에서 사용이 가능할 것으로 사료된다.

### 참고문헌

[1] "The Voice of the Future : Next Generation Networks", ATM Form, July. 2002  
 [2] "Common Switch Interface Specification-L1", ATM Form, August 2001  
 [3] "System Packet Interface Level 4(SPI-4) Phase 2 : OC-192 System Interface for

Physical and Link Layer Devices.", OIF, January. 2001

[4] "Streaming Interface(NPSI) Implementation Agreement," NPF, October. 2002  
 [5] "SPI-4.2(PL4) Core V5.2" Xilinx, January. 2003  
 [6] "CoreEI SPI-4 Phase 2 Interface Core(CC401)", Paxonet Communications, Inc., May 2003  
 [7] "POS-PHY Level 4 MagaCore Optimization for the Intel IXP2800 Network Processor", White Paper, April 2003  
 [8] "SPI-4.2 Interoperability with the Intel IXP1110 in Stratix GX devices", Altera Corp., May 2003  
 [9] "SPI-4.2 Interoperability with PMC-Sierra XENON Family in Stratix GX devices", Altera Corp., May 2003  
 [10] "ORCA ORSPI4 - Dual SPI4 Interface and High-Speed SERDES FPSC", Lattice Semiconductor Corp., Jan. 2004  
 [11] "OIF Electrical Interfaces", OIF, 2001  
 [12] 박노식, 손승일, 최익성, 이범철, "System Packet Interface 모듈의 성능평가", 한국인터넷정보학회 추계 학술발표 논문집, pp247-250, November 2003.  
 [13] 박노식, 손승일, 최익성, 이범철 "SPI-4.2 프로토콜을 사용한 PHY-LINK 계층간의 데이터 전송 성능평가", 해양정보통신학회 논문집 8권 3호

### 저자소개

#### 손승일(Seung-Il Sonh)



1989년 연세대학교 전자공학과 (학사)  
 1991년 연세대학교 대학원 전자공학과(석사)  
 1998년 연세대학교 대학원 전자공학과(박사)

1998~2002년 호남대학교 컴퓨터공학과 조교수  
 2002년~현재 한신대학교 정보통신학과 부교수  
 ※관심분야 : ATM 통신 및 보안, ASIC 설계