論文 95-4-4-10

SUIO 및 MUIO를 이용한 Test Sequence 생성 : B-ISDN의 UNI SSCS계층에 적용

正會員 鄭 允 姬*, 李 庚 熙*, 洪 范 基*

Test Sequence Generation Using SUIO/MUIO:Applied to SSCS at UNI in B-ISDN

Yoon Hee Jung*, Kyung Hee Lee*, Beom Kee Hong* Regular Members

要 約

통신망에서 프로토콜 구현물의 정확성은 중요한 역할을 하며 적합성 시험은 그 정확성을 보장하는 중요한 접근방법이다. 일반적으로 적합성 시험은 규격으로부터 얻은 test sequence를 implementation under test(IUT)에 적용하여 그 output을 기대하는 것과 비교함으로써 이루어진다.

본 논문에서는 프로토콜 적합성 시험을 위한 test sequence를 생성하는 방법에 대하여 기술한다. Graph theory에서 잘 알려진 Chinese postman problem에 기반을 두고 single unique input/output(SUIO)와 multiple minimum-length UIO(MUIO)을 이용하여 test sequence를 생성한다. 임의의 state의 UIO는 그 UIO가 적용(apply)되기 전에 그 state에 있었는지를 검증(verify)하는 일련의 input/output sequence이다. 이 방법을 Broadband Integrated Service Digital Network(B-ISDN) User Network Interface(UNI)의 Service Specific Convergence Sublayer(SSCS)계층에 적용하여 test sequence를 구하고 field trial에서 발생한 문제들에 대하여 언급한다. 또한 fault coverage를 구하는 방법과 위에서 구한 test sequence에 적용한 결과를 기술하였다.

ABSTRACT

The correctness of a protocol implementation is vital to the operation of communication networks. Conformance testing is an important approach to granting correctness. Conformance testing is generally done by applying a test sequence, , to an implementation under tester(IUT) which is constructed from the specification of the standard. The observed outputs of the IUT is compared with the expected ones to detect errors.

* 한국전자통신연구소
Electronics and Telecommunication Research Institute
論文番號 : 94235-0901
接受日字 : 1994年 9月 1日

In this paper, a technique generating a test sequence for protocol conformance testing is presented. The approach is based on a well-known concept in graph theory as the Chinese postman problem and SUIO and MUIO sequences. A test sequence for Service Specific Convergence Sublayer(SSCS) at User Network Interface(UNI) in Broadband Integrated Service Digital Network(B-ISDN) is generated. The experience with generating test sequences are described. And the fault coverage of the resulted test sequences are compared by means of simulations.

Ⅰ. 서 론

프로토콜 구현물들은 호환성 및 연동의 가능성을 증가시키기 위하여 정의된 규격과의 적합성 여부를 시험하여야 한다. 일반적으로 프로토콜 적합성 시험은 일련의 input을 적용하여 이로부터 얻은 output들이 규격에서 기대하는(expected) output과 일치하는지를 확인하는 절차라 할 수 있다. Input들로 구성된 test sequence는 finite state machine(FSM)의 각 transition을 적어도 한번 수행하는 것을 보장하는 approximation algorithm에 의해 생성된다.

프로토콜 적합성 시험을 위하여 test sequence를 생성하는 여러가지 방법들이 제안되어 왔다. 프로토콜 구현물이 기대하는 state에 있는지를 검증하기 위하여 unique input/output(UIO or SUIO) sequence와 Rural Chinese Postman algorithm을 함께 적용한 방법은 간결한 test sequence를 제공한다[1]. [2]에서는 FSM의 각 state에서 SUIO대신 multiple minimum-length UIO sequence(MUIO))를 제안하였는데 이를 이용하면 4%~37%정도의 test sequence의 길이를 줄일 수 있다.

본 고에서는 test case의 하나로, B-ISDN UNI의 SSCS계층 프로토콜에 SUIO와 MUIO를 각각 적용하여 test sequence를 생성하였으며 이때 생기는 문제점을 기술하였다. 또한 simulation을 통해 이들 test sequence의 fault coverage를 비교하였다. 본 고의 구성으로써 2장에서는 B-ISDN UNI의 SSCS계층에 대한 test sequence생성 방법 및 실제 UNI SSCS계층에 적용 예를 보이고 4장에서는 fault coverage를 구하는 방법과 3장에서 적용한 예의 fault coverage 결과를 simulation에 의해 비교한다.

Ⅱ. UNI SSCS계층 프로토콜 및 상태 천이표

ITU-T는 OSI reference model을 참조하여 B-ISDN에서 사용되는 프로토콜을 권고하고 있는데 UNI signalling을 위한 프로토콜은 physical layer, ATM layer, SAAL 및 network layer인 Q.2931로 구성된다. SSCS(Service Specific Convergence Sublayer)계층은 SAAL을 구성하는 한 부분으로서 SSCOP(Service Specific Connection-Oriented Protocol)부계층과 SSCF(Service Specific Coordination Function)부계층으로 세분된다. SSCOP부계층은 ATM virtual channel을 사용하여 network layer사이의 정보를 전달하는 기능을 수행하며 특히 재전송에 의한 error recovery기능, 흐름제어와 keepalive의 기능을 수행한다. SSCF부계층은 SSCOP부계층과 SSCOP부계층사용자 사이의 프리미티브를 매핑하는 기능을 수행한다.

SSCS계층의 상태 천이표를 ITU-T의 Q.2100, Q.2110와 Q.2130의 규격으로부터 유도하여 표 1에 나타내었다. 첫번째 열은 event이며, 첫번째 행은 compound state P/Q로써, P는 SSCF 부계층 그리고 Q는 SSCOP부계층의 상태를 각각 나타낸다. 표 1에서의 각 entry는 remote test를 가정하였을때, 해당 compound state에서 해당되는 event를 수신하였을때 취해지는 action(output)을 나타내며 next state는 entry의 오른쪽에 나타내었다. 표 1의 entry중 illegal로 표시된 것은 대부분 부적당한 메세지를 나타내며, 여기서는 그 event가 상태변이를 일으키지 않고 output을 생성하지도 않으며 무시되는 것으로 가정하였다. SSCS계층 프로토콜은 상태 4/10에서 input으로 SD, POLL, STAT 또는 USTAT를 받은 경우 내부변수의 값에 따라 output이 다르므로 extended finite state machine(EFSM)으로 다루어야 한다.

표 1. UNI의 SSCS 계층의 상태 천이표
Table 1. Transition table of UNI SSCS

Compound State Event	1/1(V1)	2/2(V2)	4/1(V5)	3/4(V3)	2/5(V5)
AAL-EST.req	BGN 2/2	Illegal	RS 2/5	BGNN 2/2	Illegal
AAL-REL.req	AAL-REL.con 1/1	END 3/4	END 3/4	Illegal	END 3/4
AAL-DATA.req	1/1	Illegal	SD 4/10	Illegal	Illegal
AAL-UD.req	UD 1/1	UD 2/2	UD 4/10	UD 3/4	UD 2/5
BGNo	BGREJ 1/1	2/2	BGAK 4/10	BGAK,END 4/10	BGAK,RS 2/5
BGNx	BGAK 3/4	BGAK 4/10	BGAK 3/4	BGAK 3/4	BGAK 3/4
BGAK	END 1/1	4/10	4/10	4/10	2/5
BGREJ	1/1	1/1	4/10	1/1	1/1
END	ENDAK 1/1	2/2	ENDAK1/1	ENDAK1/1	ENDAK1/1
ENDAK	1/1	2/2	1/1	1/1	1/1
RSo			RSAK 4/10 주2		2/5
RSx	END 1/1	2/2	RSAK 4/10 주2	3/4	RSAK 4/10
RSAK	END 1/1	2/2	4/10	3/4	4/10
ER	END 1/1	2/2	ERAK 4/10	3/4	2/5
ERAK	END 1/1	2/2	ERAK 4/10	3/4	2/5
SD	END 1/1	2/2	USTAT,ER or- 4/10 주1	3/4	2/5
POLL	END 1/1	2/2	STAT,ER or SD 4/10 주1	3/4	2/5
STAT	END 1/1	2/2	ER, SD or- 4/10 주1	3/4	2/5
USTAT	END 1/1	2/2	SD or ER 4/10 주1	3/4	2/5
UD	UD 1/1	UD 2/2	UD 4/10	UD 3/4	UD 2/5

주1 : Test를 받는 system이 ER을 송신한 후에는 일시적으로 4/9의 상태를 거치며 ERAK를 수신하여야 4/10의 상태로 간다.
주2 : Test를 받는 system이 RS을 송신한 후에는 일시적으로 4/9의 상태를 거치며 RSAK를 수신하여야 4/10의 상태로 간다.
O : 재전송된 것, X : 처음으로 전송된 것

SSCF 1 : AAL CONNECTION RELEASED
2 : AWAITING ESTABLISH
3 : AWAITING RELEASE
4 : AAL CONNECTION ESTABLISHED

SSCOP 1 : Idle
2 : Outgoing Connection Pending
4 : Outgoing Disconnection Pending
5 : Outgoing Resynchronization Pending
10 : Data Transfer Ready

그러나 여기서는 변수를 고려하지 않고 단지, 여러 가능한 output중 하나를 받아도 규격을 만족하는 것으로 가정하여 SSCS계층 프로토콜을 FSM으로 처리하였다. 이 경우 변수에 대한 적합성 시험을 할 수 없으므로 이를 해결하여야 한다. 재전송되는 BGN 또는 RS에 대한 처리는 tester가 BGN 또는 RS를 처음으로 송신한 후 그 state에서 적절한 output을 받아도 이를 받지 않은 것으로 가정하여 다시 동일한 BGN 또는 RS를 보내어 처리한다. 그리고 표 1의 주 1과 주 2에 나타냈듯이 test를 받는 system이 일시적인 상태를 가지는데 여기서는 이를 고려하지 않았다. 왜냐하면, 이 문제는 SSCOP와 SSCF를 함께 시험하기 때문에 생기는 것으로써 SSCOP와 SSCF를 하나의 block box로 보면 보이지 않는 state이므로 두 계층을 함께 시험한다면 무시할 수 있기 때문이다.

Ⅲ. Test Sequence 생성

일반적으로 프로토콜 적합성 시험은 가능한 상태 천이를 모두 수용하여야 한다. 상태천이는 directed graph G=(V, E)에 의해 FSM으로 표현될 수 있다. 여기서 V는 유한개이고 non-empty인 vertex의 집합이고 E는 edge의 집합을 나타내며 V_i에서 V_j로의 edge는 $(V_I, V_J; a_k/o_k)$로 표현될 수 있으며 a_k와 o_k는 각각 input과 output operation을 나타낸다. Input은 lower와 upper layer의 interface로 부터 받은 메세지이고 output은 외부에서 관측 가능한 event이며 null일 수도 있다.

FSM으로 표현된 프로토콜의 적합성 시험은 vertex V_i에서 V_j로의 edge를 테스트한다고 하면, 먼저 FSM의 상태를 vertex V_i에 해당하는 state에 위치 시켜, input operation을 적용하여 output이 규격과 일치하는지를 관찰하여 일치할 경우 FSM이 새로운 state V_j에 위치하는지를 확인함으로써 이루어진다. 따라서 프로토콜 적합성 시험을 위해 다음의 3단계를 수행한다.

1. Test edge로 구성된 test graph G'을 구한다. 여기서 test edge는 G의 edge의 input에 그 edge의 tail을 확인하는 input을 결합하여 생성된 새로운 edge를 말하며 G의 각 edge에 하나씩 매핑된다. Test edge의 집합을 E_c라 하자. (그러면 프로토콜 적합성 시험을 위한 test sequence는 모든 test edge를 적어도 한번 지나는 Chinese Postman Problem을 푸는 것과 같다.
2. G'을 G의 edge를 사용하여 symmetric하게 만든다. 이 때 만들어진 새로운 graph를 symmetric test graph G*라 하자. G*를 만들기 위해 다음의 equation을 푼다.

$$\begin{aligned} min \quad & \textstyle\sum_{all\ edge}(J_{ij} + I_{ij}) \\ s.t. \quad & \textstyle\sum_{all\ incoming\ edges\ of\ V_i}(J_{ij} + I_{ij}) \\ & J_{ij} = 1 \qquad for\ all\ i, j \\ & \textstyle -\sum_{all\ outgoing\ edges\ of\ V_i}(J_{ij} + I_{ij}) = 0 \end{aligned}$$

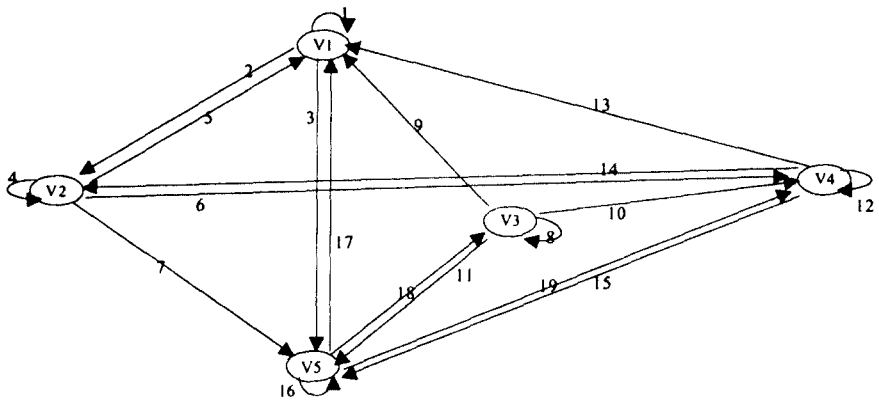
여기서 J_{ij}는 V_i에서 V_J의 test edge $\in E_c$, I_{ij}는 V_i에서 V_j의 edge $\in E$ 모든 i와 j에 대하여 $J_{ij} = 1$의 값을 갖는 것은 모든 test edge를 한번 지나는 것을 나타내며 한번만 지나가게 한 이유는 E_c의 edge보다 cost가 비싸기 때문이다.

3. 2단계에서 구한 graph로 부터 Euler Tour를 찾는다. (Stronly connected이고 symmetric한 graph에서 Chinese Postman Problem은 Euler Tour를 찾는 것과 같다[5].)

1. SUIO를 이용한 Test Sequence생성

[1]은 앞에서 언급한 3단계중, 1단계에서 tail을 확인 하기 위하여 각 state에서 하나의 UIO을 이용하는 것을 제안하였다. UIO를 이용하여 test sequence를 구하는 방법은 FSM이 minimal, strongly connected 그리고 fully specified의 가정을 만족하여야 한다[5]. Graph G의 state수가 G와 equivalent한 어느 graph의 state의 수보다 작거나 같으면 minimal이라 한다. Stornly connected의 성질을 가진 FSM은 두 개의 서로 다른 임의의 edge pair V_i와 V_j에 대하여 V_i에서 V_j로의 path가 존재한다. 각 state에서 각 input에 대하여 transition이 존재하면 fully specified라 한다. 규격에 output이 명확히 정의되어 있지 않은 input에 대하여 구현물이 error를 구동하거나 이를 무시한다면 그 FSM은 여전히 fully specified하다 할 수 있다[1].

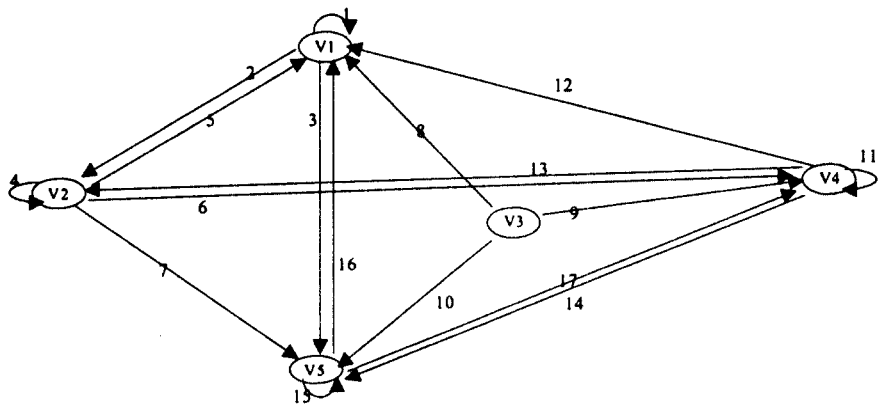
표 1의 상태 천이표를 바탕으로 UNI의 SSCS계층의



번호	INPUT/OUTPUT
1	BGN o/BGREJ, END/ENDAK, UD.req/UD, I11/END,I12/-
2	EST.req/BGN
3	BGN x/BGAK
4	UD.req/UD, I2/-
5	BGREJ/-
6	REL.req/END
7	BGAK/-, BGNx/BGAK
8	BGN o/{BGAK,RS}, RS/RSAK, UD.req/UD, I31/-
9	END/ENDAK, I32/-
10	REL.req/END
11	BGN x/BGAK
12	BGN o/{BGAK,END}, UD.req/UD, I4/-
13	ENDAK/-, BGREJ/-, END/ENDAK
14	EST.req/BGN
15	BGN x/BGAK
16	BGN o/BGAK, BGN x/BGAK, DATA.req/SD, RS o/RSAK, RS x/RSAK, ER o/ERAK, ER x/ERAK, UD.req/UD, SD/{USTAT, ER or -}, POLL/{STAT, ER or SD}, STAT/{ER,SD or -}, USTAT/{SD or ER}, I51/-
17	END/ENDAK, I52/-
18	EST.req/RS
19	REL.req/END

- : NULL, o : 재전송된 것 x : 재전송되지 않은 것
I11 : BGAK, RS, RSAK, ER, ERAK, SD, POLL, STAT, USTAT I12 : BGREJ, ENDAK, UD
I2 : END, ENDAK, RS, RSAK, ER, ERAK, SD, POLL, STAT, USTAT, BGN o, UD
I31 : BGAK, RS o, ER, ERAK, SD, POLL, RSAK, STAT, USTAT, UD
I32 : BGREJ, ENDAK I4 : BGAK, RS, RSAK, ER, ERAK, SD, POLL, STAT, USTAT, UD
I51 : BGAK, RSAK, ERAK, UD

그림 1. UNI의 SSCS계층의 FSM
Fig. 1. FSM of UNI SSCS



번호	INPUT/OUTPUT
1	{I11/END, BGAK/END}, {I12/-, BGAK/END}, {END/ENDAK, BGAK/END}, {UD.req/UD, BGAK/END}, {BGN o/BGREJ, BGAK/END}
2	{EST.req/BGN, END/-}
3	{BGN x/BGAK, ER/ERAK}
4	{I2/-, END/-}, {UD.req/UD, END/-}
5	{BGREJ/-, BGAK/END}
6	{REL.req/END, BGN o/{BGAK,END}}
7	{ BGAK/-, ER/ERAK}, {BGN x /BGAK, ER/ERAK}
8	{END/ENDAK, BGAK/END}, {I32/-, BGAK/END}
9	{REL.req/END, BGN o/{BGAK,END}}
10	{BGN x/BGAK, ER/ERAK}, {I31/-, BGN x/BGAK, ER/ERAK}, {BGN o/{BGAK, RS}, BGN x/BGAK, ER/ERAK}, {RS/RSAK, BGN x/BGAK, ER/ERAK}, {UD.req/UD, BGN x/BGAK, ER/ERAK}
11	{I4/-, BGN o/{BGAK,END}}, {BGN o/{BGAK,END}, BGN o/{BGAK,END}}, {UD.req/UD, BGN o/{BGAK,END}}
12	{ENDAK/-, BGAK/END}, {BGREJ/-, BGAK/END}, {END/ENDAK, BGAK/END}
13	{EST.req/BGN, END/-}
14	{BGN x/BGAK, ER/ERAK}
15	{EST.req/RS, BGN x/BGAK, ER/ERAK}, {BGN o/BGAK, ER/ERAK}, {BGN x/BGAK, ER/ERAK}, {DATA.req/SD, ER/ERAK}, {I51/-,ER/ERAK}, {RS o/RSAK, ER/ERAK}, {RS x/RSAK, ER/ERAK}, {ER o/ERAK, ER/ERAK}, {ER x/ERAK, ER/ERAK}, {UD.req/UD, ER/ERAK}, {SD/{USTATE, ER or -}, ER/ERAK}, {POLL/{STAT or ER}, ER/ERAK}, {STAT/{ER, SD or -}, ER/ERAK}, {USTAT/{SD or ER}, ER/ERAK}
16	{END/ENDAK, BGAK/END}, {I52/-, BGAK/END}
17	{REL.req/END, BGN o/{BGAK,END}}

그림 2. SUIO를 이용한 UNI의 SSCS계층의 test graph
Fig 2. Test graph of UNI SSCS using SUIO

FSM을 그림 1에 도시하였다. 그림 1의 FSM은 deterministic하며 위의 세가지 가정을 모두 만족한다. Deterministic FSM은 두개의 서로 다른 edge(V_i, V_{j1}; a_{k1}/o_{k1})∈ E와 (V_i, V_{j2}; a_{k2}/o_{k2})∈ E에 대하여 비록 o_{k1}과 o_{k2}가 같다하더라도 a_{k1}과 a_{k2}는 서로 다른 것을 뜻한다. FSM이 deterministic 성질을 가져야 상태천이는 초기상태와 일련의 input operation으로 구별될 수 있다. 그림 1의 FSM은 reset capability(어느 state에서 하나의 input operation으로 start state로 천이할 수 있다면 reset capability를 가졌다고 한다)를 가지지 않는데 이런 프로토콜은 각 state에 self-loop edge가 있고 stronly connected이면 minimum-cost의 test sequence를 구할 수 있다[1].

SUIO를 이용하여 UNI의 SSCS계층의 test sequence를 구해보자. 먼저, 표 2는 UNI의 SSCS계층의 UIO를 나타내고, 표 2를 이용하여 구한 test graphG"은 그림 2에 나타내었다. G"을 symmetric하게 만들기 위해 필요한 equation과 solution은 그림 3에 나타내었다. solution에 나타난 바는 symmetric test graph G*를 구하기 위하여 edge(V_1, V_2)에 4개, (V_2, V_4)에 2개 그리고 (V_5, V_3)에 18개 edge를 각각 G"에 추가하라는 것을 의미이다. 표 3은 앞에서 구한 symmetric test graph의 Euler Tour. 즉 UNI의 SSCS계층의 프로토콜 적합성 시험을 위한 test sequence를 보여주고 있으며 input은 총 245개이다.

2. MUIO를 이용한 Test Sequence생성

MUIO 를 이용한 test sequence생성 방법은 tail을 확인하기 위하여 여러개의 minimum-length UIO를 이용한다. $\zeta(V_i)$를 G에서 V_i로 들어오는 edge의 갯수와 V_i에서 나가는 edge갯수의 차이, 즉 $\zeta(V_i)=d^{Gc}_{in}(V_i)-d^{Gc}_{out}(V_i)$라 하자. 만약 G'의 모든 state에서 $\zeta(V_i)=0$이면 G*를 만들기 위해 G'에 어떤 edge도 첨가할 필요가 없다. 그러나 $\zeta(V_i)\neq 0$ 인 state가 있으면 edge를 첨가하여야 한다.

UIO^j_i를 state V_i에서 V_j를 tail state로 갖는 minimum-length UIO라 하자. 그러면 MUIO를 이용한 test sequence 생성 방법은 각 state V_j에서 $\Sigma^n_{i=1}|\zeta(V_i)|$를 최소로 하는 UIO^j_i를 구한다. 이를 위해 다음의 2단계를 수행한다.

1. Graph $G_M=(V_M, E_M)$를 구한다.
   여기서 $V_M=\{S, T\}\cup V_X\cup V_Y$,
   $V_X=\{X_1, X_2, \cdots, X_n\}$ and $V_Y=\{Y_1, Y_2, \cdots, Y_n\}$.
   $E_M=E_S\cup E_T\cup E'_T\cup E^{\wedge}$,
   $E_S=\{(S, X_i): X_i\in V_X\}$, $E_T=\{(Y_j, T): Y_j\in V_Y\}$,
   $E'_T=\{(Y_j, T): Y_j\in V_Y\}$, and $E^{\wedge}=\{(X_i, Y_j)\}$:There exists UIO^j_i}.
   $(S, X_i)\in E_S$인 edge는 0 cost 와 $d^g_{in}(V_i)$capacity, $(Y_j, T)\in E'_T$는 -1의 cost와 $d^g_{out}(V_j)$capacity, $(Y_j, T)\in E+T$는 +1의 cost와 infinity capacity를 그리고 $(X_i, Y_j)\in E^{\wedge}$인 edge는 0

표 2. UNI의 SSCS계층에 대한 UIO sequence
Table 2. UIO sequence of UNI SSCS

STATE	UIO SEQUENCE	NEXT STATE
V1	BGAK/END	V1
V2	END/-	V2
V3	BGN x/BGAK, ER/ERAK	V5
V4	BGN o/{BGAK,END}	V4
V5	ER/ERAK	V5

MIN $I_{12}+I_{15}+I_{21}+I_{24}+I_{25}+I_{31}+I_{34}+I_{35}+I_{41}+I_{45}+I_{51}+I_{53}+I_{54}$
S.T. $-I_{12}-I_{15}+I_{21}+I_{31}+I_{41}+I_{51}=-8$
$I_{12}-I_{21}-I_{24}-I_{25}+I_{42}=2$
$-I_{31}-I_{34}-I_{35}+I_{53}=18$
$I_{24}+I_{34}-I_{41}-I_{42}-I_{45}+I_{54}=2$
$I_{15}+I_{25}+I_{35}+I_{45}-I_{51}-I_{53}-I_{54}=-14$
$I_{ij}>=0$, integer for all i,j
Sol : $I_{12}=4$, $I_{15}=4$, $I_{21}=2$, $I_{53}=18$

그림 3. Symmetric graph를 구하는 equations과 solution
Fig. 3. Equations and solution finding a Symmetric graph

표 3. UNI의 SSCS계층의 test sequence(Euler Tour)
Table 3. Test sequence of UNI SSCS

1	{I11/END, BGAK/END}, {I12/-, BGAK/END}, {END/ENDAK, BGAK/END}, {UD.req/UD, BGAK/END}, {BGN o/BGREJ, BGAK/END}
2	{EST.req/BGN, END/-}
4	{I2/-, END/-}, {UD.req/UD, END/-}
5	{BGREJ/-, BGAK/END}
3	{BGN x/BGAK, ER/ERAK}
15	{EST.req/RS, BGN x/BGAK, ER/ERAK}, {BGN o/BGAK, ER/ERAK}, {BGN x/BGAK, ER/ERAK}, {DATA.req/SD, ER/ERAK},{I51/-,ER/ERAK}, {RS o/RSAK, ER/ERAK}, {RS x/RSAK, ER/ERAK}, {ER o/ERAK, ER/ERAK}, {ER x/ERAK, ER/ERAK}, {UD.req/UD, ER/ERAK}, {SD/{USTATE, ER or -}, ER/ERAK}, {POLL/{STAT or ER}, ER/ERAK}, {STAT/{ER, SD or -},ER/ERAK}, {USTAT/{SD or ER}, ER/ERAK}
16	{END/ENDAK, BGAK/END}
y12	
6	{REL.req/END, BGN o/{BGAK,END}}
11	{I4/-, BGN o/{BGAK,END}}, {BGN o/{BGAK,END}, BGN o/{BGAK,END}}, {UD.req/UD, BGN o/{BGAK,END}}
12	{ENDAK/-, BGAK/END}
y12, y24	
13	{EST.req/BGN, END/-}
7	{ BGAK/-, ER/ERAK}
16	{BGREJ/-, BGAK/END}
y12	
7	{BGN x /BGAK, ER/ERAK}
16	{ENDAK/-, BGAK/END}
y12, y24	
14	{BGN x/BGAK, ER/ERAK}
y53	
9	{REL.req/END, BGN o/{BGAK,END}}
12	{BGREJ/-, BGAK/END}
y15	
17	{REL.req/END, BGN o/{BGAK,END}}
12	{END/ENDAK, BGAK/END}
y15	

y53 {BGN x/BGAK, ER/ERAK},
y53 { BGAK/-, BGN x/BGAK, ER/ERAK},
y53 {ER/-, BGN x/BGAK, ER/ERAK},
y53 {SD/-, BGN x/BGAK, ER/ERAK},
y53 {RSAK/-, BGN x/BGAK, ER/ERAK},
y53 {USTAT/-, BGN x/BGAK, ER/ERAK},
y53 {RS/RSAK, BGN x/BGAK, ER/ERAK},
y53 {END/ENDAK, BGAK/END},
y15, y53 {ENDAK/-, BGAK/END}

y53 {BGN o/{BGAK, RS}},{BGNx/BGAK,ER/ERAK}
y53 { RS o/-, BGN x/BGAK, ER/ERAK},
y53 {ERAK/-, BGN x/BGAK, ER/ERAK},
y53 {POLL/-, BGN x/BGAK, ER/ERAK},
y53 {STAT/-, BGN x/BGAK, ER/ERAK},
y53 {UD/-, BGN x/BGAK, ER/ERAK},
y53 {UD.req/UD, BGN x/BGAK, ER/ERAK}
y15, y53 {BGREJ/-, BGAK/END}

cost와 infinite capacity를 갖는다.

2. F를 G_M의 flow라 하면 $\Sigma|\zeta(V_i)|$를 minimize하는 UIO을 찾기 위하여 다음의 equation을 푼다.[2].

$$min \quad \sum_{(Y_j,T)^+ \in E_T^+} F(Y_j,T)^+ - \sum_{(Y_j,T)^- \in E_T^-} F(Y_j,T)^- \qquad (5)$$

$$s.t. \quad F(S,X_i) = \sum_{(X_i,Y_j)\in E^\wedge} F(X_i,Y_j) \quad for\ X_i \in V_X \qquad (1)$$

$$F(Y_j,T)^+ + F(Y_j,T)^- = \sum_{(X_i,Y_j)\in E^\wedge} F(X_i,Y_j) \quad for\ Y_j \in V_Y \qquad (2)$$

$$F(S,X_i) \le d_{in}^E(V_i) \qquad for\ (S,X_i) \in E_S \qquad (3)$$

$$F(Y_j,T)^- \le d^E_{out}(V_j) \quad for\ (Y_j,T) \in E_T^- \qquad (4)$$

(1)과 (2)의 수식은 G_M의 모든 state는 들어오는 flow의 양과 나가는 flow의 양이 같다는 것을 나타내며 목적식은 가능한 한 기존의 edge $F(Y_j, T)$ 를 최대한 이용하고 새로이 추가되는 edge의 갯수 $F(Y_j, T)'$ 를 최소로 하는 것을 나타낸다. 또한 $E^{\wedge}$와 E_T는 infinite capacity를 가지므로 $f(X_i, Y_j) = d^E_{in}(V_i)$, 즉 G의 각 edge에 하나의 UIO가 할당되는 것을 나타낸다. 따라서 윗 식의 equation을 풀어서 얻은 solution $f(X_i, Y_j)$는 V_j로 들어오는 edge에 UIO_i^j를 $f(X_i, Y_j)$개 assign하여 test graph G'의 값을 구하라는 것을 의미한다.

MUIO를 이용하여 UNI의 SSCS계층의 test sequence를 구해보자. 먼저 표 4는 UNI의 SSCS계층의 minimum-length MUIO를 나타내었으며 $G_M=(V_M, E_M)$은 그림 4에 나타내었다. $\Sigma|\zeta(V_i)|$를 minimize하기 위해 필요한 equation과 solution은 그림 5에 나타내었다.solution에서도 나타났듯이 V_1로 들어오는 17개의 edge에 UIO_1^j, V_2로 들어오는 15개의 edge에 UIO_2^j, V_3로 들어오는 14개의 edge에 UIO_3^j, V_4로 들어오는 15개의 edge에 UIO_4^j 그리고 V_5로 들어오는 21개의 edge에 UIO_5^j 사용하여 test graph G'를 구한다.

Symmetric test graph를 만들기 위해 3.1절에서와 같이 solution을 구하면 edge (V_1, V_2)에 2개, (V_1, V_2)에 6개, (V_1, V_2)에 4개, 그리고 (V_1, V_2)에 2개의 edge를 첨가하여야 함을 알 수 있다. 따라서 총 14개의 edge가 새로이 첨가된다. 표 5는 minimum-length MUIO를 이용한 Euler Tour, 즉 UNI의 SSCS계층의 프로토콜 적합성 시험을 위한 test sequence를 나타내며 input은 총 231개이며 이는 SUIO를 이용하여 구한 것보다 약 6%정도 test sequence의 길이를 줄였다.

표 4. UNI의 SSCS계층의 MUIO
Table 4. MUIO of UNI SSCS

STATE	MUIO	NEXT STATE
V1	BGAK/END	V1
V2	END/-	V2
V3	BGN o/{BGAK, RS}	V3
	BGN x/BGAK, ER/ERAK	V5
V4	ENDAK/-, BGAK/END	V1
	EST.req/BGN,END/-	V2
	BGN o/{BGAK, END}	V4
V5	ER/ERAK	V5
	EST.req/RS	V3

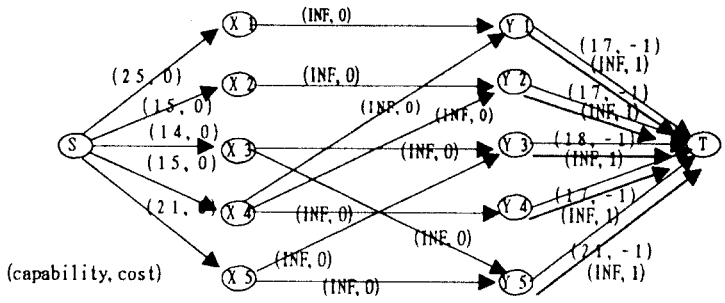


그림 4. UNI의 SSCS계층의 $G_M(V_M, E_M)$
Fig. 4. $G_M(V_M, E_M)$ of UNI SSCS

MIN $F(Y1,T)^+ + F(Y2,T)^+ + F(Y3,T)^+ + F(Y4,T)^+ + F(Y5,T) - F(Y1,T)^- - F(Y2,T)^- - F(Y3,T)^- - F(Y4,T)^- - F(Y5,T)^-$

S.T. (1) $F(S, X1) = F(X1, Y1)$
$F(S, X2) = F(X2, Y2)$
$F(S, X3) = F(X3, Y3) + F(X3, Y5)$
$F(S, X4) = F(X4, Y1) + F(X4, Y2) + F(X4, Y4)$
$F(S, X5) = F(X5, Y3) + F(X5, Y5)$

(2) $F(Y1,T)^+ + F(Y1,T)^- = F(X1, Y1) + F(X4, Y1)$
$F(Y2,T)^+ + F(Y2,T)^- = F(X2, Y2) + F(X4, Y2)$
$F(Y3,T)^+ + F(Y3,T)^- = F(X3, Y3) + F(X5, Y3)$
$F(Y4,T)^+ + F(Y4,T)^- = F(X4, Y4)$
$F(Y5,T)^+ + F(Y5,T)^- = F(X3, Y5) + F(X5, Y5)$

(3) $F(S, X1) <= 25$
$F(S, X2) <= 15$
$F(S, X3) <= 14$
$F(S, X4) <= 15$
$F(S, X5) <= 21$

(4) $F(Y1,T)^- <= 17$
$F(Y2,T)^- <= 17$
$F(Y3,T)^- <= 18$
$F(Y4,T)^- <= 17$
$F(Y5,T)^- <= 21$

모든 변수는 0보다 크거나 같은 integer

Sol : $f(Y1,T)^+ = 8$,
$f(Y1,T)^- = 17$, $f(Y2,T)^- = 15$, $f(Y3,T)^- = 14$, $f(Y4,T)^- = 15$, $f(Y5,T)^- = 21$

$f(X1, Y1)=17$, $f(X2, Y2) = 15$, $f(X3, Y3) = 14$, $f(X4, Y4) = 15$, $f(X5, Y5) = 21$

그림 5. $\Sigma_i|\zeta(V_i)|$를 minimize하는데 필요한 equations과 solution
Fig. 5. Equations and soluttion minimizing $\Sigma_i|\zeta(V_i)|$

표 5. MUIO를 이용한 UNI의 SSCS계층의 test sequence
Table. 5. Test sequence of UNI SSCS using MUIO

1 {BGN o/BGREJ, BGAK/END}, {END/ENDAK, BGAK/END}, {UD.req/UD, BGAK/END}, {I11/END, BGAK/END}, {I12/-, BGAK/END}
2 {EST.req/BGN, END/-}
4 {UD.req/UD, END/-}, { I2/-, END/-}
5 {BGREJ/-, BGAK/END}
y12
6 {REL.req/END, BGN o/{BGAK, END}}
14 {EST.req/BGN, END/-}
7 {BGAK/-,ER/ERAK}
16 {BGN o/BGAK,ER/ERAK}, { BGN x/BGAK,ER/ERAK}, { DATA.req/SD,ER/ERAK}, {RS o/RSAK,ER/ERAK}, {RS x/RSAK, ER/ERAK}, {ER o/ERAK, ER/ERAK}, {ER x/ERAK, ER/ERAK}, {UD.req/UD, ER/ERAK}, {SD/{USTAT, ER or -}, ER/ERAK}, {POLL/{STAT, ER or SD}, ER/ERAK}, {STAT/{ER,SD or -}, ER/ERAK}, {USTAT/{SD or ER}, ER/ERAK}, {I51/-,ER/ERAK}
17 {END/ENDAK, BGAK/END}
y12
7 {BGNx/BGAK, ER/ERAK}
17 {BGREJ/-, BGAK/END}
3 {BGN x/BGAK, ER/ERAK}
18 {EST.req/RS, BGN o/{BGAK,RS}}
8 {BGN o/{BGAK,RS},BGN o/{BGAK,RS}}, { RS/RSAK,BGN o/{BGAK,RS}}, { UD.req/UD, BGN o/{BGAK,RS}},{ I31/-, BGN o/{BGAK,RS}}
11 {BGN x/BGAK, ER/ERAK}
19 {REL.req/END, BGN o/{BGAK, END}}
12 {BGN o/{BGAK,END}, BGN o/{BGAK, END}}, { UD.req/UD, BGN o/{BGAK, END}}, {I4/-, BGN o/{BGAK, END}}
15 {BGN x/BGAK, ER/ERAK}
17 {ENDAK/-, BGAK/END}
y15, y53, 9 {END/ENDAK, BGAK/END}
y15, y53, 9 {BGREJ/-, BGAK/END}
y15, y53, 9 {ENDAK/-, BGAK/END}
y15, y53 10 {REL.req/END,BGN o/{BGAK, END}}
13 {ENDAK/-,BGAK/END}
y15, y54, {BGREJ/-, BGAK/END},
y15, y54 {END/ENDAK, BGAK/END}

프로토콜 구현물이 규격과 일치하는지를 결정하는 test sequence의 능력은 그것이 감지할 수 있는 fault나 error의 범위에 의존한다. 그러나 test sequence의 fault coverage의 estimation은 시험하여야 할 machine의 수가 많기 때문에 어려운 문제이다. 예를 들어 n개의 state, m개의 input과 p개의 output이 있는 규격은 $(np)^{nm}$개의 실현 가능한 구현물이 있을 수 있다.

따라서 이와 같은 모든 실현 가능한 구현물을 시험하는 것은 불가능하다. 그러므로 모든 가능한 faulty 구현물 대신 규격의 tail state(s)나 하나이상의 output을 수정하여 생성한 random faulty machine을 만들어 시험하며 그 category는 다음과 같다[4]. 여기서의 edge는 random edge를 의미한다.

1. 하나의 edge의 output을 수정
2. 하나의 edge의 tail state를 수정
3. 두개의 edge의 output을 수정
4. 두개의 edge의 tail states를 수정
5. 하나의 edge의 output과 또 다른 edge의 tail state를 수정
6. 하나의 edge의 output과 tail state를 수정
7. 두개의 edge의 output과 tail state를 수정
8. 하나의 edge의 output과 또 다른 두개의 edge의 tail state를 수정
9. 두개의 edge의 output과 또 다른 두개의 edge의 tail state를 수정
10. 세개의 edge의 output과 또 다른 두개의 edge의 tail state를 수정

모든 실현 가능한 faulty 구현물을 포함하기 위하여 extra 또는 missing state를 고려하여야 한다. missing state를 만들기 위하여 두개의 state를 merge하여야 한다. Merge된 state는 각 input에 대하여 두개의 원래 output중 하나만을 생성한다. Extra state를 만들기 위해서 하나의 state는 split되어야 하며 split된 두개으 새로운 state는 원래의 input이 명백하게 나누어지며 output은 원래의 것과 같다. Test sequence의 fault coverage를 추정하는 절차는 다음과 같다[4]

1. 규격의 FSM을 읽는다.
2. 생성된 test sequence를 읽는다.
3. 위에서 언급한 random faulty machine을 만든다.
4. 3번에서 생성한 machine에 test sequence를 적용하여 규격의 output과 일치하는지 비교한다.
5. 4번을 통과한 machine에 대하여 실제 규격을 conform하는지를(equivalent)체크한다. 이를 위해 다음의 algorithm을 적용한다. is_1을 machine F_1의 initial state라 하고 is_2를 machine F_2의 initial state라 하자.

```
Set={ }
Set2={is1, is2}
while (Set2≠{ }){
  Pick an element(s1, s2)∈Set2
  for (each outgoing edge e1 from s1) {
    if (there exists an outgoing edge e2 from s2
        with the same label as e1)
        tuple={tail(e1), tail(e2)}
        if( ~(tuple∈Set1) and~(tuple∈Set2))
          set2=Set2∪tuple
        set2=Set2-{(s1, s2)}
        set1=Set1∪{(s1, s2)}
    else F2 does not conform to F1
  }
}
F2 conforms to F1
```

Random하게 만든 machine중 몇몇은 faulty machine이 아닐 수 있으므로 Fault coverage는 실제 fault에 대하여 detect된 fault의 비율로 구한다. 따라서 r을 faulty machine의 갯수, c를 규격과 equivalent하다고 판단된 machine의 갯수 그리고 d를 test sequence에 의해 fault라고 판단된 machine의 갯수라 하자. 그러면 fault coverage는 $d/(r-c)$라 할 수 있다.

SUIO와 MUIO를 이용하여 구한 B-ISDN UNI의 SSCS계층의 test sequence에 적용한 fault coverage의 결과로써, 규격과 동일한 갯수의 state를 갖는 10000개의 faulty machine에 대하여 위에서 언급한 10가지의 fault class를 적용한 결과는 표 6에 나타내었고 extra state를 갖는 faulty machine에 대한 결과는 표 7에 나타내었다. Merge state의 경우는 모든

방법에 대하여 100%의 fault coverage를 갖는다.

Extra state를 가진 구현물에 대한 test sequence의 fault coverage는 기대한 바와 같이 일반적으로 약함을 알 수 있다. SUIO와 MUIO를 이용하여 구한 test sequence 모두 class 1과 3에 해당하는 fault를 100% 발견함을 알 수 있다. Class 2에 대하여는 각각 85%와 97%의 fault coverage를 보이고 있으며 복수개의 fault(class 4~class 10)에 대하여는 각각 98.9%와 99.4%의 fault coverage를 보이고 있다. 따라서 SSCS계층의 프로토콜 구현물의 적합성 시험을 위해 SUIO를 이용하는 것보다 MUIO를 이용하여 구한 test sequence를 적용하는 것이 좋음을 알 수 있다.

Ⅴ. 결 론

표 6. SUIO와 MUIO의 fault coverage(동일한 갯수의 STATE를 가지는 경우)
Table. 6. Fault coverage of SUIO and MUIO(When the implement has the same number of states as the spec.)

Test Class	랜덤하게 생성된 m/c의 수		Test Seq.를 통과한 수		m/c Equ.를 통과한 수	
	SUIO	MUIO	SUIO	MUIO	SUIO	MUIO
1	10,000	10,000	481	481	481	481
2	10,000	10,000	3,198	2,175	2,027	1,975
3	10,000	10,000	3	8	3	8
4	10,000	10,000	1,004	461	392	377
5	10,000	10,000	151	131	95	75
6	10,000	10,000	153	336	91	109
7	10,000	10,000	6	10	1	1
8	10,000	10,000	7	16	6	7
9	10,000	10,000	6	1	0	0
10	10,000	10,000	0	0	0	0

표 7. SUIO와 MUIO의 fault coverage(EXTRA STATE를 가지는 경우)
Table. 7. Fault coverage of SUIO and MUIO(When the implement has one more states than the spec.)

Test Class	랜덤하게 생성된 m/c의 수		Test Seq.를 통과한 수		m/c Equ.를 통과한 수	
	SUIO	MUIO	SUIO	MUIO	SUIO	MUIO
1	10,000	10,000	2,096	2,249	509	509
2	10,000	10,000	4,276	3,328	1,997	1,954
3	10,000	10,000	313	358	6	7
4	10,000	10,000	2,093	1,168	300	383
5	10,000	10,000	844	704	95	103
6	10,000	10,000	1,692	1,797	82	83
7	10,000	10,000	312	336	1	2
8	10,000	10,000	167	159	4	6
9	10,000	10,000	67	33	4	1
10	10,000	10,000	30	12	1	0

프로토콜 적합성 시험을 위하여 test sequence를 생성하는 여러 방법들이 제안되었다. Test sequence는 FSM의 각 transition을 적어도 한번 지나는 것을 보장하는 approximation algorithm에 의해 생성된다. [1]에서는 프로토콜 구현물의 적합성 시험을 위하여 프로토콜이 기대되는 state에 있는지를 검증하기 위하여 SUIO를 이용하는 것을 제안하였고 [2]에서는 MUIO를 이용하는 방법을 제안하였다

본 고에서는 case study의 하나로써, B-ISDN의 UNI SSCS계층 프로토콜에 앞에서 언급한 2가지 방법을 적용하여 test sequence를 구하였으며, 이때 생기는 문제점을 제시하였다. 특히 내부 변수에 관한 문제는 변수에 상관없이 여러 가능한 output중 하나를 받아도 규격을 만족하는 것으로 가정하여 SSCS계층의 프로토콜 FSM으로 처리하였으나, 이 방법은 내부 변수 문제를 완전히 해결한 것이 아니므로 이에 대하여 더 많은 연구를 필요로 한다.

MUIO를 이용하여 구한 test sequence의 길이는 SUIO를 이용하여 구한 것보다 6%정도 짧았다. 또한 두개의 test sequence는 하나의 fault(class 1~class 3)에 대하여 95%와 99%, 그리고 복수개의 fault에 대하여 98.8%와 99.4%의 fault coverage를 보였다. 이는 SSCS계층의 프로토콜 구현물의 적합성 시험은 SUIO를 이용하는 것보다 MUIO를 이용하여 구한 test sequence를 적용하는 것이 좋음을 나타낸다. 앞에서 언급한 방법으로 생성된 test sequence는 사람의 조정없이 자동적으로 작동할 수 있는 하나의 큰 test sequence로써, 만일 error를 발견한 후에도 계속 시험하는 것은 시간을 낭비하는 일이 된다. 이 문제는 test sequence를 작은 독립된 test script로 나누면 해결되는데 이 경우에 alignment part 즉, 모든 test가 원하는 state에서 시작되도록 하는 노력을 필요로 한다. 따라서 가능한 alignment part를 적게 하면서 많은 독립된 test script를 구하는 연구가 이루어져야 할 것이다.

참고문헌

1. Alfred Aho, Anton T. Dahbura, David Lee, and Umit Uyar, "An Optimization Technique for Protocol Conformance Test Generation Based on UIO sequence and Rural Postman Tours", IEEE Transaction on Communications, VOL 39. No.11. Nov. 1991.
2. Y.N. Shen, F.Lombardi, "Protocol Testing Using Multiple UIO Sequence", Protocol Specification Testing and Verification, Ⅸ, North-Holland, 1989.
3. M.Umit Uyar, Anton T.Dahbura, "Optimal Test Sequence Generation Protocols:The Chinese Postman Algorithm Applied to Q.931", proceedings of the IEEE Global Communications Conference, Vol.1, 1986.
4. Howard Motteler, Anthony Chung, and Deepinder Sidhu, "Fault Coverage of UIO-based Methods for Protocol Testing", International IFIP Workshop on Protocol Test System, Ⅵ, 1993.
5. Deepinder P. Sidhu and Ting-Kau Leung "Formal Methods for Protocol Testing:A Detailed Study" IEEE Transactions on Software Engineering Vol. 15, No 4, April. 1989.
6. ITU-T SG 11, WP3/11 Geneva, Nov. 29, 1993, Q.2100.
7. ITU-T SG 11, WP3/11 Geneva, Nov. 29, 1993, Q.2110.
8. ITU-T SG 11, WP3/11 Geneva, Nov. 29, 1993, Q.2130.

鄭 允 姬 (Yoon Hee Jung) 정회원

1967년 12월 15일생

1990년 : 한양대학교 산업공학과(학사)

1992년 : 한국과학기술원 산업공학과(석사)

1992년~현재 : 한국전자통신연구소 초고속정보통신연구본부 초고속망연구실 연구원

李 庚 熙 (Kyung Hee Lee) 정회원

1983년 2월 : 경북대학교 전자공학과(학사)

1990년 2월 : 충남대학교 계산통계학과(석사)

1993년 8월 : 충남대학교 전산학과(박사)

1993년 3월~현재 : 한국전자통신연구소 초고속망연구실 선임연구원

洪 范 基 (Beom Kee Hong) 정회원

1982년 2월 : 홍익대학교 전자계산학과(학사)

1984년 8월 : 홍익대학교 전자계산학과(석사)

1982년 3월~현재 : 한국전자통신연구소 초고속망연구실 선임연구원