KAERI/TR-745/96

TRANSX 코드의 군축약 기능 개선

Improvement of Group Collapsing in TRANSX Code

1996. 7.

한국원자력연구소

제 출 문

한국원자력연구소장 귀하

본 보고서를 "TRANSX코드의 군축약 기능 개선"에 대한 기술보고서로 제출합니다.

1996. 7.

주 저 자 : 정현태(액체금속로설계기술개발)
공동저자 : 김영철(액체금속로설계기술개발)
김영인(액체금속로설계기술개발)
김영균(액체금속로설계기술개발)

책임감수위원 : 김 동 훈
감 수 위 원 : 김 영 진

요 약

한국원자력연구소 액체금속로 노심설계기술개발 분야에서 구축중인 노심설계 계산체제 K-CORE 시스템의 유효단면적 생성 및 처리 모듈체계의 구성 전산코드 TRANSX Version 2.15는 단면적 라이브러리 MATXS를 사용하여 중성자속 계산모듈들이 요구하는 단면적 입력파일들을 생산하는 전산코드이다. TRANSX는 K-CORE 시스템에서 군축약에 사용하는 비중함수로 double precision으로 쓰여진 구역 평균 중성자속 파일 RZFLUX를 처리하는 기능이 없다. 따라서, 본 보고서에서는 TRANSX를 이용한 RZFLUX 처리 기능을 개발, 보강함으로써 K-CORE 시스템의 군축약 기능을 개선하였다. 이로써, 단면적 라이브러리 파일인 MATXS, 유효단면적 생성 및 처리 전산코드 TRANSX, 수송 이론 계산 코드인 TWODANT로 이어지는 계산체제에 RZFLUX를 이용한 순환 연계 계산이 가능하도록 하였으며, 보기 모형 계산을 통하여 이를 확인하였다.

ABSTRACT

A cross section generating and processing computer code TRANSX version 2.15 in the K-CORE system, being developed by the KAERI LMR core design technology development team produces various cross section input files appropriate for flux calculation options from the cross section library MATXS. In this report, a group collapsing function of TRANSX has been improved to utilize the zone averaged flux file RZFLUX written in double precision as flux weighting functions. As a result, an iterative calculation system using double precision RZFLUX consisting of the cross section data library file MATXS, the effective cross section producing and processing code TRANSX, and the transport theory calculation code TWODANT has been set up and verified through a sample model calculation.

목 차

1. 서 론

원자로에서 중성자의 물리적 특성들을 분석하기 위한 전산코드들을 이용하는 데 있어서 가장 기본이 되는 자료가 바로 중성자에 대한 여러 가지 원소의 핵반응 단면적 자료이다. 따라서 계산코드들과 반응단면적 자료는 서로 보완적 관계를 유지하면서 계속 새롭게 개선되어 왔다. 그러나, 여러 다른 사용자 그룹이 개발에 참여하였기 때문에 단면적 자료의 형식이 하나로 통일되어 개발되지 못하고 개발자의 목적과 사용 분야에 따라 서로 다른 형태로 발전되었으며, 이들 다양한 단면적 자료와 계산코드를 연결하는 것이 또 하나의 중요한 문제가 되었다. 더욱이 동일한 단면적 자료를 이용할 때도 사용하는 계산코드에 따라 입력 파일의 형식이 달라서 사용자는 주어진 단면적 자료에서 필요한 단면적들을 골라내고 다시 이를 그가 사용하고자 하는 계산코드가 허용하는 형식으로 변환하는 작업을 해야만 한다. TRANSX (Transport Cross Section) 전산코드[1]는 이런 문제를 해결하기 위하여 LANL(Los Alamos National Laboratory)에서 개발한 코드로써, NJOY 91의 형식을 만족하도록 LANL에서 새롭게 정의한 범용 단면적 라이브러리형식인 MATXS(Material Cross Section)[1]를 이용하여 ONEDANT[2]나 TWODANT[2] 또는 DIF-3D와 같은 코드들이 요구하는 입력 파일을 만들어 준다. 사용자는 간편하게 TRANSX가 생성하는 단면적 라이브러리에서 원하는 형식의 입력 파일을 얻을 수 있게 되어, 단면적 자료 파일이나 입력 파일의 상세한 형식에 맞추기 위한 시간과 노력을 크게 절약할 수 있다.

TRANSX는 MATXS에 있는 단면적 자료들을 정리하여 입력 파일을 만드는 과정에서 단순한 형식 변환이외에 자기차폐 단면적 계산, Dancoff 수정, 탄성충돌 수정, 군축약 등을 할 수 있다. 이 중에서 군축약은 특히 중요한 과정이다. 중성자속을 계산하는 과정의 부정확성을 줄이기 위해서는 가능한 한 작은 에너지 구간을 가지는 미세에너지 군구조를 사용하는 것이 바람직하지만, 에너

지 구간을 작게 할수록 계산의 수행시간이 길어지고 또한 필요한 컴퓨터 메모리의 양이 증가하여 많은 시간과 비용이 소모된다. 이런 경우에 주어진 단면적 자료를 좀 더 큰 에너지 구간을 가지는 군 구조로 재정리하는 것이 군축약이다. 군축약 과정에서는 각 군의 단면적을 합칠 때 일정한 비중함수(weighting function)를 필요로 하며 TRANSX는 이 비중함수로 자신의 고유한 모형계산함수 외에 CCCC(Committee for Computer Code Coordination)형식의 연계 파일인 RTFLUX[3], RZFLUX[3]와 LANL에서 정의하여 사용하는 RZMFLX[1] 형식으로 쓰여진 파일들로부터 비중함수를 읽어들인다. K-CORE 시스템에서 문제가 되는 점은 TRANSX version 2.15가 사용하는 RZFLUX와 S_N 수송 이론 계산 전산코드의 하나인 TWODANT가 사용하는 RZFLUX가 서로 호환되지 않는다는 사실이다. TRANSX에서는 RZFLUX의 중성자속을 single precision으로 처리하는데 비해 TWODANT에서는 이를 double precision으로 처리하기 때문에 두 전산코드들이 생성한 각각의 RZFLUX파일들은 상호 교환되지 않는다. TRANSX를 사용하여 단면적 자료를 군축약하기 위해서는 비중함수가 필요하다. 이 비중함수로는 TWODANT에서 얻은 RZFLUX를 사용하는 것이 군축약 과정에서 가장 바람직하며 TRANSX와 TWODANT를 반복 사용하여 바람직한 결과를 얻을 수 있다. 하지만 이 두 코드가 결과를 공유하지 못하면 이러한 반복의 고리가 끊어지므로 군축약 기능을 갖게 하기 위해서는 두 코드가 같은 형식의 RZFLUX를 사용하도록 하는 것이 매우 중요하다.

본 보고서에서는 위와 같은 목적을 달성하기 위하여 TRANSX 코드를 수정·개량한 내용과 그 결과를 보여주고 있다. 먼저, 제 2장에서는 CCCC형식의 표준 RZFLUX 파일의 내용을 설명하고 있으며, 제 3장에서는 수송이론계산코드 TWODANT가 RZFLUX를 출력하는 방법을 설명하고 있다. 제 4장에서는 TRANSX가 RZFLUX를 읽어들이는 방법을 검토하고 표준화된 K-CORE 시스템으로 병합하기위해 수정·개량한 내용을 다루었다. 제 5장에서는 군축약 기능이 개선된 TRANSX를 사용하여 계산 결과를 확인하였으며, 자유로운 파일 교환을 이룬 예로서 RTFLUX와 RZFLUX를 사용하여 보기 모형의 k_{eff} 값을 계산한 예를 보였다.

참고로 본 보고서 작성을 위한 컴퓨터 계산은 HP-UX 10.01을 OS로 하여 운영되는 HP 9000 Series 700 J200 워크스테이션을 사용하였고, HP f77과 HP cc를 사용하여 실행코드를 작성하였으며, 코드 분석은 HP에서 제공하는 DDE(Distributed Debugging Environment)를 사용하였다.

2. 표준 RZFLUX의 구조

RZFLUX 파일은 CCCC에서 정의한 중성자속 기록 파일 중 하나이다. 전산코드 TWODANT와 TRANSX에서 RZFLUX 파일을 다루는 방법의 차이점을 이해하기 위하여 표준 RZFLUX[3] 파일 구조를 여기에 명시한다. 여기서 우리가 주목해야 할 것은 모든 실변수들의 정밀도가 명시되어 있지 않다는 사실이다. 예를 들어 실제 중성자속을 나타내는 배열 ZGF는 실변수이지만, 이것이 single precision인지 double precision 변수인지는 분명하게 정해져 있지 않기 때문에 코드에 따라 이를 다르게 정의할 수 있으며, 그런 경우에는 그 전산코드들이 생성한 RZFLUX는 호환성을 가지지 못한다.

RZFLUX

****

REVISED 11/30/76

RZFLUX-IV

REGULAR ZONE FLUX BY GROUP, AVERAGED OVER EACH ZONE

****

FILE IDENTIFICATION

HNAME, (HUSE(I), I=1,2), IVERS

→ 1 + 3*MULT

HNAME	HOLLERITH FILE NAME - RZFLUX - (A6)
HUSE	HOLLERITH USER IDENTIFICATION (A6)
IVERS	FILE VERSION NUMBER

MULT	DOUBLE PRECISION PARAMETER 1 - A6 WORD IS SINGLE WORD 2 - A6 WORD IS DOUBLE PRECISION WORD

SPECIFICATIONS (1D RECORD)

TIME,POWER,VOL,EFFK,EIVS,DKDS,TNL,TNA,TNSL,TNBL,TNBAL,TNCRA,
1(X(I),I=1,3),NBLOK,ITPS,NZONE,NGROUP,NCY

→ 20 = NUMBER OF WORDS

TIME	REFERENCE REAL TIME, DAYS
POWER	POWER LEVEL FOR ACTUAL NEUTRONICS PROBLEM, WATTS THERMAL
VOL	VOLUME OVER WHICH POWER WAS DETERMINED, CC
EFFK	EFFECTIVE MULTIPLICATION FACTOR
EIVS	EIGENVALUE OF SEARCH PROBLEM
DKDS	DERIVATIVE OF SEARCH PROBLEM
TNL	TOTAL NEUTRON LOSSES
TNA	TOTAL NEUTRON ABSORPTIONS
TNSL	TOTAL NEUTRON SURFACE LEAKAGES
TNBL	TOTAL NEUTRON BUCKLING LOSS
TNBAL	TOTAL NEUTRON BLACK ABSORBER LOSS
TNCRA	TOTAL NEUTRON CONTROL ROD ABSORPTIONS
X(I), I=1,3	RESERVED
NBLOK	DATA BLOCKING FACTOR. THE GEOMETRIC ZONE VARIABLE IS BLOCKED INTO NBLOK BLOCKS.
ITPS	ITERATIVE PROCESS STATE = 0, NO ITERATIONS DONE = 1, CONVERGENCE SATISFIED = 2, NOT CONVERGED, BUT CONVERGING = 3, NOT CONVERGED, NOT CONVERGING
NZONE	NUMBER OF GEOMETRIC ZONES
NGROUP	NUMBER OF ENERGY GROUPS
NCY	REFERENCE COUNT (CYCLE NUMBER)

FLUX VALUES (2D RECORD)

((ZGF(K,J), K=1,NGROUP), J=JL,JU) --- SEE INSTRUCTIONS BELOW ---

NGROUP*(JU-JL+1) = NUMBER OF WORDS

```
DO 1 M=1,NBLOK
1   READ(N) *LIST AS ABOVE*
```

WITH M AS THE BLOCK INDEX, JL=(M-1)*((NZONE-1)/NBLOK+1)+1
AND JU=MIN0(NZONE,JUP) WHERE JUP=M*((NZONE-1)/NBLOK+1)

ZGF(K,J) REGULAR ZONE FLUX BY GROUP, AVERAGED OVER ZONE.
NEUTRONS/SEC-CM**2.

3. TWODANT의 RZFLUX

현재 우리가 사용하는 수송이론 계산코드 TWODANT는 1984년 University of California에서 개발한 것으로 여러 개의 포트란 파일과 하나의 C 파일이 결합하여 하나의 실행파일을 만드는 형식을 취하고 있다. 여기서는 TWODANT가 중성자속을 RZFLUX형식으로 출력하기 위해 사용하는 순차적 구조를 살펴보고, 직접 RZFLUX를 작성하는 부프로그램을 분석하여 그 특성을 살펴본다.

3.1 RZFLUX 생성 구조

TWODANT의 실행 파일은 다음과 같은 포트란 원시코드 5개와 C 원시코드 1 개를 각각 컴파일한 뒤에 형성된 목적 파일을 결합하여 만들어진다.

cdriver.f cinps.f csolvno.f csolvop.f couts.f morec.c

여기에서 cdriver.f는 프로그램의 전체적 진행을 총괄하여 다른 작업들을 불러 실행하거나 종료시키는 역할을 한다. cinps.f는 프로그램에 대한 입력을 다루고 couts.f는 출력을 책임진다. csolvno.f와 csolvop.f는 TWODANT 코드의 핵심으로서, 1차원 다군 중성 입자 볼츠만 수송 방정식을 푸는 일을 한다. morec.c는 포트란에서 다루기 힘든 컴퓨터 메모리를 확보하고 풀어주는 작업을 한다.

RZFLUX는 TWODANT를 수행하여 얻어지는 결과의 일부이므로 당연히 couts.f 에서 만들어지고, couts.f 중에서도 서브루틴 RZRITE가 RZFLUX파일을 만드는 일을 한다. TWODANT에서 RZRITE가 불리는 구조 및 순서는 다음 그림 1과 같다.

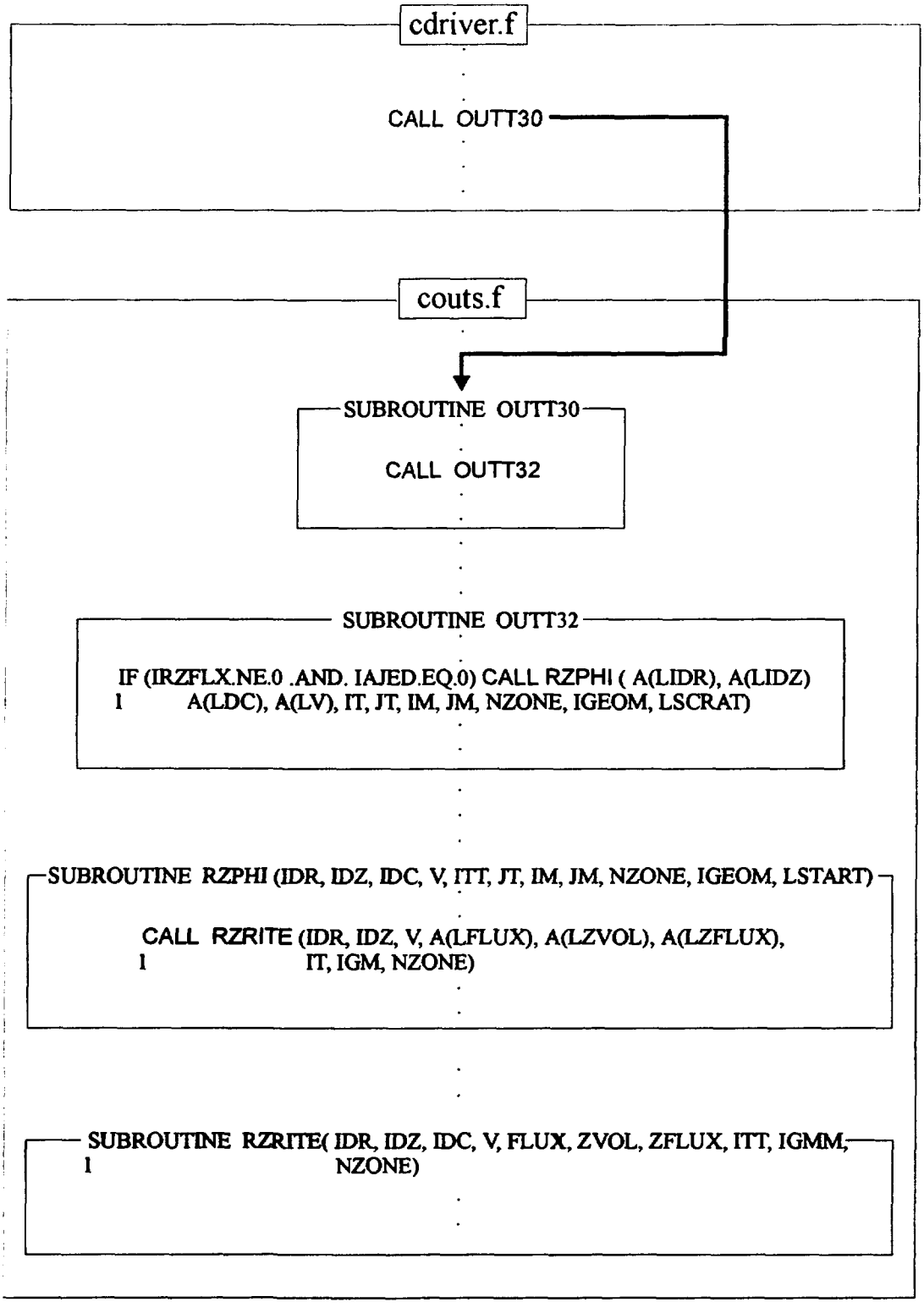

그림 1. TWODANT에서 RZRITE가 수행되는 순서도.

출력을 담당하는 모듈 couts.f는 세 개의 중요한 서브루틴; OUTT30, OUTT31, OUTT32를 포함하고 있다. OUTT30은 출력 모듈을 총괄하는 드라이버로서 여러 서브모듈들을 부르고 종료시키면서 출력 프로그램의 흐름을 조절한다. OUTT31은 반응률을 계산하며, OUTT32는 출력을 규격화하고 구역 평균값들(zone averaged values)을 구한 뒤 여러 가지 출력 파일들을 준비한다. 서브루틴 RZPHI는 RZFLUX를 출력하기 위한 드라이버로서, 출력 파일을 만들기 전에 필요한 임시 저장 영역을 확보하고 RZRITE를 불러서 RZFLUX를 출력한 다음 저장 영역을 다시 풀어주는 일을 한다. 실제로 RZFLUX를 작성하는 일은 서브루틴 RZRITE가 하므로 다음절에서는 이 서브루틴을 자세하게 살펴보도록 한다.

3.2 서브루틴 RZRITE

서브루틴 RZRITE는 250줄에 달하는 다소 긴 것이지만 TWODANT가 RZFLUX를 다루는 방법을 살펴보기 위해 여기에 포함시킨다.

```
      SUBROUTINE RZRITE (IDR, IDZ, IDC, V, FLUX, ZVOL, ZFLUX, ITT, IGMM,     5370.0
     1                   NZONE)                                              5371.0
      IMPLICIT DOUBLE PRECISION ( A-H, O-Z )                                 5372.0
      SAVE                                                                   5373.0
C*************************************************************************  5374.0
C                                                                            5375.0
C     WRITES THE STANDARD INTERFACE FILE RZFLUX                              5376.0
C          1. READS RTFLUX                                                   5377.0
C          2. INTEGRATES OVER ZONES                                          5378.0
C          3. WRITES RZFLUX                                                  5379.0
C                                                                            5380.0
C*************************************************************************  5381.0
C                                                                            5382.0
      PARAMETER (NCSIZE=8)                                                   5383.0
C                                                                            5384.0
      COMMON / IAE / IA ( 200 )                                              5385.0
C                                                                            5386.0
CCOSDS                                                                       5387.0
      COMMON / IBMPCM / A ( 1 )                                              5388.0
```

```
CCOSDE                                                                  5389.0
CCOSCCCOMMON / POINTA / IPTTOA                                          5390.0
C                                                                       5391.0
CCOSCCPOINTER ( IPTTOA, A ( 1 ) )                                       5392.0
C                                                                       5393.0
      EQUIVALENCE  (IA( 1),IEDOPT), (IA( 2),IPTED ), (IA( 3),NIPE  ),   5394.0
     1             (IA( 4),IKND  ), (IA( 5),IZNED ), (IA( 6),NZNS  ),   5395.0
     2             (IA( 7),IXSUM ), (IA( 8),IRSUM ), (IA( 9),NPOS  ),   5396.0
     3             (IA(10),NMAT  ), (IA(11),NCONS ), (IA(12),MACRO ),   5397.0
     4             (IA(13),IDOSE ), (IA(14),IGRPED), (IA(15),LNG   ),   5398.0
     5             (IA(16),NIGM  ), (IA(17),IDEN  ), (IA(18),IGM   ),   5399.0
     6             (IA(19),IAJED ), (IA(21),IHE   ), (IA(22),NON   ),   5400.0
     7             (IA(23),NEDT  ), (IA(24),IT),(IA(25),IEDOTF)         5401.0
C                                                                       5402.0
      EQUIVALENCE (IA(151),LNGGAL)                                      5403.0
C                                                                       5404.0
      CHARACTER *(NCSIZE) FNREF, FNLOC, FNREN, HOLOUT, HOLIN            5405.0
C                                                                       5406.0
      COMMON / SEEKIT / FNREF ( 70 ), FNLOC ( 70 ), FNREN ( 20 ),       5407.0
     IHOLOUT ( 4 ), HOLIN ( 4 )                                         5408.0
C                                                                       5409.0
      COMMON / SEEKNT / NFNREF                                          5410.0
C                                                                       5411.0
      COMMON / SEEKXT / JVERWT                                          5412.0
      COMMON /UNITS/ NINP, NOUT, NTTY, NXLIB, NBCDOT, NBCDOX            5413.0
C                                                                       5414.0
      COMMON / INSTAL / IOLYCS, IRCOVY, LEDINF, IDROPX, INSLS,          5415.0
     1INLLS, INLLP, INLLX, INLLI, INLLD,                                5416.0
     2IBASCM, IBALCM, IBXSCM, IBXLCM, IBSSCM, IBSLCM, IBOSCM, IBOLCM,   5417.0
     3IDFSCM, IDFLCM, IFREVS                                            5418.0
C                                                                       5419.0
C                                                                       5420.0
      CHARACTER *(NCSIZE) WHERE                                         5421.0
C                                                                       5422.0
      DIMENSION FLUX(ITT,1), IDR(1), IDZ(1), IDC(1), ISPEC(20), ZVOL(1),  5423.0
     1    V(ITT,1), ZFLUX(IGMM,1)                                       5424.0
C                                                                       5425.0
      INTEGER G, FIRST                                                  5426.0
C                                                                       5427.0
      DATA WHERE /'RZRITE'/                                             5428.0
C                                                                       5429.0
C************************************************************************5430.0
C                                                                       5431.0
C  *** DONT WRITE IF ADJOINT...                                         5432.0
      IF (IAJED.NE.0) GO TO 500                                         5433.0
C                                                                       5434.0
```

```
C                                                                       5435.0
C  *** READ FLUX FILE, GET ZONE FLUX CONTRIBUTIONS...                   5436.0
C                                                                       5437.0
      M=5                                                               5438.0
      IF (IAJED.NE.0) M=6                                               5439.0
C                                                                       5440.0
           CALL OPENRD (M,IFILN,JRECF,IVERF,WHERE)                      5441.0
C                                                                       5442.0
      CALL SREED ( IFILN, ISPEC, 0, 9, 0, 1, JRECF )                    5443.0
      NDIM=ISPEC(1)                                                     5444.0
      NGROUP=ISPEC(2)                                                   5445.0
      NINTI=ISPEC(3)                                                    5446.0
      NINTJ=ISPEC(4)                                                    5447.0
      NINTK=ISPEC(5)                                                    5448.0
      CALL FEBYTE (ISPEC(7),EV,1)                                       5449.0
      CALL FEBYTE (ISPEC(8),POWER,1)                                    5450.0
      NBLOK=ISPEC(9)                                                    5451.0
C                                                                       5452.0
      IF ( NBLOK .LE. 0 ) NBLOK = 1                                     5453.0
      CALL FILECK(0,NGROUP,0,0,0,0,0,NINTI,0,NINTJ,0,0,FNREF(5),        5454.0
     1   'RZRITE',1)                                                    5455.0
      IF (NOWERR(1).NE.0) GO TO 400                                     5456.0
C                                                                       5457.0
      IF (NDIM.GT.1) GO TO 200                                          5458.0
C                                                                       5459.0
C***********************************************************************5460.0
C                                                                       5461.0
C  *** 1D FLUX, POSSIBLY BLOCKED BY GROUP ...                           5462.0
C                                                                       5463.0
      CALL CLEAR (0.0, ZFLUX, NGROUP*NZONE)                             5464.0
C                                                                       5465.0
C     NUMBER OF GROUPS IN A FULL BLOCK                                  5466.0
C                                                                       5467.0
      NGRPS = (NGROUP-1)/NBLOK + 1                                      5468.0
C                                                                       5469.0
      LAST=0                                                            5470.0
      DO 160 N = 1, NBLOK                                               5471.0
      FIRST=LAST+1                                                      5472.0
      LAST=MIN0(FIRST+NGRPS,NGROUP)                                     5473.0
      LEN = (LAST-FIRST+1)*NINTI                                        5474.0
C                                                                       5475.0
      CALL SREED ( IFILN, FLUX, LEN, 0, 0, 1, JRECF)                    5476.0
C                                                                       5477.0
C     ADD IN FLUX FROM THIS BLOCK                                       5478.0
C                                                                       5479.0
      DO 150 I=1,NINTI                                                  5480.0
```

```
      IB=IDR(I)                                                          5481.0
      IB=IDC(IB)                                                         5482.0
      IF (IB.EQ.0) GO TO 150                                             5483.0
C                                                                        5484.0
      DO 140 G=FIRST,LAST                                                5485.0
      ZFLUX(G,IB) = ZFLUX(G,IB) + FLUX(I,G-FIRST+1)*V(I,1)               5486.0
  140 CONTINUE                                                           5487.0
  150 CONTINUE                                                           5488.0
C                                                                        5489.0
  160 CONTINUE                                                           5490.0
      GO TO 300                                                          5491.0
C                                                                        5492.0
C************************************************************************5493.0
C                                                                        5494.0
C  *** 2D OR 3D FLUX, POSSIBLY BLOCKED BY LINES...                       5495.0
C                                                                        5496.0
  200 CALL CLEAR (0.0, ZFLUX, NGROUP*NZONE)                              5497.0
C                                                                        5498.0
C     NUMBER OF LINES IN A FULL BLOCK                                    5499.0
C                                                                        5500.0
      NLINES = (NINTJ-1)/NBLOK + 1                                       5501.0
C                                                                        5502.0
      DO 280 G=1,NGROUP                                                  5503.0
      DO 270 K=1,NINTK                                                   5504.0
C                                                                        5505.0
      LAST=0                                                             5506.0
      DO 260 N = 1, NBLOK                                                5507.0
      FIRST=LAST+1                                                       5508.0
      LAST=MIN0(FIRST+NLINES,NINTJ)                                      5509.0
      LEN = (LAST-FIRST+1)*NINTI                                         5510.0
C                                                                        5511.0
      CALL SREED ( IFILN, FLUX, LEN, 0, 0, 1, JRECF)                     5512.0
C                                                                        5513.0
C     ADD IN FLUX FROM THIS BLOCK                                        5514.0
C                                                                        5515.0
      DO 250 I=1,NINTI                                                   5516.0
      DO 240 J=FIRST,LAST                                                5517.0
      IB=IDR(I)+IDZ(J)                                                   5518.0
      IB=IDC(IB)                                                         5519.0
      IF (IB.EQ.0) GO TO 240                                             5520.0
C                                                                        5521.0
      ZFLUX(G,IB) = ZFLUX(G,IB) + FLUX(I,J-FIRST+1)*V(I,J)               5522.0
C                                                                        5523.0
  240 CONTINUE                                                           5524.0
  250 CONTINUE                                                           5525.0
C                                                                        5526.0
```

```
  260 CONTINUE                                                          5527.0
C                                                                       5528.0
  270 CONTINUE                                                          5529.0
  280 CONTINUE                                                          5530.0
C                                                                       5531.0
C***********************************************************************5532.0
C                                                                       5533.0
C     CALCULATE ZONE VOLUMES                                            5534.0
C                                                                       5535.0
  300 DO 305 J=1,NINTJ                                                  5536.0
      DO 304 I=1,NINTI                                                  5537.0
      IB=IDR(I)+ IDZ(J)                                                 5538.0
      IB=IDC(IB)                                                        5539.0
      IF (IB.EQ.0) GO TO 304                                            5540.0
C                                                                       5541.0
      ZVOL(IB) = ZVOL(IB) + V(I,J)                                      5542.0
  304 CONTINUE                                                          5543.0
  305 CONTINUE                                                          5544.0
C                                                                       5545.0
C  *** CALCULATE ZONE AVERAGE FLUX...                                   5546.0
C                                                                       5547.0
      DO 320 I=1,NZONE                                                  5548.0
      IF (ZVOL(I).EQ.0.0) GO TO 315                                     5549.0
C                                                                       5550.0
      DO 310 G=1,NGROUP                                                 5551.0
      ZFLUX(G,I) = ZFLUX(G,I)/ZVOL(I)                                   5552.0
  310 CONTINUE                                                          5553.0
      GO TO 320                                                         5554.0
C                                                                       5555.0
  315 CALL CLEAR (0.0, ZFLUX(1,I), NGROUP)                              5556.0
  320 CONTINUE                                                          5557.0
C                                                                       5558.0
  400 CALL SREED (IFILN,ITEMP,0,0,0,4,JRECF)                            5559.0
C                                                                       5560.0
C***********************************************************************5561.0
C                                                                       5562.0
C     WRITE ZONE FLUX FILE (RZFLUX)                                     5563.0
C                                                                       5564.0
      IDN=27                                                            5565.0
      NWDS=24+NGROUP*NZONE+LEDINF                                       5566.0
C                                                                       5567.0
      CALL OPENWR (IDN, IFILN, JRECF, IVERF, NWDS, 'RZRITE')            5568.0
C                                                                       5569.0
      TIME=0.0                                                          5570.0
      VOL=0.0                                                           5571.0
      EIVS=0.0                                                          5572.0
```

```
      DKDS=0.0                                                          5573.0
      TNL =0.0                                                          5574.0
      TNA =0.0                                                          5575.0
      TNSL=0.0                                                          5576.0
      TNBL=0.0                                                          5577.0
      TNBAL=0.0                                                         5578.0
      TNCRA=0.0                                                         5579.0
      NBLOK=1                                                           5580.0
      ITPS=0                                                            5581.0
      NCY=0                                                             5582.0
      CALL EFBYTE (ISPEC(1), TIME, 1)                                   5583.0
      CALL EFBYTE (ISPEC(2), POWER, 1)                                  5584.0
      CALL EFBYTE (ISPEC(3), VOL, 1)                                    5585.0
      CALL EFBYTE (ISPEC(4), EV, 1)                                     5586.0
      CALL EFBYTE (ISPEC(5), EIVS, 1)                                   5587.0
      CALL EFBYTE (ISPEC(6), DKDS, 1)                                   5588.0
      CALL EFBYTE (ISPEC(7), TNL, 1)                                    5589.0
      CALL EFBYTE (ISPEC(8), TNA, 1)                                    5590.0
      CALL EFBYTE (ISPEC(9), TNSL, 1)                                   5591.0
      CALL EFBYTE (ISPEC(10), TNBL, 1)                                  5592.0
      CALL EFBYTE (ISPEC(11), TNBAL, 1)                                 5593.0
      CALL EFBYTE (ISPEC(12), TNCRA, 1)                                 5594.0
      ISPEC(13)=0                                                       5595.0
      ISPEC(14)=0                                                       5596.0
      ISPEC(15)=0                                                       5597.0
      ISPEC(16)=NBLOK                                                   5598.0
      ISPEC(17)=ITPS                                                    5599.0
      ISPEC(18)=NZONE                                                   5600.0
      ISPEC(19)=NGROUP                                                  5601.0
      ISPEC(20)=NCY                                                     5602.0
C                                                                       5603.0
      CALL SRITE (IFILN,ISPEC,0,20,0,1,JRECF)                           5604.0
C                                                                       5605.0
      CALL SRITE (IFILN,ZFLUX,NGROUP*NZONE,0,0,1,JRECF)                 5606.0
C                                                                       5607.0
C     CLOSE RZFLUX FILE                                                 5608.0
C                                                                       5609.0
      CALL SRITE (IFILN,ITEMP,0,0,0,4,JRECF)                            5610.0
      WRITE (NOUT,700) HOLOUT(1)                                        5611.0
      CALL MESSAG(89)                                                   5612.0
C                                                                       5613.0
  500 CONTINUE                                                          5614.0
      RETURN                                                            5615.0
C                                                                       5616.0
  700 FORMAT (2H */2H */2H *,44X,18H...INTERFACE FILE ,A6,11H WRITTEN...  5617.0
     1 /)                                                               5618.0
```

```
C                                                          5619.0
    END                                                    5620.0
```

위의 코드 리스트에서 우리가 관심 있게 보아야 할 것은 줄 5372.0과 5423.0의 두 줄인데, 먼저 줄 5372.0에서 모든 실변수들을 크기가 8 바이트인 double precision으로 정의한 뒤, 줄 5423.0에서는 실변수 배열 ZFLUX를 정의하고 있다. ZFLUX가 중성자속의 구역 평균값을 저장하는 배열이므로 TWODANT에서는 중성자속을 double precision으로 처리하고 있음을 볼 수 있다. 이와 같은 실변수의 크기에 대해 CCCC에서는 구체적으로 언급하지 않은 채 단지 중성자속을 저장하는 배열을 실변수로 정의하고 있을 뿐이다. 따라서 응용코드마다 중성자속 배열을 single precision 실변수로 정의할 수도 있고, 여기 TWODANT처럼 double precision 실변수로 정의 할 수도 있어서 서로간에 호환성의 문제가 발생하는 것이다.

일반적으로 Binary 파일에 값을 저장할 때, 하나의 변수가 single precision이면 4 바이트를 차지하고 double precision이면 8 바이트를 차지하기 때문에 보통은 CCCC 표준 부함수 RITE에서 저장하는데 필요한 바이트 수를 계산할 때 변수 MULT를 곱해주어 계산한다. 즉, single precision 실변수를 처리할 경우에는 MULT = 1로 값을 지정하고, double precision 실변수를 다루는 경우에는 MULT = 2로 지정하여 사용한다. 이에 비해 TWODANT는 독특하게 자신의 고유한 서브루틴들인 SREED, SRITE를 사용하여 이를 처리한다. 이제 TWODANT가 RZFLUX를 출력하는 방법에 대해 자세하게 살펴보기로 한다.

공간 분포 중성자속 파일 RTFLUX를 읽어서 구역 평균 중성자속을 계산하고 이를 출력하는 과정은 줄 5561.0부터 시작된다. 먼저 부프로그램 OPENWR을 불러서 RZFLUX 파일을 열고 파일 시작부분(file identification)을 쓴다. 여기에서 NWDS에 들어가는 LEDINF는 임시 파일의 크기이므로 우리의 관심인 RZFLUX의 크기와는 무관한 것이다. 줄 5570.0에서 5602.0 까지는 RZFLUX의 1D RECORD에 해당하는 file specification을 지정하는 과정이며 20개의 서로 다른 변수를 하나의 레코드에다 묶기 위해 배열 ISPEC을 사용하였다. 배열 ISPEC의 각 원소들이 지니는 의미는 2장에 있는 RZFLUX 파일 설명에 주어져있다. 이 과정에서 부프로그램 EFBYTE는 double precision으로 정의되어 있

는 실변수들의 크기를 4 바이트로 줄여주는 역할을 한다. 20개의 변수들을 한 배열의 원소들로 묶기 위해서는 그 크기가 일정해야 하므로 실변수들의 크기를 정수 변수들의 크기인 4 바이트로 변환시키는 것이다. 줄 5604.0이 실제로 20개 원소로 구성된 배열 ISPEC을 RZFLUX에 쓰는 문장이다.

CALL SRITE (IFILN, ISPEC, 0, 20, 0, 1, JRECF) 5604.0

여기서, 서브루틴 SRITE는 TWODANT에서 정의한 서브루틴 SREED의 한 ENTRY로서 표준 CCCC 함수인 REED/RITE를 사용하여 데이터를 출력하는 일을 한다. 이 문장에서 사용되는 변수 IFILN은 입출력 파일 번호이며 ISPEC은 저장하려는 변수들이 놓여있는 배열의 이름이다. 0, 20, 0은 각각 double precision 실변수의 개수, single precision 실변수의 개수, 글자변수 개수이고, 1은 저장하려는 레코드가 한 개임을 뜻하며, JRECF는 그 레코드의 번호이다. SRITE는 CCCC 표준함수 RITE를 부르기 위해 필요한 단어(word)수를 계산할 때 single precision 실변수는 1 단어, double precision 실변수는 2 단어, 문자변수는 2 단어로 계산한다. 따라서 이 경우는 20개의 단어가 저장되어 80 바이트의 저장 공간을 차지한다.

줄 5606.0이 중성자속의 구역 평균값들을 저장하는 줄이다.

CALL SRITE (IFILN,ZFLUX,NGROUP*NZONE,0,0,1,JRECF) 5606.0

이 문장에서 ZFLUX는 중성자속의 구역 평균값들을 갖고 있는 배열이며 이 배열이 double precision으로 정의되어 있으므로, 저장되는 변수의 수를 전달할 때, 저장되는 변수의 수, 즉 배열 ZFLUX의 원소의 개수인 NGROUP*NZONE을 double precision 실변수 개수 위치에 놓았다. 같은 레코드에 저장되는 single precision 실변수나 문자 변수가 없으므로 당연히 이들의 개수는 0으로 지정하였다. 뒤에 따라오는 나머지 부분은 RZFLUX를 닫는 단순한 과정이다.

4. TRANSX 개선

4.1 TRANSX의 RZFLUX

이 보고서 작성 시에 사용한 TRANSX는 Los Alamos National Laboratory의 R. E. MacFarlane이 작성한 version 2.15이다. 이 코드의 분석 및 수행 과정 점검은 TRANSX version 2.0에 대한 사용자 설명서[1]를 바탕으로 하여 HP 9000/700 시리즈 J200 워크스테이션에서 HP-UX 10.01 상에서 운용되는 디버깅 도구인 DDE(Distributed Debugging Environment)를 사용하였다.

TRANSX는 CCCC에서 정의한 대로(2장에 있는 RZFLUX 설명이 TRANSX가 사용하는 것임) RZFLUX 파일을 읽어들인다. 그렇지만, 중성자속의 구역 평균값들을 double precision이 아닌 single precision 배열로 읽어들인다. 따라서 중성자속을 double precision으로 저장하는 TWODANT와는 서로 호환이 불가능하다. TRANSX가 파일 RZFLUX를 다루는 방법을 조사하기 위해 여기에 TRANSX가 RZFLUX를 읽어들이는 부분을 발췌하여 포함하였다.

```
      PROGRAM TRANSX                                                    TRANSX.2
C     **********************************************************************  TRANSX.3
C                                                                       TRANSX.4
C     TRANSPORT CROSS SECTIONS                                          TRANSX.5
C     FROM MATXS LIBRARIES                                              TRANSX.6
C     VERS. 2.15 (10 NOV 94)                                            VERS.4
C                                                                       TRANSX.8
C     ---DESCRIPTION-----------------------------------------------     TRANSX.9
C                                                                       TRANSX.10
C    CONVERT CROSS SECTIONS FROM A MATXS FORMAT INTERFACE FILE FOR USE   TRANSX.11
C     IN VARIOUS TRANSPORT CODES.  TRANSX MAY BE USED TO PRODUCE NEUTRON TRANSX.12
C     TABLES, PHOTON TABLES, OR COUPLED SETS.  TABLES CAN BE CONSTRUCTED TRANSX.13
C     IN MATWISE OR GROUPWISE ORDERING AND IN DIRECT OR ADJOINT FORM.   TRANSX.14
C     VARIOUS TRANSPORT CORRECTIONS CAN BE APPLIED IF DESIRED.  FISSION  TRANSX.15
C     NEUTRON PRODUCTION AND SPECTRA CAN BE PROMPT OR INTEGRATED TO     TRANSX.16
```

```
C     INFINITE TIME.  ISOTOPES CAN BE MIXED AND/OR CONVERTED TO MACRO-      TRANSX.17
C     SCOPIC UNITS.  FLEXIBLE EDITS CAN BE CONSTRUCTED THAT ARE ANY         TRANSX.18
C     LINEAR COMBINATION OF VECTOR PARTIALS FROM THE MATXS FILE.  TABLES    TRANSX.19
C     MAY BE COLLAPSED TO A SUBSET GROUP STRUCTURE.  THERMAL COHERENT       TRANSX.20
C     AND INCOHERENT SCATTERING FOR MODERATORS CAN BE USED AT LOW           TRANSX.21
C     ENERGIES WITH UPSCATTER.  SELF-SHIELDING CAN BE DONE FOR HOMO-        TRANSX.22
C     GENEOUS AND SIMPLE HETEROGENEOUS SYSTEMS INCLUDING CELL HOMOGENI-     TRANSX.23
C     ZATION.  VARIOUS CONSTITUENT MIXES OR MICROSCOPIC CROSS SECTIONS      TRANSX.24
C     CAN BE CONSTRUCTED USING THE COMPONENTS OF A MACROSCOPIC MIX.  THE    TRANSX.25
C     FINE GROUP FLUXES CAN BE READ FROM AN INTERFACE FILE OR CARDS, OR     TRANSX.26
C     THE LIBRARY WEIGHT FUNCTION CAN BE USED AS THE FLUX.  WHEN A FLUX     TRANSX.27
C     IS AVAILABLE, ELASTIC SCATTERING MAY BE CORRECTED FOR THE             TRANSX.28
C     DEVIATIONS FROM THE NOMINAL MODEL FLUX.  THE SCATTERING MATRIX        TRANSX.29
C     MAY BE MODIFIED TO BE ANY LINEAR COMBINATION OF THE PARTIAL           TRANSX.30
C     MATRICES FOR SENSITIVITY STUDIES.  OUTPUT CAN BE OBTAINED             TRANSX.31
C     IN SIMPLE CARD FORMATS, FIDO FORMAT, ISOTXS FORMAT, OR THE BINARY     TRANSX.32
C     GROUP-ORDERED GOXS FORMAT.  TRANSX USES PAGING, VARIABLE DIMEN-       TRANSX.33
C     SIONING, AND MULTIPLE PASSES THROUGH THE MATXS FILE TO HANDLE         TRANSX.34
C     LARGE JOBS.  THE MATXS LIBRARY CAN BE EITHER A FILE, OR A             UP10.4
C     DIRECTORY CONTAINING SINGLE-MATERIAL FILES FOR RANDOM ACCESS.         UP10.5
C                                                                           TRANSX.36
C     ---USER INPUT (XFREE FORM)-----------------------------------------   TRANSX.37
C     ***READ FLUX FROM INTERFACE FILE (IF ANY)                             TRANSX.771
                              .
                              .
                              .
      NWDS=NFINE*NRFL                                                       TRANSX.781
      IF (INITF.EQ.2) THEN                                                  UP11.8
        IF (MOD(LC,2).EQ.1) LC=LC+1                                         UP11.9
        CALL REED(NFLX,JREC,A(LC+1),NWDS*MULT,0)                            UP11.10
      ELSE                                                                  UP11.11
        CALL REED(NFLX,JREC,A(LC+1),NWDS,0)                                 UP11.12
      ENDIF                                                                 UP11.13
                              .
                              .
                              .
      IF (INITF.EQ.4) LCX=LC+JJ-1+NFINE*(IRFL-1)                            TRANSX.814
      LFX=LFF+JJ+NFINE*(IREG-1)                                             TRANSX.815
      IF (ICOLL.NE.0) LFY=LFLUX+JG+NGROUP*(IMT-1)                           TRANSX.816
                              .
                              .
                              .
      IF (INITF.EQ.2) THEN                                                  UP11.38
```

```
      A(LFX)=HA((LCX+JL*MULT)/MULT)*A(LRVOL+IREG)                    UP11.39
    ELSE                                                             UP11.40
      A(LFX)=A(LCX+JL)*A(LRVOL+IREG)                                 UP11.41
    ENDIF                                                            UP11.42
    LFX=LFX-NFINE*NREG                                               TRANSX.824
                                    .
                                    .
                                    .
    IF (INITF.EQ.2) THEN                                             UP11.48
      A(LFY)=A(LFY)+HA((LCX+JL*MULT)/MULT)*A(LRVOL+IREG)             UP11.49
    ELSE                                                             UP11.50
      A(LFY)=A(LFY)+A(LCX+JL)*A(LRVOL+IREG)                          UP11.51
    ENDIF                                                            UP11.52
    LFY=LFY+NGROUP*NMIX                                              TRANSX.829
                                    .
                                    .
                                    .
C                                                                    TRANSX.840
C ***COMPUTE SIGMA-ZERO VALUES AND SELF-SHIELDING FACTORS            TRANSX.841
                                    .
                                    .
                                    .
```

위 코드를 이해하기 위해 중요한 변수들의 의미를 보이면 다음과 같다.

NFINE : Number of fine groups

NRFL : Number of regions(zones)

INITF : Type of initial flux -- 0 : No input flux
1 : Card input flux
2 : RTFLUX
3 : RZMFLX
4 : RZFLUX

A : An array contains all data treated in TRANSX(Equivalenced to IA and HA)

LC + 1 : Array index of the first input flux read from RZFLUX

LFX : Array index of input flux for each group

LFY : Array index of input flux summed over fine energy groups at each zone

줄 TRANSX.781에서 줄 UP11.13까지가 RZFLUX를 읽어들이는 부분이다. 여기서 UP11.13은 LANL에서 코드를 수정(Update)할 때 그 것을 표시하기 위해 사용하는 표현법으로 코드가 나중에 수정된 부분임을 뜻한다. 이 부분에서는 입력 중성자속 파일이 RTFLUX(or INITF=2)인 경우와 RZFLUX(or INITF=4)인 경우를 구분하여 전자의 경우에는 double precision인 경우에 대비하고 있다. 다시 말해서, single precision 실변수가 1 단어이고 double precision 실변수가 2 단어인 시스템에서는 보통 6 바이트인 문자변수가 2 단어로 처리되며, 이 경우 MULT의 값이 2가 되어, RTFLUX를 다룰 때는 중성자속을 double precision으로 읽어들인다. 이에 반해 후자의 경우에는 NWDS에 MULT를 곱하지 않으므로 RZFLUX를 다룰 때 읽혀지는 단어의 수는 그대로 NWDS이며 이 것은 중성자속을 single precision으로 다룰 때의 값이다. 이 부분을 보면 CCCC에서 중성자속을 단순하게 실변수로 정의하고 그 크기를 분명하게 지적하지 않아서 여러 가지 혼동이 생기고 있음을 짐작할 수 있다. 즉 이 부분은 TRANSX가 변천하는 과정에서 수정된 부분으로(줄의 끝 부분이 UP**.** 이다.) 처음부터 TRANSX가 double precision인 RTFLUX에 대비하지 못하고 나중에 이 부분이 덧붙여진 것임을 알 수 있다.

4.2 TRANSX 수정

앞장에서 살펴 본 바와 같이 TWODANT는 중성자속을 double precision으로 저장하므로 위 4.1절에 주어져 있는 TRANSX 원시 코드를 다음과 같이 고쳐서 RTFLUX 뿐 아니라 RZFLUX도 double precision인 경우에 대비하게 한다.

```
NWDS=NFINE*NRFL                                  TRANSX.781
IF (MOD(LC,2).EQ.1) LC=LC+1                      KAERI1.3
CALL REED(NFLX,JREC,A(LC+1),NWDS*MULT,0)         KAERI1.4
```

이제 하나의 구역 평균된 중성자속을 읽어올 때, 한 원소의 크기(MULT)를 2 단어로 하여 처리하므로 double precision으로 저장되어 있는 RZFLUX를 정확하게 읽어온다.

전체 코드를 정확하게 수행하기 위해서는 일단 읽어들인 중성자속을 이용하여 다른 계산을 하는 부분에서도 제대로 double precision으로 처리해 주어야만 한다. TRANSX는 다른 일반적인 코드들과는 달리 각 중성자속이나 단면적과 같은 물리적 양들을 독립적인 배열로 처리하지 않고 하나의 거대한 배열 A를 정의하여 사용한다. 이 배열 A는 포트란의 EQUIVALENCE문을 사용하여 배열 IA와 HA에 동등한 배열로 선언되어 있어서 실변수뿐만 아니라 정수 변수와 문자 변수도 모두 하나의 배열의 일부로 취급한다. 각 물리량들은 이 커다란 배열 안에서 각각 부분 배열로 처리되며 그 시작하는 위치와 크기는 아주 조심해서 계산되고 있다. 예를 들어, 위에서 구역 평균된 중성자속을 저장하는 부분 배열은 LC+1에서 시작한다. 이런 방법은 컴퓨터 메모리를 절약하고 관리하는데 편리한 이점이 있지만 제작자 이외 다른 사람이 코드를 분석하고 수정하는 것을 매우 어렵게 한다. 즉, 하나의 부분 배열이 바뀌면 그 영향이 배열 내의 모든 부분 배열에 미치게 되므로 극히 조심하여 이를 다루어야만 한다.

TRANSX가 사용하는 거대 배열 A는 single precision 배열이므로 이제 double precision으로 읽혀진 중성자속을 다룰 때 한 번에 두 인자(index)씩 건너뛰어야 한다. TRANSX는 군축약을 할 때 사용하는 비중함수로 RZFLUX 뿐 아니라 여러 형식의 중성자속을 허용하여 그 형식에 따라 다른 부분 배열로 읽어 들인 뒤에 이 것을 하나의 비중함수 배열로 정의하여 사용하기 때문에 구역 평균 중성자속을 다시 정렬하여 비중함수에 대응시키는 정렬 과정이 필요하다. TRANSX가 다른 부분에서 사용하기 위하여 구역 평균 중성자속을 정렬시키는 부분이 줄 TRANSX.814에서 TRANSX.816 부분이다. 이 부분은 구역 평균 중성자속을 저장할 새로운 위치 즉 거대 배열 A에서 구역 평균 중성자속 부분 배열이 시작하는 새로운 인자수를 결정한다. double precision으로 저장된 중성자속을 처리하기 위해 줄 TRANSX.814를 다음과 같이 변경한다. 여기서는 중

성자속의 한 저장인자와 다음 중성자속의 저장인자가 단어 단위로 MULT 만큼 떨어져있게 된다.

```
IF (INITF.EQ.4) LCX=LC+(JJ-1+NFINE*(IRFL-1))*MULT    KAERI1.5
```

이제 배열 A에 저장되어있는 중성자속을 불러올 때도 MULT가 2인 경우를 고려하여 한번에 2 단어씩 건너뛰도록 조정해야한다. TRANSX에서는 이를 위해 double precision인 경우 배열 A 대신 배열 HA를 사용하고 있다. HA는 문자 변수의 배열이므로 하나의 원소가 4*MULT 바이트를 차지한다. MULT가 1이 아닌 경우에는 거의 모두 2이므로 편의상 이제부터는 MULT가 2라 하고 진행한다. 배열 A와 HA는 동등한 배열이다. 다시 말해서 EQUIVALENCE문으로 선언되어 있다. 하지만 A는 single precision 실변수의 배열로서 한 원소가 4 바이트를 차지하지만, HA는 문자 변수의 배열로서 한 원소가 8 바이트를 차지한다. 그러므로 HA(1)의 시작 번지는 A(1)과 같지만, HA(2)의 시작 번지는 A(2)가 아니라 A(3)과 같다. 따라서 double precision으로 정의된 중성자속을 읽어오기 위해 HA를 사용하는 경우에는 그 인자 번호를 배열 A의 인자 번호의 반으로 해야한다. 이에 해당하는 부분이 줄 UP11.38에서 TRANSX.829까지이므로 이 부분을 다음과 같이 고친다.

```
A(LFX)=HA((LCX+JL*MULT)/MULT)*A(LRVOL+IREG)          KAERI1.6
LFX=LFX+NFINE*NREG                                   TRANSX.824
.
.
A(LFY)=A(LFY)+HA((LCX+JL*MULT)/MULT)*A(LRVOL+IREG)   KAERI1.7
LFY=LFY+NGROUP*NMIX                                  TRANSX.829
.
```

위의 수정된 코드에서는 HA의 인자를 계산할 때 배열 A에서 중성자속이 존

재하는 인자 번호(LCX + JL*MULT)를 다시 MULT로 나누어서 올바른 인자를 얻고있다. 이 계산에 나타나는 모든 변수(LCX, JL, MULT)는 정수 변수들이므로 나눗셈 또한 정수의 나누기로 행하여짐을 주목해야한다. 즉, 나누기의 결과 나타나는 소수점이하의 부분은 모두 제거되고 단순히 정수 값만 취하게 되므로 HA의 인자번호는 모두 정수로만 주어진다.

앞 4.1 절에 주어진 고치기 전의 TRANSX 리스트와 비교할 수 있도록 새로운 TRANSX 리스트에서 위에 수정한 내용을 정리하여 아래에 보였다.

```
      PROGRAM TRANSX                                                    TRANSX.2
C     ********************************************************************* TRANSX.3
C                                                                       TRANSX.4
C     TRANSPORT CROSS SECTIONS                                          TRANSX.5
C     FROM MATXS LIBRARIES                                              TRANSX.6
C     VERS. 2.15 (10 NOV 94)                                            VERS.4
CK              Last Updated on 7 June 1996                             KAERI1.1
CK                   ----  RZFLUX in double precision is allowed.      KAERI1.2
C                                                                       TRANSX.8
C     ---DESCRIPTION-----------------------------------------------     TRANSX.9
C                                                                       TRANSX.10
C    CONVERT CROSS SECTIONS FROM A MATXS FORMAT INTERFACE FILE FOR USE  TRANSX.11
C     IN VARIOUS TRANSPORT CODES.  TRANSX MAY BE USED TO PRODUCE NEUTRON TRANSX.12
C     TABLES, PHOTON TABLES, OR COUPLED SETS.  TABLES CAN BE CONSTRUCTED TRANSX.13
C     IN MATWISE OR GROUPWISE ORDERING AND IN DIRECT OR ADJOINT FORM.   TRANSX.14
C     VARIOUS TRANSPORT CORRECTIONS CAN BE APPLIED IF DESIRED.  FISSION TRANSX.15
C     NEUTRON PRODUCTION AND SPECTRA CAN BE PROMPT OR INTEGRATED TO     TRANSX.16
C     INFINITE TIME.  ISOTOPES CAN BE MIXED AND/OR CONVERTED TO MACRO-  TRANSX.17
C     SCOPIC UNITS.  FLEXIBLE EDITS CAN BE CONSTRUCTED THAT ARE ANY     TRANSX.18
C     LINEAR COMBINATION OF VECTOR PARTIALS FROM THE MATXS FILE.  TABLES TRANSX.19
C     MAY BE COLLAPSED TO A SUBSET GROUP STRUCTURE.  THERMAL COHERENT   TRANSX.20
C     AND INCOHERENT SCATTERING FOR MODERATORS CAN BE USED AT LOW       TRANSX.21
C     ENERGIES WITH UPSCATTER.  SELF-SHIELDING CAN BE DONE FOR HOMO-    TRANSX.22
C     GENEOUS AND SIMPLE HETEROGENEOUS SYSTEMS INCLUDING CELL HOMOGENI- TRANSX.23
C     ZATION. VARIOUS CONSTITUENT MIXES OR MICROSCOPIC CROSS SECTIONS   TRANSX.24
C     CAN BE CONSTRUCTED USING THE COMPONENTS OF A MACROSCOPIC MIX. THE TRANSX.25
C     FINE GROUP FLUXES CAN BE READ FROM AN INTERFACE FILE OR CARDS, OR TRANSX.26
C     THE LIBRARY WEIGHT FUNCTION CAN BE USED AS THE FLUX.  WHEN A FLUX TRANSX.27
C     IS AVAILABLE, ELASTIC SCATTERING MAY BE CORRECTED FOR THE         TRANSX.28
C     DEVIATIONS FROM THE NOMINAL MODEL FLUX.  THE SCATTERING MATRIX    TRANSX.29
C     MAY BE MODIFIED TO BE ANY LINEAR COMBINATION OF THE PARTIAL       TRANSX.30
```

```
C     MATRICES FOR SENSITIVITY STUDIES.  OUTPUT CAN BE OBTAINED                TRANSX.31
C     IN SIMPLE CARD FORMATS, FIDO FORMAT, ISOTXS FORMAT, OR THE BINARY        TRANSX.32
C     GROUP-ORDERED GOXS FORMAT.  TRANSX USES PAGING, VARIABLE DIMEN-          TRANSX.33
C     SIONING, AND MULTIPLE PASSES THROUGH THE MATXS FILE TO HANDLE            TRANSX.34
C     LARGE JOBS.  THE MATXS LIBRARY CAN BE EITHER A FILE, OR A                UP10.4
C     DIRECTORY CONTAINING SINGLE-MATERIAL FILES FOR RANDOM ACCESS.            UP10.5
C                                                                              TRANSX.36
C     ---USER INPUT (XFREE FORM)-----------------------------------------      TRANSX.37
C     ***READ FLUX FROM INTERFACE FILE (IF ANY)                                TRANSX.771
                                      .
                                      .
                                      .
      NWDS=NFINE*NRFL                                                         TRANSX.781
      IF (MOD(LC,2).EQ.1) LC=LC+1                                             KAERI1.3
      CALL REED(NFLX,JREC,A(LC+1),NWDS*MULT,0)                                KAERI1.4
                                      .
                                      .
                                      .
      IF (INITF.EQ.4) LCX=LC+(JJ-1+NFINE*(IRFL-1))*MULT                       KAERI1.5
      LFX=LFF+JJ+NFINE*(IREG-1)                                               TRANSX.815
      IF (ICOLL.NE.0) LFY=LFLUX+JG+NGROUP*(IMT-1)                             TRANSX.816
                                      .
                                      .
                                      .
      A(LFX)=HA((LCX+JL*MULT)/MULT)*A(LRVOL+IREG)                             KAERI1.6
      LFX=LFX+NFINE*NREG                                                      TRANSX.824
                                      .
                                      .
                                      .
      A(LFY)=A(LFY)+HA((LCX+JL*MULT)/MULT)*A(LRVOL+IREG)                      KAERI1.7
     -LFY=LFY+NGROUP*NMIX                                                     TRANSX.829
                                      .
                                      .
                                      .
C                                                                             TRANSX.840
C ***COMPUTE SIGMA-ZERO VALUES AND SELF-SHIELDING FACTORS                     TRANSX.841
                                      .
                                      .
                                      .
```

5. 개선 결과 검토

여기서는 개선된 TRANSX를 사용하여 만들어진 RZFLUX 파일이 TWODANT와 호환되는가를 검토한다. 먼저 간단한 모형을 정한 뒤에 TRANSX를 수행하여 TWODANT의 입력으로 사용되는 단면적 파일을 얻는다. 이 단면적 파일 외에 TRANSX계산에 사용된 모형과 같은 모형이 되도록 TWODANT의 입력파일을 준비하여 k_{eff} 계산을 수행하면서 RZFLUX파일을 만든다. 끝으로 다시 한번 TRANSX를 사용하여 군축약을 하는 데, 이 때 위 RZFLUX 파일을 읽어들여서 비중함수로 이용한다. 한편, 본 보고서의 작성자가 만든 프로그램을 이용하여 기계어 파일인 RZFLUX를 단순한 문자 파일로 변형한 뒤 그 내용과 두 번째 TRANSX 수행과정에서 읽어들인 RZFLUX의 내용을 비교하여 두 중성자속이 일치하는지 확인한다.

5.1 전산코드 READRZ

먼저 TRANSX가 RZFLUX 파일을 잘 읽어들이는 것을 확인하기 위하여 TWODANT가 작성한 RZFLUX를 읽어서 ASCII Code로 보여주는 전산코드를 작성한다. 전산코드 READRZ는 포트란 소스 코드 readrz.f를 휴렛 패커드의 f77을 이용하여 컴파일한 실행 가능 코드로써, RZFLUX를 읽어서 그 내용을 출력파일 ARZF에 쓴다. 출력파일 ARZF는 단순 문자 파일이며 그 형식은 RZFLUX의 내용을 쉽게 보여주기 위해 각 구역(Zone)에 대해 군별 중성자속을 순서대로 나열한 형식을 취하였다. 따라서 ARZF는 CCCC의 어떤 규약도 따르지 않으며 ARZF에서 반대로 RZFLUX를 만들 수는 없다. readrz.f의 내용은 길지 않으므로 본 보고서의 이해를 돕고 비슷한 일을 하는 경우에 도움이 되도록 하기 위해 그 내용을 여기에 포함하였다.

```
CDECK READRZ
C   A program to read the CCCC standard RZFLUX file and produce
C  corresponding BCD format file named ARZF. The output file ARZF
C  is neither a standard CCCC file nor available as an input file instead
C  of RZFLUX.
C                                                    1996, 4, 9
C                                                    Hyun-Tai Chung
C                                                    KAERI
C
      Program READRZ
C
C -- Subroutines called by READRZ ------------------------------------
C
C    CHRFLT  Routine to transfer real variables from a real array
C              to a character array
C    ERROR   Error message routine
C    REED    CCCC routine to read binary files
C    SEEK    Returns file reference number
C    SEKPHL  Correspondence between file refernce number and LUN.
C
C -- Files referenced in READRZ ----------------------------------------
C     FILE NAME   LOGICAL UNIT NUMBER   RECORD POINTER
C     =========   ======================   ================
C       RZFLUX            NRZFLX
C       ARZF              NARZF
C
      implicit double precision (A-H, O-Z)
      character*8 dsname(3), hrzflx, arzf
      character*8 hname(3)
      integer*4  nrzflx, narzf, iarzf, nref(3)
      integer*4  iver, mult, irec, nwds
      integer*4  ivers
      real*8  fname(4)
      real*4  farray(20)
      real*4  time, power, vol, effk, eivs, dkds, tnl
      real*4  tna, tnsl, tnbl, tnbal, tncra, x(3)
      integer*4  iarray(20)
      integer*4  nblok, itps, nzone, ngroup, ncy
      integer*4  i, j, jl, ju, k, m
```

```
      real*8  zgf(50000)

      common/IOPUT/nin, nout, nout2

      equivalence (fname(4),ivers)
      equivalence (iarray, farray)
      data dsname/'RZFLUX', 'ARZF', '$'/
      data hrzflx/'RZFLUX'/, arzf/'ARZF'/
      data nref/11, 12, 0/
      data i0/0/, i1/1/, i3/3/

C  Set parameters and initialize SEEK table.

      nin   = 5
      nout  = 6
      nout2 = 0
      iver  = 1
      mult  = 2
      irec  = 0

      call SEEK(dsname, i0, nref, i3)

      call SEEK(hrzflx, iver, nrzflx, i1)
      if (nrzflx .le. i0) go to 100

      call SEEK(arzf, iver, iarzf, i1)
      call SEKPHL(iarzf, narzf, i0)
      if (narzf .le. i0) go to 100

C  Read file identification

      nwds = 3*mult + 1
      irec = irec + 1
      call REED(nrzflx, irec, fname, nwds, i0)
      call CHRFLT(hname, fname, 3)
      write(narzf, 1000) (hname(i), i=1,3), ivers

C  Read specifications  (1D record)
```

```
      irec = irec + 1
      nwds = 20
      call REED(nrzflx, irec, iarray, nwds, i0)

C         Following variable assignments are dummy routines to
C         show the name of each element of farray or iarray.

      time   = farray(1)
      power  = farray(2)
      vol    = farray(3)
      effk   = farray(4)
      eivs   = farray(5)
      dkds   = farray(6)
      tnl    = farray(7)
      tna    = farray(8)
      tnsl   = farray(9)
      tnbl   = farray(10)
      tnbal  = farray(11)
      tncra  = farray(12)
      x(1)   = farray(13)
      x(2)   = farray(14)
      x(3)   = farray(15)
      nblok  = iarray(16)
      itps   = iarray(17)
      nzone  = iarray(18)
      ngroup= iarray(19)
      ncy    = iarray(20)
      write(narzf,1010) (farray(i), i=1,15)
      write(narzf,1020) (iarray(i), i=16,20)

C  Read flux values

      if ( nblok .ge. 1 ) then
         index = 1
         do 10 m=1, nblok
            jl = (m-1)*((nzone-1)/nblok+1) + 1
            ju = MIN0(nzone, m*((nzone-1)/nblok+1))
            nwds = mult*ngroup*(ju-jl+1)
```

```
            irec = irec + 1
            call REED(nrzflx, irec, zgf(index), nwds, i0)
            index = index + nwds
 10       continue
          do 30 j=1, nzone
            write(narzf,*) '                ZONE NUMBER  : ', j
            write(narzf,*) 'Zone averaged flux for each group'
            do 20 k=1, ngroup
              write(narzf,1030) k, zgf((j-1)*ngroup+k)
 20         continue
 30       continue
       else
          call ERROR('READRZ',-100)
          write(narzf,*) 'nblok = ', nblok, ' is less than 1.'
       endif

 100 continue

 1000 format(A6, ' *', 2A6, '*', I6)
 1010 format(1P5E12.5)
 1020 format(5I12)
 1030 format('Group : ', I6, '   ', 1PE12.5)

       STOP
       END
```

5.2 READRZ와 TRANSX에서 읽은 RZFLUX의 비교

수정한 TRANSX가 올바르게 동작하는지를 확인하기 위하여 TWODANT가 작성한 RZFLUX파일을 READRZ와 TRANSX가 각각 읽은 내용을 비교한다. 이 때 사용되는 RZFLUX를 얻는 과정부터 보면 다음과 같다.

5.2.1 TRANSX 수행

먼저 다음과 같이 TRANSX를 실행시켜 단면적 파일을 만든다.

transx < ik_trans > ok_trans

이 때 입력 파일인 ik_trans는 TWODANT 설명서[2]에 있는 보기 입력 파일을 기준으로 만든 것으로 그 내용은 아래와 같다.

```
RTFLUX Check file with sample of TWODANT - level 1.
0 5 0 1 2 1 0 3 0 0
80 1 84 0 0 5 2 11 1 0
FUEL  SODIUM  STEEL   SODIU2   STEEL2
CORE   300  1.27170E+05  1  20/
REFLEC 300  3.497175E+05 1  15/
 1     1  PU239   0.0103/
 1     1  U238    0.0103/
 1     1  O16     0.0412/
 2     1  NA23    0.025/
 3     1  FE56    0.05/
 3     1  CR52    0.016/
 3     1  NI58    0.01/
 4     2  NA23    0.025/
 5     2  FE56    0.05/
 5     2  CR52    0.016/
 5     2  NI58    0.01/
 CHI/
STOP
```

우리의 목적은 RZFLUX 파일의 호환성에 관한 것이며 TRANSX의 사용 방법에 관한 것이 아니므로 이 입력파일에서 사용하는 노심 모형과 입력 변수들의 의미는 TRANSX 사용자설명서[1]와 TWODANT 사용자설명서[2]를 참조하도록 하고 여기서는 이에 대한 설명을 생략한다. 단지 이 과정에서 만들어지는 단면적 파일의 이름은 GOXS(Group Ordered Cross Section) 인데, 이 파일의 형식은 LANL에서 자체적으로 정의하여 사용하는 것으로 같은 디렉토리에서 MACRXS와 SNXEDT라는 이름으로 링크되어 있어서 TWODANT가 바로 읽어들일 수 있다. TRANSX가 사용하는 단면적 자료 파일인 MATXS 파일은 80×24 그룹으로 이루어진 MATXS11[1]을 사용하였다. 이 파일은 ENDF/B-VI

를 기초로 하여 액체금속로 계산을 위해 작성되었으며 80개의 중성자 그룹과 24개의 광자 그룹이 결합된 형식을 가지고 있다.

5.2.2 TWODANT 수행

앞 과정 5.2.1에서 만든 단면적 파일 GOXS를 이용하여 주어진 모형의 k_{eff} 값을 계산하는 과정으로서 다음과 같은 명령으로 실행시킨다.

twodant < ik_two > ok_two

TWODANT를 수행한 결과로 k_{eff} 값을 얻는 것과 동시에 RTFLUX, RZFLUX 등의 여러 가지 중성자속 파일을 만들 수 있다.
이 때 사용되는 입력 파일 ik_two는 TWODANT 사용자 설명서[2]에 있는 보기 입력 파일에서 물질 배합 방법만을 약간 바꾼 것으로, 물질의 구성 성분 및 비율은 똑 같다. 참고로 그 내용을 아래에 포함하였다.

```
   1
 Same geometry with Sample input of TWODANT
/ Geometry        - R,Z
/ Cross sections  -  80 group, isotropic scatter from MATXS11
/ Mixing          -  Sodium, Fuel, Steel are mixed in TRANSX
/ Zone            -  Materials are assigned to make zone
/                    named CORE and REFLEC
/ Solver          -  Standard k calculation
/ Edits           -  As detail as possible
/
/ ------------------------ Block  I ------------------------
   IGEOM= R-Z NGROUP=80  ISN=4   NISO=5  MT=5  NZONE=2
   IM=2  IT=25  JM=3  JT=30
   T
/ --------------------- Block II (Geomtery) -----------------
   XMESH= 0, 30, 45        XINTS= 15, 10
   YMESH= 0, 15, 60, 75    YINTS= 5,  20, 5
   ZONES= 2R2;   1,2;   2R2
   T
/ -------------------- Block III (Cross Sections) -----------
```

```
    LIB= MACRXS
    T
/ ------------------- Block IV (Mixing) -------------------
    MATLS= ISOS
    ASSIGN= CORE     FUEL   0.35   SODIUM 0.4   STEEL 0.25;
            REFLEC   SODIU2 0.7    STEEL2 0.3
    T
/ ------------------- Block V (Solver) --------------------
    IEVT=1  ISCT=0  IBR=0  IBT=0  IBB=0
    NORM=1  FLUXP=2  XSECTP=2  FISSRP=1  BALP=3
    T
/ ------------------- Block VI (Edits) --------------------
    PTED=1 ZNED=1
    EDISOS=1  RZFLUX=1
    T
```

5.2.3 READRZ 수행

위 5.2.2의 결과로 만들어진 RZFLUX파일을 readrz를 써서 단순 문자 파일인 ARZF를 만든다. ARZF의 내용은 다음과 같다.

```
            ZONE NUMBER  :  1
 Zone averaged flux for each group
Group :     1   6.45488E-10
Group :     2   2.21139E-09
Group :     3   6.34901E-09
Group :     4   2.27152E-08
Group :     5   1.00644E-07
Group :     6   5.72321E-07
Group :     7   1.81774E-06
Group :     8   4.19766E-06
Group :     9   7.33811E-06
Group :    10   1.12809E-05
Group :    11   1.67584E-05
Group :    12   1.79383E-05
Group :    13   2.13372E-05
Group :    14   1.12709E-05
```

Group :	15	1.11711E-05
Group :	16	9.59947E-06
Group :	17	1.24171E-05
Group :	18	1.45183E-05
Group :	19	1.51500E-05
Group :	20	1.65941E-05
Group :	21	1.69641E-05
Group :	22	1.26804E-05
Group :	23	9.49385E-06
Group :	24	2.67536E-05
Group :	25	2.50592E-05
Group :	26	2.22426E-05
Group :	27	1.88230E-05
Group :	28	1.91740E-05
Group :	29	1.51633E-05
Group :	30	1.30842E-05
Group :	31	9.65827E-06
Group :	32	8.94061E-06
Group :	33	6.83631E-06
Group :	34	2.20599E-06
Group :	35	1.77707E-06
Group :	36	1.93842E-06
Group :	37	3.22944E-06
Group :	38	2.71568E-06
Group :	39	2.18154E-06
Group :	40	1.44948E-06
Group :	41	2.28100E-06
Group :	42	1.85040E-06
Group :	43	1.57265E-06
Group :	44	1.34161E-06
Group :	45	1.13553E-06
Group :	46	9.76547E-07
Group :	47	8.33867E-07
Group :	48	7.00467E-07
Group :	49	5.79446E-07
Group :	50	4.61595E-07
Group :	51	3.43171E-07
Group :	52	2.16759E-07
Group :	53	9.02495E-08

Group : 54 3.44580E-08
Group : 55 1.23129E-07
Group : 56 2.50189E-07
Group : 57 3.09119E-07
Group : 58 3.14986E-07
Group : 59 2.89529E-07
Group : 60 2.48571E-07
Group : 61 2.03066E-07
Group : 62 1.67935E-07
Group : 63 2.58939E-07
Group : 64 1.85012E-07
Group : 65 1.04616E-07
Group : 66 7.49643E-08
Group : 67 4.75019E-08
Group : 68 4.46658E-08
Group : 69 1.69899E-08
Group : 70 4.80125E-09
Group : 71 1.42588E-09
Group : 72 1.19137E-09
Group : 73 3.94289E-10
Group : 74 2.06379E-10
Group : 75 9.27098E-11
Group : 76 8.68804E-11
Group : 77 5.80701E-11
Group : 78 5.41935E-12
Group : 79 6.53724E-14
Group : 80 2.23390E-14

ZONE NUMBER : 2

Zone averaged flux for each group

Group : 1 8.08889E-11
Group : 2 2.51759E-10
Group : 3 7.71261E-10
Group : 4 2.74383E-09
Group : 5 1.15650E-08
Group : 6 6.65436E-08
Group : 7 2.09045E-07
Group : 8 4.91795E-07

Group :	9	8.89789E-07
Group :	10	1.44696E-06
Group :	11	2.44778E-06
Group :	12	3.00865E-06
Group :	13	3.98084E-06
Group :	14	2.21033E-06
Group :	15	2.26820E-06
Group :	16	2.12165E-06
Group :	17	2.35626E-06
Group :	18	2.48522E-06
Group :	19	2.90925E-06
Group :	20	3.83011E-06
Group :	21	4.52096E-06
Group :	22	3.59167E-06
Group :	23	2.67018E-06
Group :	24	6.91920E-06
Group :	25	5.91144E-06
Group :	26	5.28549E-06
Group :	27	4.64496E-06
Group :	28	5.39586E-06
Group :	29	4.02916E-06
Group :	30	3.60991E-06
Group :	31	2.39909E-06
Group :	32	2.53208E-06
Group :	33	1.88738E-06
Group :	34	5.09733E-07
Group :	35	4.31927E-07
Group :	36	6.05218E-07
Group :	37	1.01104E-06
Group :	38	8.20921E-07
Group :	39	6.35528E-07
Group :	40	3.81558E-07
Group :	41	7.38788E-07
Group :	42	6.33155E-07
Group :	43	5.27751E-07
Group :	44	4.48235E-07
Group :	45	3.83197E-07
Group :	46	3.29593E-07
Group :	47	2.81712E-07

Group :	48	2.38001E-07
Group :	49	1.95830E-07
Group :	50	1.51969E-07
Group :	51	1.06658E-07
Group :	52	6.08930E-08
Group :	53	2.22658E-08
Group :	54	7.95415E-09
Group :	55	3.23120E-08
Group :	56	8.53175E-08
Group :	57	1.36603E-07
Group :	58	1.66043E-07
Group :	59	1.71586E-07
Group :	60	1.62802E-07
Group :	61	1.39590E-07
Group :	62	1.24834E-07
Group :	63	2.07156E-07
Group :	64	1.55588E-07
Group :	65	1.12836E-07
Group :	66	8.27133E-08
Group :	67	6.23923E-08
Group :	68	8.06436E-08
Group :	69	3.61780E-08
Group :	70	1.59154E-08
Group :	71	4.90001E-09
Group :	72	2.84684E-09
Group :	73	1.55642E-09
Group :	74	8.45912E-10
Group :	75	4.48046E-10
Group :	76	2.43530E-10
Group :	77	1.69874E-10
Group :	78	4.89131E-11
Group :	79	1.18072E-11
Group :	80	2.53671E-12

5.2.4 TRANSX를 이용한 군축약

다음과 같은 명령으로 5.2.2절에서 만들어진 RZFLUX를 읽어서 80 그룹을 8

그룹으로 축약하는 작업을 수행한다.

transx < ik_trans.rzf > ok_trans.rzf

그러면 출력 파일인 ok_trans.rzf에 TRANSX가 RZFLUX에서 읽은 중성자속이 포함되어 있다. 다만 여기에 포함된 중성자속은 RZFLUX에서 읽은 중성자속에 해당 구역의 부피를 곱한 값이므로 두 경우를 비교하기 위해서는 ARZF에 나타난 값에 해당 구역의 부피를 곱해야 한다. 아래에 ok_trans.rzf에서 TRANSX가 RZFLUX에서 구역 평균 중성자속을 읽어서 출력한 부분을 나타내었다.

```
REGION        TEMP(K)       SIZE    HETEROGENIETY
------        -------       ----    -------------
  1  CORE     3.000E+02     1.272E+05    1  2.000E+01
  2  REFLEC   3.000E+02     3.497E+05    1  1.500E+01

 FINE FLUX FOR REGION  1    SUM  5.1826E+01
  8.209E-05 2.812E-04 8.074E-04 2.889E-03 1.280E-02 7.278E-02 2.312E-01 5.338E-01 9.332E-01 1.435E+00
  2.131E+00 2.281E+00 2.713E+00 1.433E+00 1.421E+00 1.221E+00 1.579E+00 1.846E+00 1.927E+00 2.110E+00
  2.157E+00 1.613E+00 1.207E+00 3.402E+00 3.187E+00 2.829E+00 2.394E+00 2.438E+00 1.928E+00 1.664E+00
  1.228E+00 1.137E+00 8.694E-01 2.805E-01 2.260E-01 2.465E-01 4.107E-01 3.454E-01 2.774E-01 1.843E-01
  2.901E-01 2.353E-01 2.000E-01 1.706E-01 1.444E-01 1.242E-01 1.060E-01 8.908E-02 7.369E-02 5.870E-02
  4.364E-02 2.757E-02 1.148E-02 4.382E-03 1.566E-02 3.182E-02 3.931E-02 4.006E-02 3.682E-02 3.161E-02
  2.582E-02 2.136E-02 3.293E-02 2.353E-02 1.330E-02 9.533E-03 6.041E-03 5.680E-03 2.161E-03 6.106E-04
  1.813E-04 1.515E-04 5.014E-05 2.625E-05 1.179E-05 1.105E-05 7.385E-06 6.892E-07 8.313E-09 2.841E-09

 FINE FLUX FOR REGION  2    SUM  3.3028E+01
  2.829E-05 8.804E-05 2.697E-04 9.596E-04 4.044E-03 2.327E-02 7.311E-02 1.720E-01 3.112E-01 5.060E-01
  8.560E-01 1.052E+00 1.392E+00 7.730E-01 7.932E-01 7.420E-01 8.240E-01 8.691E-01 1.017E+00 1.339E+00
  1.581E+00 1.256E+00 9.338E-01 2.420E+00 2.067E+00 1.848E+00 1.624E+00 1.887E+00 1.409E+00 1.262E+00
  8.390E-01 8.855E-01 6.600E-01 1.783E-01 1.511E-01 2.117E-01 3.536E-01 2.871E-01 2.223E-01 1.334E-01
  2.584E-01 2.214E-01 1.846E-01 1.568E-01 1.340E-01 1.153E-01 9.852E-02 8.323E-02 6.849E-02 5.315E-02
  3.730E-02 2.130E-02 7.787E-03 2.782E-03 1.130E-02 2.984E-02 4.777E-02 5.807E-02 6.001E-02 5.693E-02
  4.882E-02 4.366E-02 7.245E-02 5.441E-02 3.946E-02 2.893E-02 2.182E-02 2.820E-02 1.265E-02 5.566E-03
  1.714E-03 9.956E-04 5.443E-04 2.958E-04 1.567E-04 8.517E-05 5.941E-05 1.711E-05 4.129E-06 8.871E-07
```

위 출력 파일에서 구역은 region으로 정의되어 있으며, 구역의 부피는 region의 size에 주어져 있다. 즉, 위 파일에는 CORE와 REFLEC 두 개의 구역이 있고

그 부피는 각각 1.27170×10^{5} cm^{3}과 3.497175×10^{5} cm^{3}이다.

이제 두 파일에 있는 중성자속들을 비교해보자. 먼저 구역(zone) 1의 첫 번째 군을 본다. ARZF에 있는 중성자속에 구역 1의 부피를 곱하면 다음과 같다.

$$(6.45488\times10^{-10})\times(1.27170\times10^{5}) = (8.20867\times10^{-5})$$

이것은 ok_trans.rzf에 있는 중성자속 8.209E-05와 일치한다. 구역 1의 56 번째 군을 비교해 보자. 이 결과도 ok_trans.rzf의 3.182E-02와 정확하게 일치한다.

$$(2.50189\times10^{-7})\times(1.27170\times10^{5}) = (3.18165\times10^{-2})$$

끝으로 구역 2의 77 번째 군을 비교해본다.

$$(1.69874\times10^{-10})\times(3.497175\times10^{5}) = (5.94079\times10^{-5})$$

역시 TRANSX가 double precision으로 쓰여진 RZFLUX를 정확하게 읽고 있다.

5.3 RTFLUX와 RZFLUX를 사용한 군축약

군축약할 때 비중함수로 RTFLUX를 쓴 경우와 RZFLUX를 사용한 경우의 k_{eff}값에 생기는 변화를 관찰하기 위하여 하나의 보기 프로그램을 수행하였다. 먼저 TRANSX에 대해 군축약을 하지 않는 보기 입력파일을 작성하여 MATXS파일에서 GOXS 파일을 만든다. 이 GOXS 파일을 사용하여 TWODANT를 수행시켜서 RTFLUX와 RZFLUX를 얻는다. 그 다음 각각의 중성자속 파일을 이용하여 TRANSX로 군축약을 하고 축약된 GOXS 파일을 사용하여 다시 한번 TWODANT를 수행시켜 얻어진 k_{eff} 값을 군축약하기 전에 계산한 값과 비교한다.

5.3.1 군축약을 위한 중성자속 파일의 형성

먼저 군축약을 하기 전에 TRANSX와 TWODANT를 사용하여 주어진 노심 모형에 대한 k_{eff}를 계산한다. 이 때 단면적 파일은 소내 핵자료 평가체제 구축 분야에서 작성한 50개의 중성자군을 가진 MATXS 파일[4]을 단면적 자료 파일로 사용하였다. TRANSX와 TWODANT를 사용하여 RTFLUX와 RZFLUX 파

일을 만들고 k_{eff} 값을 계산하는 과정은 다음과 같다.

i) transx < ib_trans > ob_trans

ii) ln -s GOXS MACRXS

iii) ln -s GOXS SNXEDT

iv) twodant < ib_two > ob_two

과정 i)에서는 입력 ib_trans를 사용하여 MATXS에서 TWODANT계산에 필요한 단면적 파일 GOXS를 만든다. 입력 파일 ib_trans는 다음과 같으며, 이것은 CORE와 BLANKT로 이루어진 단순한 구형 임계노심 Big Ten에 대한 계산이다.

```
KAERI FAST SET TEST --- BIG TEN
0 5 0 1 2 1 0 3 0 0
50 4 60 0 0 2 2 7 7 7
CORE BLANKT
CORE 300 1.1861333E5 1 40.64/
BLANKT 300 2.81706E5 -1/
1 1 U234  0.00005/
1 1 U235  0.00484/
1 1 U238  0.04268/
1 1 U233  1.E-20/
1 1 NP237 1.E-20/
1 1 PU239 1.E-20/
2 2 U238  0.04797/
CHI F233 F235 F238 F237 F239 C238/
1 CHI/
2 NFTOT 4.84E17  U233/
3 NFTOT 1        U235/
4 NFTOT 0.1134020619 U238/
5 NFTOT 4.84E17 NP237/
6 NFTOT 4.84E17 PU239/
7 NG    0.1134020619 U238/
STOP
```

과정 ii)와 iii)에서는 각각 i)의 결과로 만들어진 단면적 파일 GOXS를

TWODANT가 요구하는 MACRXS와 SNXEDT로 이름을 바꾸어 연결하는 작업을 한다.

과정 iv)에서와 같이 TWODANT를 실행하여 우리가 필요로 하는 RTFLUX와 RZFLUX를 만든다. 이 때 사용한 입력 파일 ib_two의 내용은 아래에 주어져 있으며 TWODANT를 한 번 수행하여 동시에 RTFLUX와 RZFLUX를 생성하도록 입력 파일을 만들었다. 군축약을 하기 전에 50그룹을 사용하여 계산한 k_{eff} 값은 ob_two에 있으며 1.0123이다.

```
    1
    BIG TEN ** CORE/REFLECTOR SPHERE ** ONEDANT-50G
/ BLK I
    DIMENS= 3 50 16 2 2 2 2 100 T
/ BLK II
    XMESH= 0. 30.48 45.72 XINTS= 50, 50
    ZONES= 1 2 T
/ BLK III
    LIB= MACRXS
    MAXORD=3 IHM=60 IHT=10 IHS=11 IFIDO=0 ITITL=0
    T
/ BLK IV
    MATLS=ISOS
    ASSIGN=MATLS T
/ BLK V
    SOLIN= 1 3 0 0 0 EPSI=1.E-5 NORM=1.0  SOLOUT=1 0 0 0 0 0 0 1 T
/ BLK VI
    PTED=1 ZNED=1 EDXS=5,6,7,8,9,10
    EDISOS=1 RZFLUX=1
    MICSUM=1,0,5,0,1,0,6,0,1,0,7,0,1,0,8,0,1,0,9,0,1,0,10 T
```

5.3.2 RTFLUX를 사용한 군축약

RTFLUX를 비중함수로 이용하여 50 개 군으로 이루어진 MATXS에서 6개의 군으로 군축약한 단면적 파일을 얻고 다시 이 것을 사용하여 k_{eff} 값을 계산하는 과정은 다음과 같다.

i) transx < ib_trans.rtf > ob_trans.rtf

ii) twodant < ib_two.rtf > ob_two.rtf

먼저 TRANSX를 실행하기 위해 입력 파일을 준비한다. 적당한 편집기로 다음과 같이 5.3.1절에서 사용한 입력 파일 ib_trans의 처음 두 줄을 고치고 맨 뒤에 한 줄을 추가한다.

```
0 5 0 1 2 1 0 3 0 0   --->  0 5 0 1 2 1 0 3 50 2
0 1 12 0 0 5 2 11 1 0 --->  6 1 12 0 0 5 2 11 1 0
맨 뒤에 추가된 줄     :      5  2  10  6  12  15
```

여기에서 첫 번 줄의 맨 뒤에 있는 두 변수는 축약 전의 그룹수(50)와 축약할 때 사용하는 비중함수가 RTFLUX임을 표시한다(2). 두 번째 줄의 처음 변수는 새로이 축약된 그룹의 수(6)를 나타내고 맨 뒤에 추가된 줄은 6 그룹에 각각 포함될 전 그룹들의 개수이다.

이제 i)에서 TRANSX를 실행하여 새로운 GOXS 파일을 얻는다. 새로 만들어진 GOXS파일은 RTFLUX를 사용하여 6개의 군으로 축약한 단면적 자료 파일이 된다.

과정 ii)를 수행하기 위해 5.3.1절에서 사용한 TWODANT의 입력파일 ib_two를 복사하여 새로운 입력 파일 ib_two.rtf를 준비하고 편집기로 다음과 같이 수정한다.

```
DIMENS= 3 50 16 2 2 2 2 100 T   -->  DIMENS= 3 6 16 2 2 2 2 100 T
IHM=60                          -->  IHM=16
```

첫 번째 줄에서는 단면적 자료 파일의 수를 50에서 6으로 줄였으며, 두 번째 줄에서는 단면적을 저장하는 배열의 차원을 60에서 16으로 줄였다. 이제 과정 ii)와 같이 TWODANT를 사용하여 k_{eff}값을 얻는다. 이 경우에는 1.0192이며 이는 군축약 전(1.0123)에 비해 0.68% 큰 값이다.

5.3.3 RZFLUX를 사용한 군축약

이 작업은 기본적으로 5.3.2절과 같은 과정으로 진행되며 단지 TRANSX의 입력 파일에서 RTFLUX 대신 RZFLUX를 비중함수로 사용하도록 바꾸어주기만 하면 된다.

i) transx < ib_trans.rtf > ob_trans.rzf

ii) twodant < ib_two.rtf > ob_two.rzf

먼저 ib_trans.rtf를 ib_trans.rzf로 복사한 뒤 다음과 같이 한 줄을 고친다.

0 5 0 1 2 1 0 3 50 2 ---> 0 5 0 1 2 1 0 3 50 4

여기서 맨 끝 변수를 2에서 4로 고쳐서 RTFLUX대신 RZFLUX를 비중함수로 사용하도록 한 다음 i)에서 TRANSX를 실행하여 새로운 GOXS 파일을 얻는다.

ii)에서는 ib_two.rtf를 ib_two.rzf로 복사하여 TWODANT의 입력 파일로 사용한다. 두 경우 입력 변수가 모두 같으므로 수정할 필요는 없다. 이제 TWODANT를 사용하여 k_{eff}값을 얻는다. 이 경우에는 1.0176이며 축약 전에 비해 0.53% 크지만 RTFLUX를 사용한 경우보다는 0.16% 만큼 축약전의 값에 가깝다. 따라서 잘 조정된 군축약 구조를 사용할 경우 RTFLUX보다 RZFLUX를 비중함수로 하여 군축약을 하는 것이 군축약 전의 값에 보다 가까운 결과를 준다.

이와 같이 TRANSX에서 작성한 GOXS 파일을 TWODANT에서 바로 읽어들일 수 있고, 역으로 TWODANT에서 작성한 RZFLUX 파일을 TRANSX가 사용할 수 있음을 확인하였다. 따라서, 단면적 라이브러리 MATXS에서 TWODANT를 사용하기 위한 단면적 자료 파일 생성과정, TWODANT를 사용한 구역 평균 중성자속 파일 RZFLUX의 생성과정, 그리고 이 RZFLUX를 이용하여 군축약된 새로운 단면적 자료 파일 생성과정으로 이어지는 순환 과정이 완성되었다고 할 수 있다.

6. 결 론

국내 액체금속로 노심설계시스템인 K-CORE 시스템에서 TRANSX는 다군 단면적 라이브러리 MATXS를 사용하여 확산이론 계산코드나 수송이론 계산코드와 같은 중성자속 계산코드가 요구하는 입력 파일들을 작성하는 도구이며, 이 과정에서 필요한 경우 군축약을 할 수 있다. K-CORE system내에서의 유효단면적 생성 및 처리 모듈 체계 전산코드인 TRANSX v2.15는 군축약에 사용하는 비중함수로 double precision으로 쓰여진 구역 평균 중성자속 파일 RZFLUX를 처리하지 못하고 있었다. 본 보고서에서는 TRANSX가 가지고 있던 이와 같은 단점을 제거함으로써 double precision으로 저장되어 있는 구역 평균 중성자속 파일인 RZFLUX를 처리할 수 있도록 하였다. 결과적으로 평가 핵자료 파일인 MATXS, 단면적 생성파일 TRANSX, 수송이론 계산코드인 TWODANT들 사이의 순환 연계 계산이 가능하게 되었으며 이를 보기 모형 계산으로 확인하였다.

본 보고서에서 구축한 유효단면적 생성 및 처리 모듈 체계는 노심설계 계산 절차의 관점에서 매우 중요하다. 또한 다른 계산 과정, 즉 중성자속/연소 계산 모듈과의 연계체제 수립으로 국내 액체금속로 노심핵설계 계산체제의 기본 연계체제를 구축하게 되었으며, 국내 개발 액체금속로 KALIMER의 노심설계 개념개발 및 성능분석의 효율적인 수행이 기대된다.

참고 문헌

1. R. E. MacFarlane, "TRANSX 2: A Code for Interfacing MATXS Cross Section Libraries to Nuclear Transport Codes", Los Alamos National Laboratory, LA-12312-MS (December 1993).

2. R. Douglas O'Dell, Forrest W. Brinkley Jr., Duane R. Marr and Raymond E. Alcouffe, "Revised User's Manual for ONEDANT: A Code Package for One-Dimensional, Diffusion-Accelerated, Neutral-Particle Transport", Los Alamos National Laboratory, LA-9184-M (December 1989).

3. 조만 외 17인, "고속증식로 기본기술 개발연구(한·불 공동연구) 별책부록", KAERI/RR-1010-1/90, 한국원자력연구소 (1990).

4. 김정도, 길충섭, "핵자료 처리 및 다군 단면적 생산", KAERI/RR-1599/95, 한국원자력연구소 (1996).

서 지 정 보 양 식			
수행기관 보고서 번호	위탁기관 보고서 번호	표준 보고서 번호	INIS 주제코드
KAERI/TR-745/96			

제목/부제

TRANSX 코드의 군축약 기능 개선

연구책임자 및 부서명	김 영 철, 액체금속로 요소기술 개발

연구자 및 부서명 김 영 인, 김 영 균, 정 현 태

발 행 지	대 전	발 행 기 관	한국원자력연구소	발 행 일	1996년7월
페 이 지	51 p.	도 표	유(○) 무( )	크 기	26 cm
참고사항					
비밀여부	공개(○) 대외비( ) 급 비밀		보고서 종류	기술 보고서	
연 구 위 탁 기 관			계약번호		

초록(300단어 내외)

한국원자력연구소 액체금속로 노심설계기술개발 분야에서 구축중인 노심설계 계산체제 K-CORE 시스템의 유효단면적 생성 및 처리 모듈체계의 구성 전산코드 TRANSX Version 2.15는 단면적 라이브러리 MATXS를 사용하여 중성자속 계산모듈들이 요구하는 단면적 입력파일들을 생산하는 전산코드이다. TRANSX는 K-CORE 시스템에서 군축약에 사용하는 비중함수로 double precision으로 쓰여진 구역 평균 중성자속 파일 RZFLUX를 처리하는 기능이 없다. 따라서, 본 보고서에서는 TRANSX를 이용한 RZFLUX 처리 기능을 개발, 보강함으로써 K-CORE 시스템의 군축약 기능을 개선하였다. 이로써, 단면적 라이브러리 파일인 MATXS, 유효단면적 생성 및 처리 전산코드 TRANSX, 수송 이론 계산코드인 TWODANT로 이어지는 계산체제에 RZFLUX를 이용한 순환 연계 계산이 가능하도록 하였으며, 보기 모형 계산을 통하여 이를 확인하였다.

주제명 키워드 (10단어 내외) 액체금속로 노심설계, 단면적 라이브러리, 군축약, MATXS, TRANSX, RZFLUX

BIBLIOGRAPHIC INFORMATION SHEET			
Performing Org. Report No.	Sponsoring Org. Report No.	Standard Report No.	INIS Subject Code
KAERI/TR-745/96			

Title/Subtitle	Improvement of Group Collapsing in TRANSX Code
Project Manager and Dept.	Y.C. Kim, Development of LMR Design Technology
Researcher and Dept.	Y.I. Kim, Y.G. Kim, H.T. Chung

Pub. Place	Taejon	Pub. Org.	KAERI	Pub. Date	1996. 7
Page	51 p.	Fig. & Tab.	Yes(○) No( )	Size	26 cm
Note					
Classified	Open(○) Outside( ) Class		Report Type		
Sponsoring Org.			Contract No.		

Abstract (About 300 Words) A cross section generating and processing computer code TRANSX version 2.15 in the K-CORE system, being developed by the KAERI LMR core design technology development team produces various cross section input files appropriate for flux calculation options from the cross section library MATXS. In this report, a group collapsing function of TRANSX has been improved to utilize the zone averaged flux file RZFLUX written in double precision as flux weighting functions. As a result, an iterative calculation system using double precision RZFLUX consisting of the cross section data library file MATXS, the effective cross section producing and processing code TRANSX, and the transport theory calculation code TWODANT has been set up and verified through a sample model calculation.

Subject Keywords (About 10 Words) LMR Core Design, Cross Section Library, Group Collapsing, MATXS, TRANSX, RZFLUX