연구보고서
화학 92-2-21

화학공장 계측, 제어 계통의 신뢰성 평가 및 고장원인 탐색기법

- 화학설비(반응기, 저장조) 고장분석 -

1992. 12. 31

한국산업안전공단
KOREA INDUSTRIAL SAFETY CORPORATION
산업안전연구원
INDUSTRIAL SAFETY RESEARCH INSTITUTE

제 출 문

한국산업안전공단 이사장 귀하

본 보고서를 산업재해 예방기술의 연구개발 및 보급사업의 일환으로 수행한 "화학공장 계측·제어 계통의 신뢰성 평가 및 고장원인 탐색기법"의 최종보고서 [화학설비(반응기, 저장조) 고장분석]로서 제출합니다.

1992년 12월 31일

주관연구부서 : 산업안전연구원
화 학 연 구 실
연구책임자 : 실장 정 동 인
연구수행자 : 주 종 대

목　　　차

제1장 서 론

1. 연구목적

화학공장 계측제어 계통의 신뢰성평가 및 고장원인 탐색기법에 관한 연구로서 1차년도('91년)에는 계측, 제어계통의 안전성 평가기법에 관하여 화학공장에서 사용되는 각종 계측, 제어 기기류에 대한 총괄적인 고장율과 고장원인 등을 조사, 보고하였다. 동연구 과제의 2차년도 ('92) 사업으로서 화학공장의 반응기 및 대규모 위험물 저장 탱크에 대하여 이들 설비에서의 계측, 제어 계통의 안전도 분석기법 및 기초자료에 대하여 연구하였다.

이들 화학설비중 반응기는 화학공장의 안전과 관련하여 각종 사고의 위험성이 매우 높으며, 위험물 저장탱크의 경우는 사고 발생의 위험은 상대적으로 낮으나 일단 사고가 발생하면 대규모 화재·폭발을 야기시켜 그 피해 범위가 크게되므로 이들 두 종류의 화학설비를 대상으로 각종 계측, 제어기기 및 안전장치류의 고장율을 조사 분석하여 안전성 평가에 활용함으로서 화학공장의 안전조업 및 재해예방에 기여하고자 한다.

2. 연구기간 및 범위

1차년도 : 1991. 1~1991. 12 : 계측, 제어계통의 안전성 평가기법

2차년도 : 1992. 1~1992. 12 : 화학설비(반응기, 저장조) 고장 분석

3. 연구내용

가. 계측, 제어계통의 안전성 평가방법〔1차년도('91년) 완료〕

국내 계측, 제어기기의 사용실태 및 신뢰성(고장율)조사를 실시하여 이에 대한

자료를 확보하고 계측, 제어기기별 주요 고장원인과 고장율 데이타를 활용함으로서 결함수 해석법(FTA)에 의한 안전성 평가방법에 이용할 수 있도록 재해확률 등의 계산용 컴퓨터 프로그램을 개발하였다.

나. 화학설비 고장원인 탐색기법〔2차년도('92) 과제〕

화학공장의 주요설비중 화재·폭발 등 재해발생빈도가 가장 높은 반응기와 재해 규모가 큰 위험물 저장설비에 대하여 이들 설비에서의 계측, 제어기기가 나타내는 고장 발생 형태 및 고장율을 조사함으로서 단위 화학설비에 대한 안전성 평가에 필요한 기초 자료를 확보하고 1차년도 개발한 FTA계산용 프로그램을 수정 보완하여 누구나 쉽게 응용하도록 하였다.

제2장 정 비 이 론

본 연구의 1차 보고서에서는 신뢰도의 정의와 종류 및 고장율에 대한 수학적 개념을 요약 정리하였다. 본 보고서에서는 사업장에서 설비 및 장치의 유지·보전을 위한 각종 정비작업시 고려하여야 할 사항과 정책등 정비이론에 대하여 정리하고자 한다.

1. 예방정비

가. 정비 효율과 비용

공장의 설비나 부품은 노후, 마모 등의 자연적 원인이나 사용조건, 방법 등에 따라 고장을 일으키게 되는데 이때 설비의 교환 또는 수리시에 비용이 소요될 뿐만 아니라 생산차질로 인한 손실비용 등이 발생한다. 따라서 고장의 가능성을 최소로 감소시킴으로서 설비를 일정 수준으로 유지, 보전할 필요가 있으며 이를 위해 최적의 정비효율과 그 비용을 고려하지 않으면 안된다.

즉, 정비 효율과 여기에 소요되는 비용은 서로 상관관계가 존재하는데 높은 정비효율을 얻기 위해서는 비용의 부담이 증가하게 되므로 이와 같은 두가지 측면을 동시에 고려하여 최적의 방법을 결정하여야 좋은 정비결과를 기대할 수 있게 되는 것이다.

따라서 정비에 관련된 주요결정사항과 그 특성 및 요소를 들어보면 다음과 같다.

(1) 예방정비와 고장정비

(2) 정비방법의 선정 및 인력조달

(3) 장비의 수리 또는 교환

(4) 외부위탁시 위탁방법(수시 또는 기간)

(5) 부품의 재고관리

(6) 작업관리

나. 예방정비와 고장정비

일반적으로 사업장에서 수행하고 있는 정비활동은 예방정비(계획정비)와 고장정비(비계획정비)로 분류하고 있으며 이들에 대한 특성은 다음과 같다.

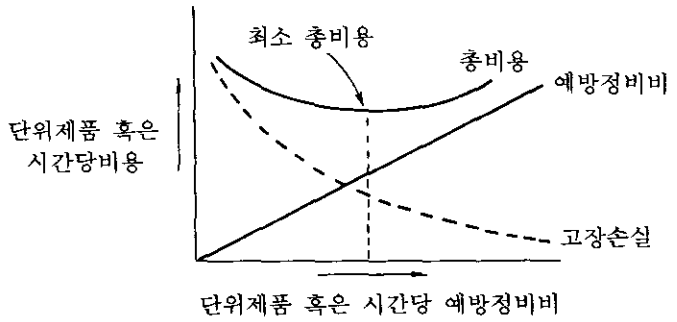
(1) 예방(계획) 정비

일정한 시간간격을 가지고 수행하는 정비형태로서 시스템 또는 부품의 상태를 의도하는 수준의 성능과 신뢰도 및 안전성을 유지하는데 목적이 있다. 예방(계획)정비는 이와같은 목적을 달성시키기 위하여 일상적인 점검, 검사 및 분해, 수리 등을 행하는데 관련 부품의 물리적 특성에 따라 정상적으로 작동하는 하부 시스템과 부품에 대하여 규칙적인 점검을 실시하고, 중요 시스템에 설치되어 있는 중복 대기 부품의 결함유무를 정기적으로 검사하거나 분해, 수리하는 정비행위를 일컫는다.

(2) 고장(비계획) 정비

시스템 또는 부품의 정상적 운영을 방해하는 원인을 찾아서 이것을 교환, 수리 또는 조정함으로서 가능한 한 빠른시간내에 시스템을 정상으로 회복시켜 주는데 목적이 있다. 그러나 고장(비계획)정비의 경우는 고장부품의 예측이 불가능하기 때문에 이에 대하여 즉각적이고 효율적인 대처방법이 매우 어렵고 이로 인하여 생산등에 파급효과가 크며 경우에 따라서는 안전상에 중대한 문제가 발생하기도 한다.

이와같이 예방 및 고장정비는 각각 그 특성과 이에 따른 장단점을 가지고 있기 때문에 어느 한쪽에 치우치는 정비계획을 작성할 수는 없으며 다음 [그림 2-1]과 같이 그 소요비용을 고려하여 두가지 방법을 병용해야 할 것이다.



〔그림 2-1〕 예방정비 대 고장정비

즉 예방정비의 비용곡선은 거의 직선에 가까운 형태로 나타나지만 고장정비의 비용곡선은 예방정비 비용 증가에 따라 급격히 감소하다가 차츰 점근선의 형태가 된다. 고장정비 비용은 고장발생시의 비용과 고장확률의 곱으로서 산출가능하지만 이들은 사실상 예방정비 수행의 수준에 따라 결정되므로 정량적으로 표시하기란 단순하지 않다.

또한 고장발생을 감소시킬 의도로 예방정비의 운영을 지나치게 넓은 범위로 확대한다면 그 소요비용은 정비를 하지 않으므로서 발생하는 비용을 훨씬 상회할 수가 있으므로 경영자는 고장비용과 예방정비 비용의 균형을 신중히 고려하여 결정하여야 할 것이다.

다. 정비방법의 선정 및 인력조달

(1) 정비방법의 선정

정비방법은 그 내용과 부서에 따라 다음과 같이 세가지로 분류할 수 있다.

(가) 자체정비

장비(설비)를 보유하고 있는 운용현장에서 자체요원에 의해 정비하는 것을 말한다. 이 경우는 장비를 작동 및 사용가능토록 시스템에 대한 간단한 정비로서 정기점검, 육안검사, 급유, 조정, 부속의 교환 등의 제한된 작업이 대부분이다.

(나) 전문정비

이 경우는 이동식, 반이동식 장비나 또는 고정설비를 이용하여 전문화된 조직에 의해 수행되는 것으로서 보통 모듈(Module), 어셈브리(Assembly) 혹은 부속품을 해체하여 세밀히 수리하거나 교환되는 것을 말한다. 이때의 정비조직은 정비에 사용되는 장비나 도구를 충분히 갖추고 숙련된 전문 정비 인력이 배치되어 있어 광범위한 수리가 가능하다.

(다) 창(廠)정비

이는 최고 수준의 정비 형태로서 전문정비 과정에서 수행할 수 없는 고난도의 정비를 수행하는 것을 말한다. 창정비는 수 많은 시스템과 장비를 지원하는 전문화된 수리시설이나 또는 장비 제작회사의 부대정비공장의 형태를 취하고 있으며 이 부분에는 고도로 숙련된 전문요원이 집중배치되어 있다.

(2) 인력조달

정비에 따른 비용문제로서 자체 작업수행시에는 작업인원에 대한 인건비가 매우 큰 비중을 갖는다. 즉 사용시간당 인건비는 노동이용율(Labor utilization)에 따라 변하기 때문에 적정 인원의 유지는 정비 비용 최적화에 반드시 필요한 것이다. 최적 정비 비용 산출을 위하여 고려해야 할 사항은 다음과 같다.

(가) 전일제 정비인원의 직접노임

(나) 전일제 노동력의 간접노임

(다) 정비 부속품의 재고유지비용

(라) 가동중단에 따른 손실(정비로 인한 지연 횟수 및 지연기간)

라. 장비의 수리 또는 교환

사용중인 장비가 고장이 발생된 경우 이것을 수리하여 계속 사용하거나 또는 신품으로 교환하여야 할 것이다. 이때의 판단기준은 공업경제학적 대안평가기법을 이용하는 것이 보다 정확하고 경제적인 것으로서 장비투자의 감가상각비, 운

영비 및 간접경비 등을 포함하여 연평균 비용에 기초를 두고 의사결정을 하여야 한다.

바. 위탁방법

자체인력에 의한 정비를 할 수 없는 비파괴검사 등의 전문정비인 경우 외부 전문기관에 위탁하는 경우도 많다. 이때에도 경제성 검토는 반드시 필요하며 의사결정에 영향을 주는 요소로는 기간중에 발생하는 예상 정비 횟수, 정비건수당 비용, 위탁정비효과 등이 있으며 이를 고려하여 기간 계약정비 혹은 수시계약정비 등의 방법을 택하여야 할 것이다. 이 두가지 위탁방법에 따른 비용은 다음과 같다.

(1) 기간계약 정비시 비용 = 추가 예방정비의 비용 + 기간당 계약 비용+계약기간 동안의 고장 확률

(2) 수시계약 정비시 비용 = 예방정비 비용 + 고장확률

2. 부품교환

가동중인 장비에서의 고장발생은 생산의 중단 및 장비파괴 등을 초래하기도 하고 나아가 인명의 피해도 우려되는 등 많은 손실을 야기시킬 수가 있으므로 예방정비 차원에서 노후 부품의 교환은 필수적이라 할 수 있다.

대개 부품교환은 사용기간에 따르는 수명교환(Age replacement) 원칙에 의해 수행되며 이 방법하에서는 부품의 고장, 또는 고장이 나지 않았다 하더라도 설치 후 t시간이 경과했을 때는 부품을 교환하여 준다. 이 방법외에도 일정시점에서 정기적으로 교환해주는 정기교환(Periodic replacement) 또는 일제교환(Block replacement) 방법이 있다.

일정시간 [o, t] 사이에 발생하는 부품 고장 횟수를 $Nf(t)$라 하고, 이 기간중의 예방정비 횟수를 $Np(t)$라 할때 [o, t] 사이에서 발생하는 기대비용은

$$C(t) = cf \cdot E[Nf(t)] + cp \cdot E[Np(t)] \quad \cdots\cdots [2.1]$$

이며 이 기간이 오랜시간이라고 가정할 때 단위시간당 발생하는 평균비용은 다음과 같다.

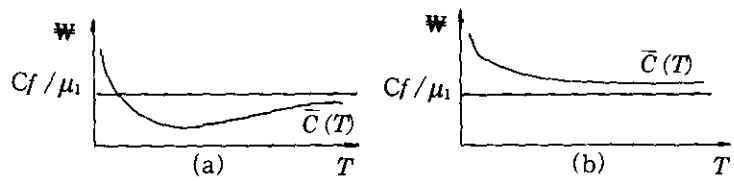
$$\overline{C} = \lim_{t=\infty} \frac{C(t)}{t} \quad \cdots\cdots [2.2]$$

가. 수명교환

이 방법은 부품이 고장나거나 교환수명 T에 이르면 부품을 교환하여 주는 것으로서 평균비용 $\overline{C}$를 최소화하는 T의 최적치 T*는 그 부품의 고장밀도 함수 f(t), 평균수명 $\overline{\mu}(T)$등의 식을 이용하면 다음과 같이 된다.

$$D(t) = h(T)\int_0^T \overline{F}(t)dt - F(T) = \frac{Cp}{Cf - Cp} \quad \cdots\cdots [2.3]$$

이때 h(t)가 연속증가함수인 경우 최적교환수명 T*은 계산이 가능하지만 F(t)가 연속 분포 함수인 경우는 T*=∞가 되어 고장난 부품만 교환한것이 최적의 방법이 되며 이를 도표로 나타내면 다음과 같다.



〔그림 2-2〕 수명교환 방법에서의 $\overline{C}$(T)의 형태

최적 교환수명 T*가 존재할 때의 평균비용 $\overline{C}(T^*)$는 다음과 같다.

$$\overline{C}(T^*) = (cf - cp)\ h(T^*) \quad \cdots\cdots [2.4]$$

나. 일제교환

일제교환 혹은 정기교환이라함은 특정한 종류의 부품전부를 일정시점 t=kT에

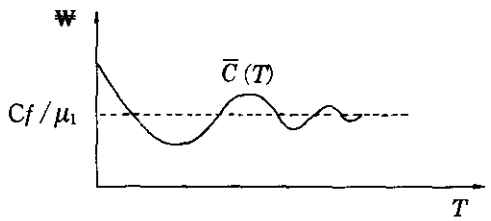
서 동시에 교환하여 주는 것으로서 수명교환과 다른점은 각각의 부품사용기간을 기록할 필요가 없으며 사용기간중 고장발생으로 중간 교환이 이루어진 부품에 대해서도 정기교환때 동시에 교환하여 주는 것으로서 컴퓨터나 아주 복잡한 전자장비등에 대하여 흔히 적용된다. 이 방법은 물자의 낭비는 증가하지만 증가고장율 함수를 갖는 부품의 경우 사용중 발생하는 고장빈도는 적어지게 된다.

이 방법을 이용할때 교환기간은 T, 고장분포함수를 F(t), 평균고장횟수를 재생함수 M(T)라 하면 단위시간당 발생하는 평균비용 $\overline{C}(T)$ 및 최적 T*에 대한 평균비용 $\overline{C}(T^*)$는 다음과 같이 표시할 수 있다.

$$\overline{C}(T)=\frac{c_f M(T)+c_p}{T} \quad \cdots\cdots [2.5]$$

$$\overline{C}(T^*) = c_f \cdot m(T^*) \quad \cdots\cdots [2.6]$$

또한 F(T)가 연속이면 $\overline{c}(T)$도 연속으로서 다음과 같이 감쇄진동을 하며 $T \rightarrow o$일때 $\overline{c}(o) \rightarrow \infty$이고 이때 $T=\infty$라는 것은 고장난 부품만 교환하여 주는 경우가 된다.



〔그림 2-3〕 일제교환 방법에서의 $\overline{C}(T)$의 형태

3. 고장진단

화학공장의 계측, 제어기기는 시스템의 구조가 매우 복잡하며 따라서 여기에 수반되는 유지, 관리작업에 관한 문제가 점점 심각해지고 있다. 이와같은 어려움을 극복하기 위하여 전문적인 진단 기술자들을 훈련, 양성하고 있으나 시스템이 복잡다양화 될수록 이들 장비의 고장원인을 모두 진단할 수 있게끔 전문기술자를

훈련시킨다는 것은 점차 더욱 힘들어 지고 있다. 즉 전문성을 보유하기 위하여는 장시간의 교육, 훈련이 필요하며 변화하는 기술에 적응하기 위한 재교육의 투자 및 전달교육의 한계 등의 문제점이 있는 것이다.

따라서 최근에는 개개인을 전문기술자로 훈련시키는 방법보다 필요한 정보를 컴퓨터기술과 접목시키는 이른바 전문가 시스템을 활용함으로서 누구나가 쉽게 이용이 가능하고 보다 전문적이고 객관적인 고장진단이 가능하게 되었다.

예를들어 주어진 시스템에서 고장을 일으킬 수 있는 고장원과, 정상 혹은 비정상 여부를 관측할 수 있는 지표에 대한 조사분석과정을 거쳐, 특정한 고장원인에 대하여 발생되는 지표의 상호배제적인 조합을 증상이라 함으로서 이증상이 나타나게 되면 이를 기본자료로 하여 고장원인을 분석하는 방법이 있다.

제 3 장 회분식 반응기 사용실태

1. 개 요

가. 조사목적

본 조사의 목적은 화학공장에서 가장 핵심적인 기능을 갖는 반응기에 대하여 운전상태, 조건 및 주변설비의 상황을 분석하고 이를 토대로 안전대책을 연구하는데 기초자료로 활용하고자 함에 있다.

나. 조사대상

조사대상 화학설비를 보유, 운영하는 사업장으로서 원료 취급 및 제조공정상 비교적 위험도가 높은 250개소의 사업장을 대상으로 하였다.

다. 조사방법 및 내용

조사대상 사업장에 설문지를 발송하여 회신을 받은 112개 업체중에서 분석이 가능한 93개 사업장에 대하여 다음 사항을 통계 분석을 하였다.

(1) 반응기의 운전 형태

(2) 반응의 종류

(3) 주변설비의 종류 및 설치현황

(4) 주요계측, 제어 및 안전장치 설비현황

2. 회분식 반응기 사용 현황

가. 조사사업장 현황

본 설문조사에 응한 사업장의 업종별 분포를 살펴보면 〈표 3-1〉과 같다. 사업

장 특성상 석유화학 등 연속공정을 채택하고 있는 업체를 포함시킨 것은 이들 사업장 내에 부분적인 단위공정에서 회분식 반응을 채택하고 있는 경우도 있기 때문이었으나 실제 설문응답은 한개 업체만으로 나타났다.

〈표 3-1〉 설문조사 사업장현황

구분 \ 업종		석유화학	정밀화학	섬유, 고분자	의약	페인트 안료	무기화학	식품	기타	계
총괄	사업장수(개소)	21	19	15	6	9	9	7	7	93
	구성비(%)	22.6	20.4	16.1	6.5	9.7	9.7	7.5	7.5	100
회분식	사업장수(개소)	1	18	12	6	9	7	7	7	66
	구성비(%)	1.5	27.3	18.2	9.1	13.6	10.6	10.6	10.6	100
연속식	사업장수(개소)	20	2	3	-	-	2	-	-	27
	구성비(%)	74.1	7.4	11.1	-	-	7.4	-	-	100

위와같이 석유화학을 제외한 거의 모든 화학관련업종에서는 주로 회분식 반응(반 연속식 포함)을 많이 채택하고 있는데 이는 생산량, 제품특성, 제조공정상에 기인하는 것으로서 특히 정밀화학, 의약 등의 고부가가치, 소량다품종 생산업체일수록 회분식 반응이 많음을 알 수 있다. 전반적으로 설문응답 업체의 약 71%인 66개 업체가 회분식 공정을, 29%인 27개 업체가 연속식 공정을 채택하고 있었다.

나. 반응의 형태

일반적으로 화학제품은 사용원료 및 제품의 물리, 화학적 성질에 따라 반응의 형태가 결정되고, 이 반응의 형태에 의해 운전조건 및 설비와 안전장치 등을 선택, 설계하여야 한다.

본 조사에서는 중합반응을 포함하여 13가지의 화학반응 형태를 설문항목으로 선정하여 이를 취합, 분석하였다. 특히 제조공정 특성에 따라 2가지 이상의 반응이 진행되는 경우도 있으므로 이들 통계는 중복 합산하였다.

〈표 3-2〉 회분식 반응의 형태별 종류

반응의 형태	중합	축합	산화	환원	일킬화	에스테르화	디아조화
사업장수(개소)	30	13	8	7	5	7	3
구성비(%)	43.9	19.7	12.1	10.6	7.6	10.6	4.5
반응의 형태	술폰화	니트로화	아미드화	할로겐화	부가	기타	계
사업장수(개소)	5	4	3	4	4	14	66
구성비(5%)	7.6	6.1	4.5	6.1	6.1	21.2	100

즉 〈표 3-2〉에서와 같이 사업장 특성에 따라 취하는 반응의 형태는 매우 다양하였으나 전반적으로 중합반응(Polymerization)이 약 44%로 주류를 이루고 있었으며 축합, 산화 반응도 응답빈도가 비교적 많았다.

다. 반응기 주변설비

화학공장에서 반응기 주변은 항상 위험이 상존한다고 볼 수 있다. 특히 회분식 반응기를 설치, 운영하는 경우는 반응 전, 후는 물론 반응중 시료채취시 등 부득이 하게 취급하는 화학물질이 반응에 주변에 누출, 오염, 확산됨으로서 외부에 점화원이 존재할 경우 이는 곧 대형 폭발·화재로 연결되는 위험을 안고 있는 것이다.

따라서 본 조사에서는 반응기 주변의 위험 방지설비로서 환기, 가스누출 및 경보, 소화설비의 설치 여부를 파악한 결과 이들 세가지를 모두 설비한 경우는 44%, 두가지 설비를 한 경우는 32%, 한가지만 설비한 경우 24%로서 업체의 반

〈표 3-3〉 반응기 주변의 위험방지 설비 현황

구분 \ 설비	설비별 현황				설비 보유 현황			
	환기 설비	가스검지, 경보설비	소화설비	설문대상	1종류	2종류	3종류	계
사업장수(개소)	49	39	50	66	16	21	29	66
구성비(%)	74.2	59.1	75.8	100	24.2	31.8	43.9	100

응특성을 고려한다하더라도 대체적으로 반응기 주변의 위험방지 설비가 취약하다고 볼 수 있다.

라. 반응기 부속 설비

사업장에서 제조, 또는 취급하는 위험물 등의 양이 노동부장관이 정하는 기준 이상인 화학설비(특수화학설비)를 설치할 때는 내부의 이상상태를 조기에 파악하기 위하여 필요한 온도계, 유량계, 압력계 등의 계측장치를 설치하도록 되어 있다(안전규칙 제292조).

또한 이들 특수 화학설비에는 위험상태를 조기에 발견, 조치할 수 있도록 자동경보장치, 긴급차단장치, 예비동력원 등의 설치를 의무화하고 있다(안전규칙 제293조, 제294조, 제295조).

대개의 회분식 반응기는 위와같이 특수 화학설비로 분류되는 경우가 대부분으로서 계측설비, 자동경보장치, 긴급차단장치, 예비동력원 등 법정설비를 구비하고 있어야 각종 이상사태 발생시 이를 효과적으로 제어 가능한 것이다. 연속식 반응기와는 달리 회분식 반응기의 경우는 그 운전특성상 유량계, 액위계 등이 공정에 필요치 않은 경우가 대부분으로서 다음 〈표 3-4〉와 같이 온도, 압력 등 9가지 항목에 대한 반응기 부속설비 설치 여부에 대하여 조사하였다.

〈표 3-4〉 반응기 부속설비 설치현황(설문대상 66개소)

설비 / 구분	계측설비		경보설비		기타설비				
	온도	압력	온도	압력	냉각수	긴급차단	불활성가스	예비동력	압력방출
사업장수(개소)	55	53	39	29	29	26	22	24	45
구 성 비(%)	83.3	80.3	59.1	43.9	43.9	39.4	33.3	36.4	68.2
평 균(%)	81.8		51.5		44.2				

즉 〈표 3-4〉에서와 같이 반응기 부속설비 설치 현황에 대하여 살펴보면 온도, 압력 등 계측설비는 80% 이상이 설치되어 있으나 경보, 기타설비는 상대적으로

낮은 설치율임을 알 수 있다. 특히 긴급차단설비 및 예비동력원의 설치율은 각각 39.4%, 36.4%로서 이상 반응 등 각종 긴급 사태 발생시 이에 적절히 대응할 수 없게 됨으로서 폭발·화재발생 등의 우려가 높다고 할 수 있다.

마. 기타사항

회분식 반응이 보유 사업장 66개소에 대하여 이상반응 등에 의한 폭발·화재 사례건수 및 중대 사고 경험에 대한 설문결과 4개소의 사업장에서 위와같은 사고 경험에 대하여 응답이 있었다. 대개의 경우가 이상반응으로 인한 압력상승이 원인이 되어 폭발·화재에 이른 사고로서 회분식 반응기의 경우 압력계통의 계측, 경보, 압력 방출장치 등이 특히 중요함을 알 수 있었다.

3. 반응 형태별 주요특징

본 조사항목에 포함되어 있는 각종 반응의 형태에 따라 그 반응 메카니즘 및 주요 특징에 대하여 기술하면 다음과 같다.

가. 중합반응(Polymerization)

(1) 반응 메카니즘

$$CH_2 = CHX \xrightarrow{\text{촉매}} (CH_2 - \underset{\displaystyle X}{\underset{|}{CH}})m$$

Monomer　　Polymer

(2) 주요사항

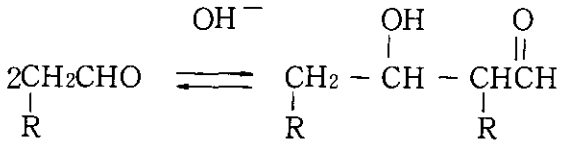
1) Monomer : $CH_2 = CX_2$, $CH_2 = CXY$, $CX_2 = CX_2$

2) 촉매 : 자유라디칼 개시제

예) R−O−O−R (Peroxide)

3) 특징 : 라디칼 반응

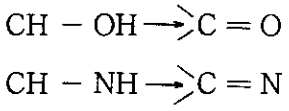
나. 축합반응(Condensation)



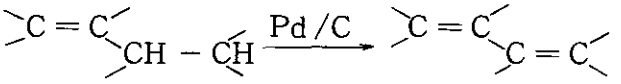
Aldehyde　　Aldol(β-Hydroxy carbonyl compound)

다. 산화반응(Oxidation)

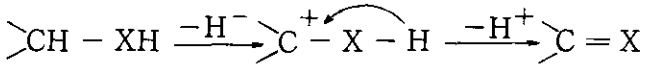
(1) 탈수소화 반응(Dehydrogenation)



(2) 촉매 탈수소화 반응(Catalytic Dehydrogenation)

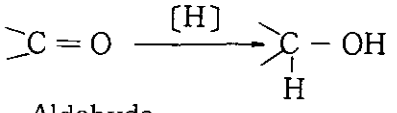


(3) Hydride Proton transfers



라. 환원반응(Reduction)

(1) 반응메카니즘



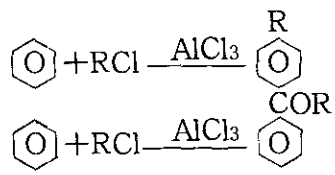
Aldehyde
or Ketone　　Alcohol

예) $RCHO \longrightarrow RCH_2OH$

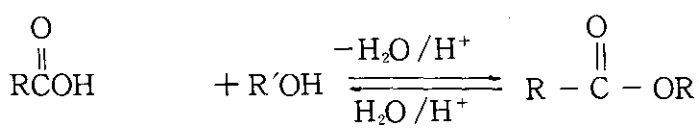
$RCOR \longrightarrow RCHR$ (OH)

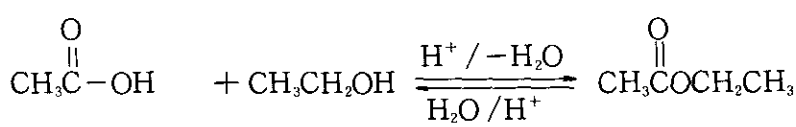
(2) 환원제 : $LiAlH_4$, $NaBH_4$, B_2H_6, AlH_3

마. 알킬화 반응(Alkylation)

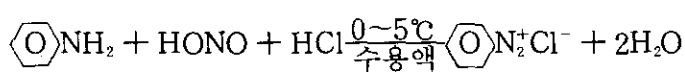


바. 에스테르화 반응(Esterfication)

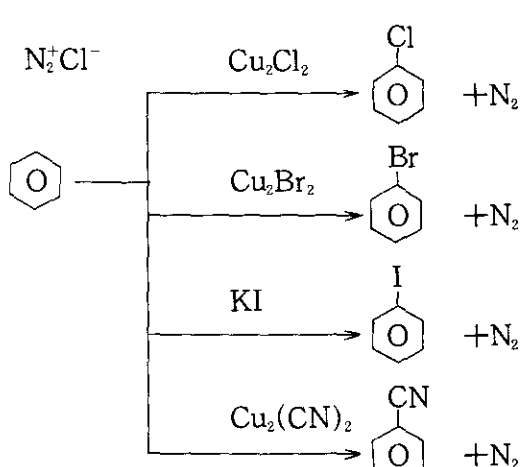




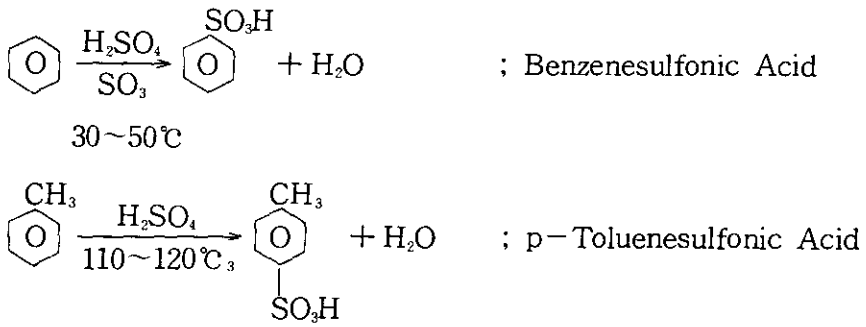
사. 디아조화반응(Diazotization)



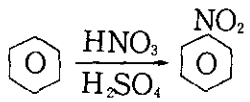
아질산　　디아조늄이온



아. 술폰화반응(Sulfonation)



자. 니트로화 반응(Nitration)



$$HNO_3 + 2H_2SO_4 \rightleftarrows NO_2^+ + H_3O^+ + 2HSO_4^-$$

Nitronium ion

차. 아미드화 반응(Amide)

$$R-\overset{O}{\overset{\|}{C}}-OEt + NH_3 \rightarrow R-\overset{O}{\overset{\|}{C}}-NH_2 + EtOH$$

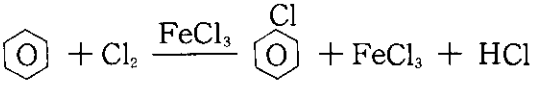
$$R-\overset{O}{\overset{\|}{C}}-Cl + 2NH_3 \rightarrow R-\overset{O}{\overset{\|}{C}}-NH_2 + NH_4Cl$$

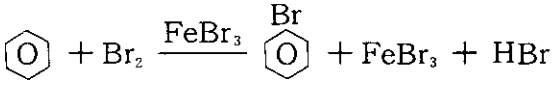
$$R-\overset{O}{\overset{\|}{C}}-Cl + 2CH_3NH_2 \rightarrow R-\overset{O}{\overset{\|}{C}}-NH_3CH_3 + CH_3NH_4Cl$$

$$R-\overset{O}{\overset{\|}{C}}-NH_2 \xrightarrow{LiAlH_4} RCH_2NH_2$$

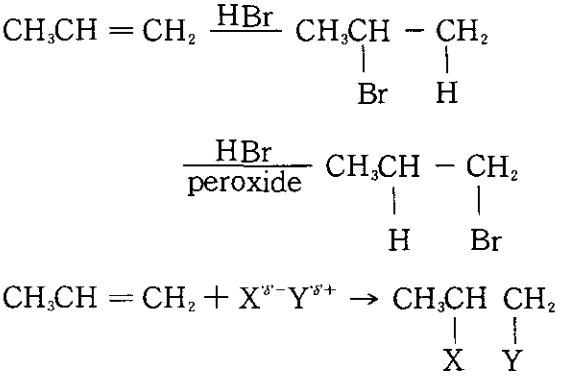
$$\xrightarrow[NaOH]{Br_2} RNH_2$$

카. 할로겐화 반응(Halogenation)





타. 부가반응(Addition)



4. 문제점 분석 및 대책

반응기 사용실태에 관한 설문조사를 분석한 결과 국내 화학공장의 문제점을 정리하면 다음과 같다.

가. 반응기 주변설비

일반적으로 화재의 3대요소로서 가연물, 산소, 점화원을 들수 있는데 화학공장의 경우 취급물질이 대개 가연물인 상태로서 외부에 점화원만 존재하면 언제라도 화재·폭발이 일어날 수 있는 가능성을 갖고 있다고 할 수 있다.

특히 회분식 반응기 주변은 작업의 특성상 원료의 투입, 제품인출 및 반응중

시료의 채취나 조작조건의 변경 등으로 인하여 부득이하게 이들 취급물질이 반응기 주변에 누출, 확산될 가능성이 높고, 또한 운전조건이 단일 설비내에서 가열, 냉각, 진공, 가압 등의 여러 단계를 거치기 때문에 다른 화학설비에 비하여 그 위험성은 매우 크다고 할 수 있다.

따라서 이와같은 문제점을 고려하여 반응기 주변에는 위험방지설비로서 환기설비, 가스누출감지 및 경보설비, 소화설비의 설치가 필수적이라고 할 수 있는데 이들 설비를 모두 갖춘 사업장은 전체의 약 44%에 지나지 않는 등 대체적으로 반응기주변의 위험방지 설비가 취약하였다.

나. 반응기 부속설비

화학공장에 설치되어 있는 회분식 반응기는 여기서 취급하고 있는 위험물 등의 양이 노동부장관이 정하는 기준이상인 특수화학설비로 분류되는 경우가 대부분으로서 계측장치, 자동경보장치, 예비동력원 및 긴급차단장치 등의 설치가 의무화되어 있다.

그러나 본 설문조사에서 나타난 바와 같이 온도, 압력 등의 계측장치는 평균 82%, 경보설비 51%, 긴급차단장치 약 40%에 지나지 않으며 특히, 예비동력원의 경우는 36.4%의 저조한 보유실태로서 이상반응 등 각종 위험사태시 이에 적절히 대처하기에는 많은 문제점이 있는 상황이었다.

특히 온도, 압력 등의 계측설비조차 제대로 갖추지 않은 곳이 약 18%에 이르고있는 실정으로서 이들 사업장은 화학반응이 필요없이 단순혼합공정을 채택하고 있는 경우가 대부분이었지만 조업조건을 전혀 알 수 없는 상황에서 작업을 진행함으로서 잠재적 위험도가 매우 높다고 할 수 있다. 또한 반응기의 압력방출장치는 약 31%, 긴급차단장치는 60%가 미설치되어 있는 실정으로서 회분식 반응기의 안전유지에 중대한 문제점이라 할 수 있다.

이와같이 반응기 부속설비로서 계측설비, 경보설비, 예비동력원 및 안전장치 등의 설치가 제대로 되어 있지 않은 원인을 살펴보면 대개의 회분식 반응은 생산

규모가 작은 중·소규모 사업장에서 채택하고 있어서 이들 사업장의 기술 및 자금력 또는 관리자의 무지에 기인하는 경우가 많았고, 또는 다품종 소량생산공정을 채택하고 있는 경우로서 하나의 반응기에 여러종류를 생산하기 때문에 공정최적화가 이루어지지 못한채 조업이 이루어지는 등 근본적인 문제점을 가지고 있었다.

제 4 장 계측, 제어기기 고장율
(반응기, 저장조를 중심으로)

1. 개 요

본 연구과제의 1차 보고서에서는 화학공장에서 사용되는 계측, 제어기기 전반에 걸쳐 고장율 및 고장원인에 대하여 조사, 보고하였다. 그러나 동일 종류의 계측, 제어기기라고 하더라도 사용장소, 운전조건, 주위환경, 제작업체에 따라 고장원인이나 고장부위, 고장률 등에 현저한 차이가 있기 때문에 1차 보고서에서의 고장율은 화학공장의 계측, 제어기기가 갖는 일반적인 특성에 따른 평균 고장율이라고 할 수 있다.

따라서 위와같은 이유로 인하여 동일 종류의 계측, 제어기기라고 하더라도 이를 설비별 운전특성에 따라 그 고장원인 및 고장율 등을 세분하여 분석하지 않으면 단위 설비별 안전도 분석에 오차가 발생하게 되며 특히 위험도가 높은 반응기 등의 경우는 현저한 차이가 발생할 수도 있는 우려가 있기 때문에 2차년도 과제에서는 화학설비별 계측, 제어기기의 고장율에 대하여 조사하였다. 즉, 화학공장에서 가장 위험도가 높은 반응기와 위험물 저장탱크에 대하여 이들 설비에서 사용되고 있는 온도, 압력, 액위 및 압력방출밸브의 고장율을 집중적으로 조사함으로서 단위설비에 대한 안전도 분석시 기초자료로서 유용하게 사용할 수 있도록 하였다.

2. 계측, 제어기기 고장원인

가. 고장원인 항목

일반적으로 계측, 제어기기의 고장원인은 종류 및 제조원에 따라 상이하며 또한 공정조건, 내용물의 물리, 화학적 성질이나 사용위치에 따라서도 특성이 달라

진다. 1차년도 조사에서는 화학공장에서 사용하는 계측, 제어기기의 고장원인에 대하여 설비별 구분없이 포괄적인 고장원인에 대하여 분석하였으나 2차년도에는 반응기 및 저장조에 대하여 설비별 고장원인을 조사하였다.

〈표 4-1〉 화학설비별 계측, 제어기기 고장원인

종 류 별	총괄('91년도조사자료)	반 응 기	저 장 소
온 도 계 측	센서이상, 접촉불량, 컨버터 불량, 부분침투, 부식, 마모, 제품결함.	센서이상, 부식, 마모, 접촉불량, 제품결함	부식, 센서이상, 제품결함. 접촉불량
압 력 계 류	접촉불양, 이물침투, 오지시, 누설, 파손, 막힘	이물침투, 누설, 막힘, 부식, 파손, 오지시.	부식, 막힘, 오지시, 누설, 이물침투, 파손
액 위 계 류	도압관막힘, 센서이상, 접촉불량, 수분침투 제품결함	도압관막힘, 이물, 부식, 센서이상, 접촉불량, 제품결함	도압관막힘, 부식, 센서이상 접촉불량, 제품결함
압력방출장치	-	이물끼임, 폴리머 형성, 부식	폴리머 형성, 이물끼임. 부식, 동결.

나. 설비별 고장원인 분석

(1) 반응기에서의 고장원인

〈표 4-1〉에서와 같이 화학공장 전반에 걸친 온도, 압력, 액위계류의 주요고장원인은 센서이상, 접촉불량, 이물침투 등이 주류를 이루었으나 반응기의 경우는 부식 및 이물침투로 인한 막힘 등이 고장의 주요원인으로 조사되었다.

이는 반응기에서 취급하는 화학물질의 부식성, 독성과도 연관되며, 특히 회분식 반응기의 경우 운전조건 변화에 따른 가열, 냉각이나 진공, 가압 등으로 인하여 온도, 압력 조작범위가 크게 변하기 때문에 검지부분의 부식이 다른 일반적인 화학설비에 비하여 자주 발생하였던 것으로 판단된다.

또한 반응중에 생성되는 부생성물이나 고형물질 등에 의한 도압관의 막힘 등도

압력 또는 액위계류의 주요 고장원인으로 파악되었다.

(2) 저장조에서의 고장원인

반응기에 비하여 동일 종류의 계측기기가 나타내는 고장율은 현저하게 낮았으나 고장발생의 원인은 비슷한 유형을 나타내었다.

즉 부식에 의한 고장은 반응기에서와 마찬가지로 저장조에서도 높은 빈도를 보여 주었으며 특기할 사항으로는 모노머 등을 저장할 경우 시간경과에 따라 중합체의 형성으로 도압관 등이 막히거나 센서에 이상을 주기도 하고 압력 방출장치(안전밸브)의 경우는 밸브시트에 중합 고형물이 생성되어 작동이 불가능한 경우도 다수 발생되었다.

3. 계측 제어기기 고장율

가. 개 요

본 고장율 조사는 1차년도와 마찬가지로 사업장을 직접 방문하여 공무관련 정비일지와 해당 반응기 및 저장조에 관한 이력카드(History Card)를 참고하여 이를 통계, 분석하였다. 〈표 4-2〉에는 본 조사에 인용된 사업장의 현황을 참고로 기록하였으며 특히 '가', '나'사업장은 1차년도 조사에 인용된 'A', 'E'사업장과 각각 동일함을 밝혀둔다.

〈표 4-2〉 고장율 조사대상 사업장 현황

구분 / 사업장	최초 가동년	업 종	주생산 품목수	통계기간 (일)	단위 공장수	비 고
가	1972년	석유화학	2품목	1087일	7개소	'91년 조사시 A사
나	1976년	정밀화학	2품목	127일	2개소	'91년 조사시 E사
다	1986년	의 약	다품목	543일	4개소	-
라	1991년	석유화학	2품목	471일	3개소	-

본조사 대상 사업장중에서 회분식 공정을 채택하고 있는 곳은 '다'사 뿐으로서 이는 대개의 회분식공정이 중·소규모 사업장에 치중되어 있어 자세한 정비기록 보유하고 있지 않았기 때문에 조사에 애로사항이 있었다.

나. 설비별 계측, 제어기기 평균고장율

1차년도 고장율 조사에서는 온도계류를 포함하여 총 11가지의 계측, 제어기기에 대하여 평균 고장율을 산출하였으나 반응기 및 저장조에는 온도, 압력 및 액위의 계측, 제어부분만 설비되어 있는 경우가 대부분으로서 본조사에서는 이상의 세가지 계측, 제어기기류와 압력용기에서의 안전밸브에 대한 고장율을 분석하였다.

즉 〈표 4-3〉에서와 같이 반응기의 경우는 화학공장 평균 고장율에 비하여 전반적으로 고장율이 높게 나타났으나 저장조의 경우는 현저히 고장율이 낮았다. 이는 화학설비 특성에 기인하는 것으로서 반응기의 경우는 취급물질의 특성과 더불어 운전조건 변화에 따라 온도, 압력 조작범위가 변하기 때문에 검지부분의 부식, 이물끼임 등의 사고가 평균치보다 높았던 것으로 판단된다.

이와같이 동일 화학공장내에서도 설비에 따라 계측, 제어기기의 고장율 및 고장원인이 많은 차이를 나타냄에도 불구하고 FTA 분석자료로서 고장율 데이타를 사용함에 있어서 그동안 주로 미국 WASH-1400에서 제시한 고장율을 인용하거나 가정을 설정하여 안전도 분석을 한 것은 큰 오류라고 할 수 있다.

즉, 현재 가동중인 공장에 대하여 안전도 분석을 수행하고자 할 경우 가장 정확한 고장율 데이타는 해당 사업장의 정비기록을 토대로 하여 산출한 값이라고 할 수 있으나, 이는 현실적으로 각각의 사업장에서 조사, 분석하기에는 많은 어려움이 있는 실정이며 따라서 국내 화학공장에서 사용되는 화학설비별로 고장 및 정비관련 데이타를 적용하는 것이 현실적으로 타당성이 있다고 할 수 있다.

〈표 4-3〉 설비별 평균고장율

(단위 : 회/년)

구분 / 종류	'92 조사		화학공장평균 ('91결과)	Anyakora등	UKAEA
	반 응 기	저 장 조			
온 도 계 류	0.8471	0.1507	0.3516	0.35	0.088
압 력 계 류	0.6776	0.0753	0.3150	1.41	0.76
액 위 계 류	1.2423	0.3316	1.1904	1.70	-
안 전 밸 브	0.1412	0.0678	-	-	-

다. 사업장별 고장율 비교분석

1차년도 조사시 평균 고장율 자료를 확보한 '가', '나'사업장에 대하여 이들 자료와 화학설비별 고장율을 비교 분석한 결과는 〈표 4-4〉와 같다.

즉 '가'사업장의 경우 공장전체의 평균 고장율에 비하여 반응기의 경우 2.3배 내지 4.7배의 높은 값을 나타내었고 저장조는 오히려 평균고장율의 20~30%에 지나지 않음을 알 수 있었다. 또한 이들 설비에 대하여 각각 살펴보면 저장조와 반응기에서의 고장율은 무려 7~20배의 엄청난 차이가 있는 것으로서 이상에서도 보는바와 같이 동일 계측, 제어기기라 할지라도 설치되어 있는 설비의 종류에 따라 매우 큰 차이가 있으므로 안전도 분석시 사용하는 고장율 데이타의 선택에 유의하지 않으면 안된다.

한편 '나'사업장의 경우는 '가'에 비하여 공장전체의 평균 고장율과 설비별 고장율의 차이가 극심하지 않으며 또한 설비간 상호 고장율의 차이도 그다지 많지 않은 편이었다. 이는 '나'사업장의 업종이 정밀화학제품으로서 사용·취급되는 원료가 매우 부식성, 독성이 높은 물질로서 공장의 운전조건보다도 취급물질로 인하여 계측, 제어기기의 고장발생이 많은 것을 의미한다.

따라서 이상과 같은 결과만으로도 고장율 데이타를 결정짓는 주요변수는 운전(가동)년수, 취급물질의 물리, 화학적 성질, 설비별 운전조건 등 여러가지가 있음을 알 수 있고, 이상의 조건이 복합적으로 이루어져 해당 사업장의 평균고장율이

결정되기 때문에 안전도 분석에 사용하는 데이타는 이상과 같은 사항을 고려하여 조사, 분석과정을 거친것을 이용하여야 할 것이다.

〈표 4-4〉 사업장별 고장율 비교

(단위 : 회/년)

사 업 장	가			나		
구 분	공장평균[1)]	반 응 기	저 장 조	공장평균[1)]	반 응 기	저 장 조
온도계류	0.1910	0.9052	0.0420	0.5143	1.2773	0.9580
압력계류	0.2715	0.6278	0.0839	0.7971	0.9580	0.4790
액위계류	0.2081	1.2847	0.0700	3.9977	4.7900	3.3530

주1) : '91년도 연구결과

라. '가'사의 고장율 분석

1차년도에 이어 본 조사에서도 '가'사업장의 정비 관련자료가 자세히 분류되어 있었다 따라서 '가'사의 고장율을 집중 분석함으로서 제품별, 가동년한별 고장 발생율의 변화를 추적해 보고자 한다. '가'사의 현황은 〈표 4-5〉와 같이 '72년부터 최초 가동이 시작되어 7개의 단위 공장을 보유하고 있었으며 2종류의 제품을 생산하고 있었다.

〈표 4-5〉 '가'사의 단위공장별 현황

공 장	1공장	2공장	3공장	4공장	5공장	6공장	7공장
가동년월	72.10	75.10	79.1	84.12	87.4	88	89.3
생산품목	X	Y	Y	X	X	후처리	Y

(1) 반응기에서의 고장율

'가'사의 반응기에서의 계측, 제어기기 평균 고장율을 분석하여 보면 공장 전체의 평균 고장율에 대하여 온도계류 4.74배, 압력계류 2.31배, 액위계류 4.17배의 높은 고장발생빈도를 나타내는 것을 알 수 있다.

이를 다시 단위공장별로 구분해 보면 가장 오래된 1공장의 경우 약 4~7배의

높은 고장 발생빈도를 보였으나 최근에 가동된 7공장의 경우 0.16~2.7배의 고장 발생을 나타내는 등 가동년한이 오래된 공장일수록 공장 전체의 평균 고장율보다 높은 빈도를 보임을 알 수 있었다. 특히 압력, 액위계통은 제7공장의 경우에서 오히려 전체 평균치의 16~32% 밖에 되지 않은 낮은 고장을 나타내는 등 반응기에서의 계측, 제어기기류의 고장은 그 수명(가동년수)과 매우 밀접한 관계를 갖는 것으로 나타났다.

이는 계측 제어기기의 형식(공압식, 전자식)이나 가동년수에 따라 센서 등의 오차 및 부식정도에 따라 그 고장 발생빈도가 연관되는 것으로서 동일 제품의 생산 공정이라 할지라도 PM계획에 차등을 두어 정비작업을 할 필요가 있다고 할 수 있다.

〈표 4-6〉 각공장 평균고장율 대비 반응기에서의 고장발생 비율

(단위 : 배)

구 분	1공장	2공장	3공장	4공장	5공장	7공장	평 균
온도계류	7.17	6.81	12.18	5.34	1.65	2.76	4.74
압력계류	3.84	4.32	5.45	1.54	1.61	0.16	2.31
액위계류	7.00	6.76	6.47	6.31	3.30	0.32	4.71

(2) 저장조에서의 고장율

반응기의 경우와는 달리 저장조의 경우는 공장전체의 평균고장율에 대해 온도계류 22%, 압력계류 30.9%, 액위계류 22.7%의 낮은 고장율을 나타 내었다. 이는 일반적으로 화학물질을 단순 저장하는 경우 설비 자체가 저장물의 물리, 화학적 특성에 맞게 설계되었을 뿐 아니라 온도, 압력 등도 일정한 조건을 갖는 등 변화가 거의 없는 상태로서 계측, 제어기기의 고장 발생빈도가 현저히 낮아짐을 알 수 있다.

〈표 4-7〉 '가'사의 단위공장 및 설비별 고장율

(단위 : 회/년)

종류 / 공장	구 분	온도계류	압력계류	액위계류	안전밸브	비 고
1 공장	공장평균	0.3045	0.2626	0.4075	-	R : 2기
	반 응 기	2.1826	1.0074	2.8542	0.0	T : 4기
	저 장 조	0.0	0.1679	0.0	0.0	
2 공장	공장평균	0.1849	0.3107	0.2982	-	R : 4기
	반 응 기	1.2592	1.3431	2.0147	0.0	T : 16기
	저 장 조	0.0630	0.0839	0.10147	0.126	
3 공장	공장평균	0.0896	0.1540	0.2723	-	R : 4기
	반 응 기	1.0913	0.8395	1.7629	0.252	T : 2기
	저 장 조	0.0	0.0	0.0	0.0	
4 공장	공장평균	0.1466	0.3641	0.1772	-	R : 3기
	반 응 기	0.7835	0.5596	1.1193	0.112	T : -
	저 장 조	-	-	-	-	
5 공장	공장평균	0.2550	0.2603	0.3560	-	R : 4기
	반 응 기	0.4197	0.4197	1.1753	-	T : -
	저 장 조	-	-	-	-	
7 공장	공장평균	0.1822	0.3448	0.3498	-	R : 6기
	반 응 기	0.5037	0.0560	0.1119	-	T : 2기
	저 장 조	0.0	0.0	0.0	0.0	
평 균	공장평균	0.1910	0.2715	0.3081	-	총보유수량
	반 응 기	0.9052	0.6278	1.2847	0.0584	R : 23기
	저 장 조	0.0420	0.0839	0.0700	0.0839	T : 24기

R : 반응기 T : 저장조

4. 문제점분석 및 대책

모든 기계, 설비 및 부품은 그 사용용도와 조작조건 및 주위환경이 변함에 따라서 동일 종류라 하더라도 각각 다른 고장율과 고장부위를 나타낸다. 즉, 이렇게

조건이 어느정도 가혹한가에 따라 상이한 결과를 갖게 되므로 어느 일정한 평균치를 부여된 조건에 대한 고려없이 일괄적으로 적용하는 것은 전혀 다른 결과를 예측하는 오류를 범할 수 있는 가능성이 매우 크다고 할 수 있다. 따라서 화학공장의 안전도 분석이나 신뢰성 평가를 위한 각종 설비 및 부품의 고장율, 분석시 다음 사항을 참고하여 보다 정확한 평가, 분석이 이루어져야 할 것이다.

가. 계측, 제어기기에 대한 사항

최근들어 각종 계측, 제어기기류는 그 원리나 작동방법 등이 매우 다양하고 눈부시게 발전하였다. 즉 과거에는 아날로그, 공압식의 것이 대부분이었으나 근래에는 디지탈, 전자식으로 바뀌면서 DCS, CIM 등 각종 첨단 제어방식에 의해 중앙집중관리형태로 전환되고 있는 실정이다. 이렇게 제어방식이 바뀜에 따라 그 고장율은 현저하게 낮아지는 반면 고장의 발생원인은 매우 다양하고 복잡한 양상을 띄며 또한 고장발생시 수리, 점검이 용이하지 않고 고도의 정비기술과 많은 시간을 필요로 하는 경우가 많다.

따라서 고장발생에 대한 원인이 개개의 부품에 기인하는 것이 아니기 때문에 각 사업장에서 취하고 있는 제어방식에 따라 씨스템 관리로 전환함이 바람직하다고 할 수 있다.

즉, 화학공장에서 주요 계측, 제어 대상으로 꼽을 수 있는 온도, 유량, 압력, 액위 및 농도에 대하여 이들 종류에 따라 분류를 하게 되면 동일 계측, 제어 설비라고 하더라도 설치위치, 조작조건에 따라 상이한 고장원인과 고장율을 나타내게 되므로 단위설비나 단위공정별로 다시 분류를 하여야 보다 정확한 분석이 가능할 것이다.

한편 계측, 제어시스템을 구성하는 각각의 요소(검지부, 기록부, 경보부, 연산제어부, 조작단부 등)에 대하여 세분하며 관리할 수 있다면 아주 바람직하겠으나 현실적으로 사업장 실정을 감안할 때 실효성이 별로 없으며 특히 고장율산출에 있어서는 이용하기 어려운 문제점이 있으므로 시스템 구성요소에 대한 개별적 관

리보다는 통합관리 분석이 오히려 유리하다고 할 수 있다.

나. 화학설비에 대한 사항

앞절에서 기술한 바와 같이 동일 공장내에서 같은 계측, 제어기기를 사용할 경우라도 그 설치장조 및 운전조건에 따라 고장율과 고장부위가 현저한 차이를 나타냄을 알 수 있었다.

즉 화학공장 평균 고장율과 비교하여 볼때 반응기에 설치되어 있는 계측기기류는 1.04~2.4배의 고장율을, 저장조의 경우는 0.24~0.42배의 고장율을 나타내는 것으로 분석되어 대상 화학설비의 종류에 따라 매우 큰 격차를 갖는 것을 알 수 있었다.

따라서 보다 정확한 안전도 분석을 위하여는 계측, 제어기기류, 회전기기류 등의 고장율을 일괄적으로 평균하여 사용할 것이 아니라 동일부품, 기기류라 할지라도 해당 설비에 따라 분류하여 사용하여야 할 것이다.

이와같은 고장율 산출은 대개의 화학공장에서 주요기기, 설비에 대한 이력카드를 기존 이용하고 있기 때문에 이를 활용하는 루프관리 형태를 취함으로서 해결 가능할 것이며 대상 화학설비의 선정은 산업안전보건법에서 분류하고 화학설비, 특수 화학설비 및 화학설비의 부속설비중 폐가스, 분진처리설비 등으로 분류하면 합리적으로 운영 가능할 것이다.

다. 이력관리 방법에 대한 사항

사업장마다 약간씩 차이는 있으나 주요기계, 기구 및 설비에 대한 이력카드를 활용함에 있어서 대개는 설비별로 이력카드를 작성 보관하는 실정이나, 최근에는 컴퓨터통신망의 발달과 더불어 각종 소프트웨어가 많이 보급됨에 따라 이들 이력관리가 종합적으로 처리, 분석이 가능하게 되었다.

즉 위와같은 컴퓨터기술을 이용함으로서 기기별, 설비별, 고장원인별, PM관련 자료별, 공장별, 계절 및 시간대별 등 모든 자료를 정확히 통계, 분석이 가능하게

되었으며 필요에 따라 이들 자료를 가공분석함으로써 재고관리, 작업관리, PM계획 등에도 반영할 수 있게 되었다. 다만 이와 같은 시스템을 도입, 적용하기 위하여는 전사적 전산망 확보와 근거리통신망의 설치, 사업장 특성에 알맞는 소프트웨어의 개발 보급 및 이에 따른 교육 등의 투자가 선행되어야 하므로 아직은 몇몇 대기업을 중심으로 이용되고 있는 실정이다.

제 5 장 FTA를 이용한 안전성 평가

1. 개 요

본 연구의 1차년도 과제 수행시 FTA를 수행할 수 있는 컴퓨터 프로그램을 개발하였다. 프로그램의 주요내용은 정상사상의 발생확률, BICS, 미니멀 컷 셋트 및 패스셋트를 계산하는 것으로서 2차년도에서는 이상과 같은 계산프로그램을 FTA에 관한 기본적 지식을 가진 사람이라면 누구나 쉽게 응용할 수 있도록 입·출력 시스템을 수정, 보완함으로서 사용의 편리성을 도모하였다.

본 연구에서 개발한 컴퓨터 프로그램은 C언어(Turbo-C)를 사용하였으며 FTA 컴퓨터프로그램을 수행하기 위한 컴퓨터는 PC-AT급을 기준으로 하였으나 분석하고자 하는 FT구성의 복잡정도에 따라 메모리 영역이 초과하는 경우도 있기 때문에 간단한 FTA에 이용하는 것이 바람직하고, 복잡한 FT의 경우는 몇개의 FT로 분할하여 프로그램을 수행하여야 한다.

프로그램운영상 하나의 게이트가 분할되는 하부 게이트의 수를 최대 10개 이내로, 게이트명의 스트링은 8자 이내로, 또한 기본사상을 제외한 게이트수를 30개 이내이거나 기본사상의 총수가 30개 이내로 수행할 수 있도록 제한하였으나 이는 분석하고자 하는 FT의 복잡성과 사용 컴퓨터의 기종에 따라 수정 변경이 가능토록 설계하였다.

프로그램에 이용된 알고리즘은 FTA에 많이 사용되는 Fussel 알고리즘으로서 BICS(Boolean Indicated Cut Sets)를 구하여 이로부터 미니멀 컷 셋트와 패스셋트를 얻는 방법을 택하였다.

본 연구에 이용된 Fussell 알고리즘은 MOCUS 프로그램 등에 사용된 것으로서 Semenderes 알고리즘이 Bottom-up형식인데 반하여 정상사상에서 순차적으로 하위단계의 사상을 탐색하는 Top-down방식을 취하고 있는 특징이 있으며 AND게이트는 켓셋구성 기본사상수를 증가시키고 컷셋수를 증가시키는 것은 OR

게이트 뿐이라고 하는 성질을 이용한 특징이 있다.

Fussell 방법으로 최소 켓셋을 구하기 전에 BICS(Boolean Indicated Cut Sets)라 불리는 켓셋을 구할 필요가 있으며 본 프로그램에서는 개개의 BICS를 △x, y 행렬의 각 행으로부터 구할 수 있도록 하였다.

2. 프로그램의 작성

본 연구에서 개발한 컴퓨터 프로그램의 세부내용은 부록에 별도 첨부하였으며 정상사상의 발생확률, BICS, 미니멀 켓 셋트 및 패스 셋트를 구하는 알고리즘은 다음과 같이 수행하였다.

가. 데이타 화일

- 각 게이트에는 형식, 입력 수, 그리고 i번째 게이트의 이름이 있다.
- 기본사상은 x로 시작한다.
- 기본사상의 확율

나. 확율의 계산

논리곱의 확율 $q(A \cdot B \cdot C \cdot \cdots N) = q_A \cdot q_B \cdot q_C \cdot \cdots q_N$

논리합의 확율 $q(A+B+C+ \cdots +N) = 1-(1-q_A)(1-q_B)(1-q_C)\cdots(1-q_N)$

데이타 화일에 입력된 트리(Tree) 구조를 참조하여 각 게이트(Gate)의 확율을 구한다.

이를 위하여 DEPTH FIRST SEATCH 방법을 사용하였다.

- 알고리즘

(1) DEPTH = 0

(2) 현재 게이트의 i = 0

(3) 현재 게이트의 i번째 하위 게이트(Gate)가 기본 사상인지 판별한다.

(4) 기본 사상이 아니면 DEPTH를 증가시키고 STEP (2)로 돌아간다.

(5) 만일 기본사상이면 확율을 구한다.

(6) i를 증가시킨다.

(7) 만일 i가 입력 게이트 수와 같으면 현재 게이트에 대한 확율을 계산하고 DEPTH를 감소시킨다.

(8) DEPTH = 0이 아니면 (3)으로 간다.

(9) 끝낸다.

다. BICS의 행수 계산(X max)

i번째 게이트의 j번째 입력에 대한 파라미터 : $X_{i,\ j}$

i번째 게이트의 파라미터 : X_i

i번째 게이트의 j번째 입력이 기본 사상이면 $X_{i,\ j} = 1$

i번째 게이트의 j번째 입력이 게이트 k이면 $X_{i,\ j} = X_k$

$$X_i = \begin{cases} X_{i,\ 1}\ X_{i,\ 2} \cdots X_{i,\ j} & \text{max i가 AND 게이트 일때} \\ X_{i,\ 1} + X_{i,\ 2} + \cdots + X_{i,\ j} & \text{max i가 OR 게이트 일때} \end{cases}$$

BICS의 열수 계산(y max)

i번째 게이트의 입력에 대한 파라미터 : $y_{i,\ j}$

i번째 게이트의 파라미터 : Y_i

i번째 게이트의 j번째 입력이 기본 사상이면 $Y_{i,\ j} = 1$

i번째 게이트의 j번째 입력이 게이트 k이면 $Y_{i,\ j} = Y_k$

$$Y_i = \begin{cases} y_{i,\ 1} + y_{i,\ 2} + \cdots y_{i,\ j} & \text{max i가 AND 게이트 일때} \\ \max\ y_{i,\ 1},\ y_{i,\ 2} \cdots + y_{i,\ j} & \text{max i가 OR 게이트 일때} \end{cases}$$

라. BICS의 계산

(1) BICS i, j가 AND 게이트인 경우 입력 게이트로의 치환

- i, j번째 게이트의 입력 게이트 수(k)를 읽는다.
- i번째 행의 j + 1부터 있는 게이트를 j + k + 1로 옮긴다.
- i번째 행의 j + 1부터 j + k까지 입력 게이트로 치환한다.

(2) BICS i, j가 OR 게이트인 경우 입력 게이트로의 치환

- i, j번째 게이트의 입력 게이트 수(k)를 읽는다.
- i + 1번째 행부터 현재까지 계산된 최대행(i max)까지를 i + k + 1번째 행부터 i + k + i max행으로 옮긴다.
- i + 1번째 행부터 현재까지 계산된 최대행(i max)까지를 i번째 행과 같이 만든다.
- i + 1번째 행부터 현재까지 계산된 최대행(i max)까지의 j번째열을 i, j번째 게이트의 입력게이트로 치환한다.
- i max = i max + k

마. 미니멀 컷 셋트(Minimal Cut Set)

(1) BICS행열에서 같은 행에 반복되는 기본사상을 제거한다.

(2) 각 기본사상에 솟수를 지정한다.

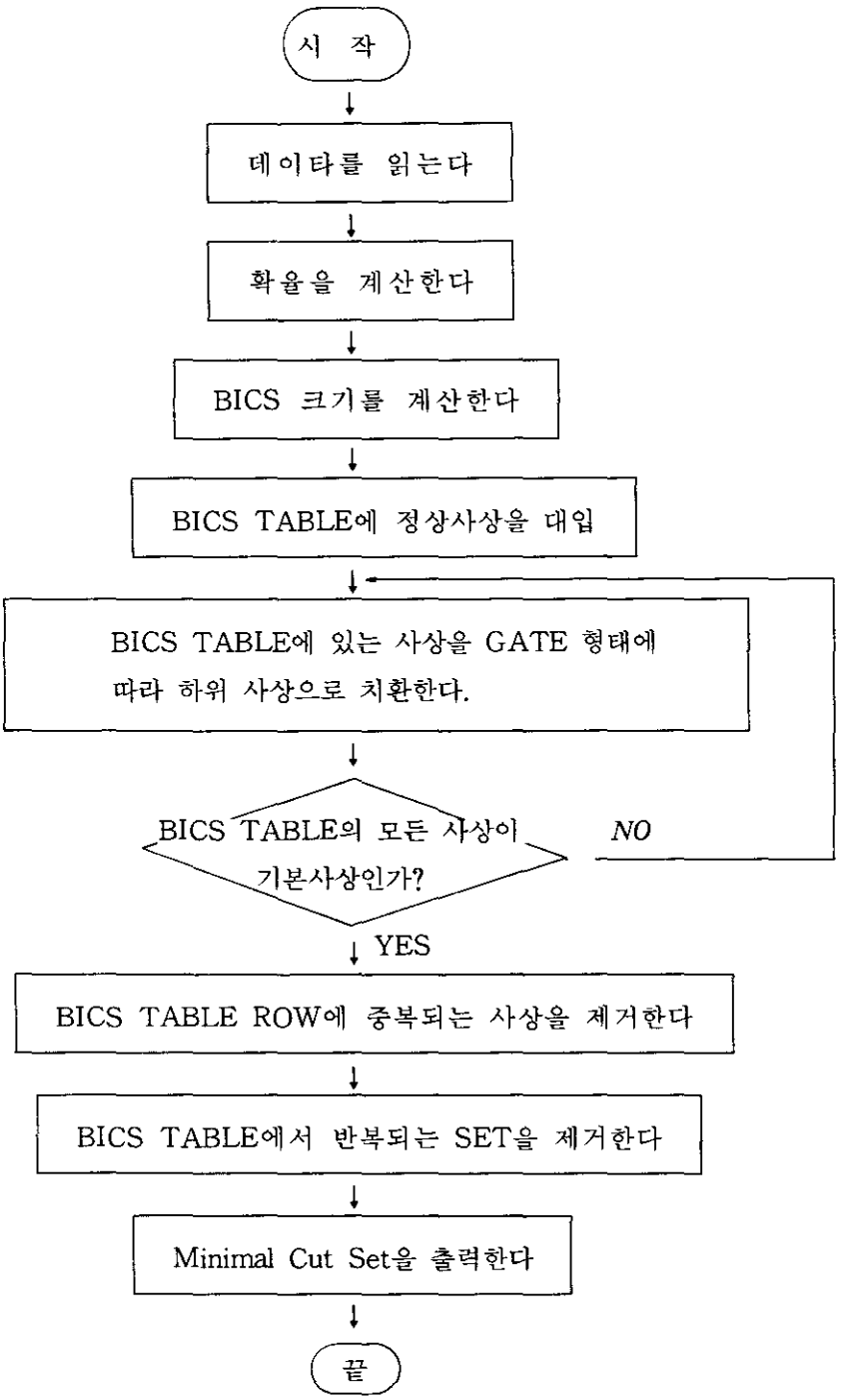
(3) 행에 있는 기본사상에 대응하는 솟수들을 곱하여 행에 대한 값을 구한다.

(4) 다른 행에서 구한 값으로 나누어지는 행이 있으면 BICS에서 제거한다.

(5) 나누어지는 행이 없으면 이것이 미니멀 컷 셋트이다.

바. 플로우 차트(Flow Chart)

본 컴퓨터 프로그램 구성의 플로우 차트는 [그림 5-1]과 같다.



〔그림 5-1〕 최소컷셋을 구하기 위한 FLOW-CHART

3. 프로그램의 구성 및 운영

가. 개 요

FTA(Fault Tree Analysis)를 위한 컴퓨터 프로그램의 개발목적은 286 이상의 PC에서 FTA를 할 수 있는 프로그램 및 toolkit을 만드는데 있다. 프로그램은 Microsoft C-언어를 사용하여 개발하였다. 편리를 위해 자체 한글을 사용하였으며 메뉴방식에 의해 동작하도록 설계하였다. 또한 FTA에 많이 사용되는 Fussell 알고리즘을 이용하였다.

본 프로그램이 수행해 주는 기능은 다음과 같다.

- GATE에 대한 확율의 계산
- MINIMAL CUT SET 계산
- MINIMAL PATH SET 계산

FTA와 관련된 일을 하는 사람이 본 프로그램을 이용할 수 있도록 하기 위해 프로그램의 개발에 사용한 TOOLKIT을 LIBRARY로 만들어 제공하였다.

본 메뉴얼에서는 FTA 프로그램의 구성·사용법 그리고 컴파일 환경을 소개한다.

나. 프로그램의 실행

FTA 방에서 RUNFTA. BAT를 실행시킨다.

RUNFTA. BAT 화일의 내용은 다음과 같다.

```
@ECHO OFF
rem :-------------------------------------------------------------:
rem :                                                             :
rem :           R U N   F T A   B A T C H   F I L E               :
rem :                                                             :
rem :          (Used to start up FAULT TREE ANALYSIS )            :
rem :                                                             :
rem :                          V1.00                              :
rem :-------------------------------------------------------------:
```

```
CD \FTA
ECHO ON

SET  DATPATH=\FTA\DATA
ECHO OFF
IF NOT EXIST FTA.EXE GOTO ERROR
LOGO 10
FTA
GOTO EXIT

:ERROR
ECHO          ***********************************************
ECHO          ***********************************************
ECHO          ****  FTA FAILED TO START                 ****
ECHO          **** CORRECT THE PROBLEM AND TRY AGAIN ****
ECHO          ***********************************************
:EXIT
```

다. FTA 프로그램의 DIRECTORY의 구성

FTA 프로그램에서 사용하는 directory는 다음과 같다.

\FTA\SRC		SOURCE가 저장된 방
	FTA.C	주프로그램
	FTAFILE.C	화일 입출력을 담당
	FTAMAIN.C	계산 함수들
	PDFTA.C	메뉴관리
	README.C	설명을 위한 프로그램
\FTA\OBJ		OBJECT 화일이 저장될 방
\FTA\INC		HEADER 화일이 저장된 방
	MENU.H	메뉴에 관한 함수 정의
	HANIO.H	한글 입출력을 위한 함수 정의
	KEYDEF.H	KEY에 대한 정의
	DCT.H	사전에 대한 정의
	COLOR.H	색깔에 대한 정의

\FTA\SRC		SOURCE가 저장된 방
	WINLIB.H	POP-UP 윈도우에 관한 정의
	DATATYPE.H	데이타에 대한 정의
	FTADEF.H	FTA 함수 정의
	PDFTA.H	데이타의 해석과 출력을 위한 정의
\FTA\LIB		LIBRARY가 저장된 방
	LHANE.LIB	(MSC 5.1, LARGE, EMULATION) 한글라이브러리
	LMENUE.LIB	(MSC 5.1, LARGE, EMULATION) 메뉴라이브러리
	LWINE.LIB	(MSC 5.1, LARGE, EMULATION) 윈도우라이브러리
	LHAN87.LIB	(MSC 5.1, LARGE, 80×87) 한글라이브러리
	LMENU87.LIB	(MSC 5.1, LARGE, 80×87) 메뉴라이브러리
	LWIN87.LIB	(MSC 5.1, LARGE, 80×87) 윈도우라이브러리
\FTA\DATA		DATA 화일이 저장될 방
\FTA		실행화일 및 유틸리티 저장
	RUNFTA.BAT	FTA를 실행키 위한 BATCH화일
	FTA.EXE	FTA 주 프로그램
	LOGO.EXE	LOGO. PIX를 화면에 나타내는 프로그램
	HALOIBMV.DEV	LOGO. EXE에 의해 사용되는 DEVICE화일
	LOGO.PIX	LOGO. EXE에 의해 사용되는 HALO그림
	F.BAT	FTA 프로그램의 컴파일을 위햐 BATCH화일
	FTA.MAK	FTA 프로그램의 컴파일을 위한 MAKE화일
	FTA.ARF	FTA프로그램의 링크를 위한 호아일
	README.EXE	프로그램의 도움말을 화면에 나타내는 프로그램 도움
	README.DOC	말 화일

라. 필요한 사양

FTA 프로그램은 VGA CARD가 설치된 PC-AT 이상에서 동작하도록 설계되었다.

FAULT TREE가 커지면 프로그램의 수행속도가 느려지므로 되도록이면 몇개의 FT로 분할하거나 PC-386 이상의 컴퓨터를 사용하기를 추천한다.

FTA 프로그램의 사용 방법

FTA를 위한 데이타의 입력순서는 다음과 같다.

① 정상사상을 입력한다.

② 기본사상을 입력한다.

③ GATE에 대한 정보를 입력한다.

④ 입력된 데이타를 화일로 저장한다.

⑤ 확율계산, MINIMAL CUT SET 등의 계산을 실행시킨다.

⑥ 출력화일을 확인하다.

출력화일은 FILENAME. OUT이다.

(1) Data type의 정의

다음의 데이타 선언은 "Datatype.h"화일에 나타나 있다.

〈표 5.1〉 DATA TYPE 선언

#define	CHAR	char
#define	UCHAR	unsigned char
#define	INT16	int
#define	UINT16	unsigned int
#define	INT32	long
#define	UINT32	unsigned long
#define	FLOAT	float
#define	DOUBLE	double
#define	VOID	void
#define	FAR	far
#define	NEAR	near
#define	EXTERN	extern

(2) FTA 주 화면

FTA 프로그램의 주 화면은 [그림 6.2.1]과 같다. 이 화면은 FTA를 위한 데이타 입력, FTA의 실행, FTA 데이타화일의 SAVE/LOAD 등을 선택할 수 있다.

```
                    FAULT TREE ANALYSIS  주 화면

                        F4 - 정상사상의 입력
                        F5 - 기본사상 입력
                        F6 - GATE정보 입력
                        F7 - FTA 실행
                        F8 - 출력 화일의 검색
F1 - 도움말        F2 - SAVE        F3 - LOAD        F10 - 빠짐
-------------------------------------------------------------
원하는 메뉴를 선택하시오 :
```

[그림 5.2] FAULT TREE ANALYSIS 주 화면

(3) 정상사상의 입력

해석하려는 정상사상의 데이타를 입력한다.

GATE 이름 : X가 아닌 8자 이내의 영문자열

GATE 형식 : AND, OR

입력 GATE : 8자 이내의 문자열(기본사상 이름 또는 GATE 이름

```
                    정상사상 입력화면

  GATE 이름 : ---
  형       식 : ---
  입  력    1 : ---
  입  력    2 : ---
```

입 력 3 : ---
입 력 4 : ---
입 력 5 : ---
입 력 6 : ---
입 력 7 : ---
입 력 8 : ---
입 력 9 : ---
입 력 10 : ---

F1 – 도움말 F10 – 빠짐 PGUP – 앞 PGDN – 뒤

자료를 입력하시오 :

〔그림 5.3〕 정상사상 입력화면

(4) 기본사상의 입력

기본사상 입력화면에서는 기본사상의 이름과 확율을 입력한다. 기본사상과 중간사상의 구분을 위해 기본사상의 첫자는 X로 시작하며 7자까지 입력가능하다.

사상이름 : X로 시작하는 8자의 영문자열

확 율 : 9.99999E−99(double precision)

기본사상 입력화면

BASE : 1

이 름 : ---
확 율 : ---

F1 – 도움말 F10 – 빠짐 PGUP – 앞 PGDN – 뒤

자료를 입력하시오 :

〔그림 5.4〕 기본사상 입력화면

(5) GATE 정보 입력 화면

GATE 정보 입력화면에서는 FAULT TREE가 형성하기 위해 필요한 정보를 받아들인다. FAULT TREE를 형성하기 위해 필요한 정보는 GATE 이름, GATE 형식, 그리고 입력 GATE의 이름이다.

GATE 이름 : X가 아닌 8자 이내의 영문자열

GATE 형식 : AND, OR

입력 GATE : 8자 이내의 문자열(기본사상 이름 또는 GATE 이름)

GATE 정보 입력화면

GATE 이름 : ---　　GATE : 1
형　　식 : ---
입 력 1 : ---
입 력 2 : ---
입 력 3 : ---
입 력 4 : ---
입 력 5 : ---
입 력 6 : ---
입 력 7 : ---
입 력 8 : ---
입 력 9 : ---
입 력 10 : ---

F1 – 도움말　　F10 – 빠짐　　PGUP – 앞　　PGDN – 뒤

자료를 입력하시오 :

[그림 5.5] Gate 정보 입력화면

(6) 입력화일의 형식

입력화일은 다음과 같은 형식을 갖는 텍스트 화일이다.

```
GATE이름    GATE형식    입력GATE 수    입력GATE 1 … 입력GATE N
…
END
기본사상 1의 이름        기본사상 1의 확율
…
END
```

(예)

```
TOP     AND     2     A1     A2
A1      AND     2     X1     X2
A2      OR      2     X1     X3
END
X1      0.1
X2      0.1
X3      0.1
END
```

입력화일을 위와 같은 형식으로 만들면 FTA 프로그램에서 읽을 수 있으며 FTA 프로그램에서 입력하여 저장하면 위와 같은 형식을 갖는 화일이 형성된다.

(7) 출력 화일

"FTA의 실행"을 선택하면 FILENAME. OUT화일이 생성된다. 화일에는 각 GATE에 대한 확율계산 결과, BICS, MINIMAL CUT SET, BIPS, MINIMAL PATH SET이 쓰여져 있다.

```
#######################################
#              PROBABILITY            #
#######################################
     TOP 1.900000e-003
      A1 1.000000e-002
      A2 1.900000e-001


#######################################
#              BICS SETS              #
#######################################
      X1       X2       X1
      X1       X2       X3


#######################################
#           MINIMAL CUT SETS          #
#######################################
      X1       X2


#######################################
#            BIPS SETS                #
#######################################
      X1
      X2
      X1       X3


#######################################
#           MINIMAL PATH SETS         #
#######################################
      X1
      X2
```

(8) 출력 화일의 검색

출력 화일은 입력된 FAULT TREE의 정보에 따라 크기가 달라지므로 일반 EDITOR를 이용하여 화일 내용을 볼 수 있게 하였다.

(9) 컴파일

프로그램의 효율적인 관리를 위해 MAKE 유틸리티를 사용하였다.

```
################################################
#
#   \FTA\FTA.MAK
#
#
################################################

LIB = \FTA\LIB
SRC = \FTA\SRC
OBJ = \FTA\OBJ
INC = \FTA\INC
EXE = \FTA
LIB_CMD = -+
M_MODEL = L
HANLIB = $(LIB)\$(M_MODEL)HANE
MENULIB = $(LIB)\$(M_MODEL)MENUE
WINLIB = $(LIB)\$(M_MODEL)WINE
LIBS = $(HANLIB) $(MENULIB) $(WINLIB)
COMPILE = CL /DOS_DOS /c /Fo$@ /G2 /FPi /A$(M_MODEL) /W3 /I$(INC) $**

# FTA MAIN PROGRAM ######################################
$(OBJ)\FTA.EBJ:     $(SRC)\FTA.C
   $(COMPILE)

$(OBJ)\PDFTA.EBJ:     $(SRC)\PDFTA.C
    $(COMPILE)

$(OBJ)\FTAFILE.EBJ:     $(SRC)\FTAFILE.C
    $(COMPILE)

$(OBJ)\FTAMAIN.EBJ:     $(SRC)\FTAMAIN.C
    $(COMPILE)

$(EXE)\FTA.EXE: $(OBJ)\FTA.EBJ \
                $(OBJ)\PDFTA.EBJ \
                $(OBJ)\FTAMAIN.EBJ \
                $(OBJ)\FTAFILE.EBJ
   link @FTA.ARF

$(OBJ)\README.EBJ:     $(SRC)\README.C
    $(COMPILE)

$(EXE)\README.EXE: $(OBJ)\README.EBJ
       LINK $**,$@,NUL,$(LIBS) GRAPHICS /NOE
```

FTA.ARF 화일 내용

```
\FTA\OBJ\FTA.EBJ +
\FTA\OBJ\PDFTA.EBJ +
\FTA\OBJ\FTAMAIN.EBJ +
\FTA\OBJ\FTAFILE.EBJ
,\FTA\FTA.EXE,NUL,\FTA\LIB\LHANE \FTA\LIB\LMENUE \FTA\LIB\LWINE GRAPHICS /NOE
```

제 6 장 결 론

화학공장에서의 사고 유형은 대개 화재·폭발 및 이로인한 유독성물질 누출 등이 있으며 이들 사고는 재해규모나 재산손실이 타업종에 비하여 매우크고 사회적 파급 영향이 심각하다. 또한 이들 공장은 일부 중·소규모의 사업장을 제외하고 거의 공업단지에 밀집되어 있어 재해가 확산될 우려가 있으며, 재해발생시 국가 기간산업에도 막대한 영향을 미칠 수 있는 잠재적 위험이 상존한다고 할 수 있다.

특히 화학공장의 경우는 타업종과는 달리 대부분의 공정이 계측, 제어기기에 의해 자동제어되는 실정임을 감안할 때 이들 설비가 안전운전에 차지하는 비중이 매우 높다고 할 수 있으나 이런 중요성에도 불구하고 실재 이들 사업장의 안전과 연계하여 계측, 제어설비에 대한 신뢰도나 고장율 자료가 국내에는 전무하고 외국의 자료도 빈약한 실정이다.

따라서 본 연구의 1차년도 과제로서 국내 화학공장의 계측, 제어기기 사용실태와 이들의 고장원인 및 고장율을 조사하고 FTA를 이용한 안전성 평가에 사용할 수 있도록 컴퓨터 계산용 프로그램을 개발하였다.

2차년도 과제에서는 위와 같은 사항을 기초로하여 화학공장에서 특히 상대적 재해 위험이 높은 반응기와 재해규모가 큰 위험물 저장조 등 단위 화학설비에서의 계측, 제어기기 고장원인과 고장율을 조사분석하고 FTA에 적용할 수 있는 컴퓨터프로그램을 수정, 보완 함으로서 안전성 평가에 관한 기본적인 지식을 가지고 있는 사람이면 누구나 쉽게 이용가능토록하여 화학공장의 재해예방에 기여하고자 하였는바 이에 대한 주요사항은 다음과 같다.

1. 반응의 형태

조사대상 사업장의 약 71%가 연속식이 아닌 회분식 반응공정을 채택하여 조업

하는상태로서 회분식 반응은 주로 소량 다품종 생산에 적합하며 연속공정에 비하여 운전조건이 오히려 까다롭고 단일설비에서 가열, 냉각, 가압, 진공 등 공정변화에 따라 운전상태가 달라지므로 설비의 유지, 관리가 어려운 실정이다. 또한 반응의 종류로는 중합반응이 44%, 축합반응 20% 등 중합·축합반응이 많은 실정이며 기타 산화, 환원반응등 11가지의 반응으로 분류할 수 있었다.

2. 반응기 주변설비

화학공장에서 반응기 주위는 화재·폭발 등 각종 재해나 사고의 위험이 가장 높은 장소로 분류 할 수 있으며 따라서 이와같은 재해예방 및 확대방지를 위하여 환기설비, 가스누출감지 및 경보설비, 소화설비 등의 위험방지설비가 필수적이라고 할 수 있다.

본 조사에서 이상과 같은 세가지의 위험방지설비를 모두다 갖춘 사업장은 44%, 두가지 설비는 32%, 한가지만 설비한 경우는 24%로서 사업장의 반응특성을 고려한다하더라도 반응기 주변의 위험방지 설비가 취약하다고 볼 수 있다.

3. 반응기 부속설비

산업안전 기준에 관한 규칙 제292조 내지 295조에 의하면 특수화학설비에는 내부의 이상상태를 조기에 파악하기위한 계측장치 및 위험상태를 발견·조치할 수 있도록 경보, 차단 등의 안전장치의 설치를 의무화 하고 있다.

회분식 반응기에서 주로 사용되는 온도, 압력의 계측 및 경보설비와 긴급차단장치 등을 설치한 현황은 계측설비 약 82%, 경보설비 53%, 긴급차단설비 등은 평균 44%에 지나지 않는 등 반응기의 재해위험성을 고려할 때 그 부속설비가 미흡한 것으로 나타났다.

4. 계측, 제어기기의 고장원인

1차년도 조사시 화학공장의 전반에 걸친 계측, 제어기기의 주요고장원인은 센서류의 이상, 접촉불량, 이물침투 등 이었으나 이를 단위 화학설비별로 조사한 결과 반응기의 경우는 부식, 도압관막힘, 이물침투로, 저장조의 경우는 부식, 도압관막힘 등이 주요 고장원인으로 파악되었다.

이는 특히 회분식 반응기의 경우 취급물질의 부식성, 독성과도 연결되며, 운전조건 변화에 따라 온도, 압력의 조작범위가 크게 변하기 때문에 검지부분의 부식이 다른 일반화학설비에 비하여 상대적으로 많았던 것으로 판단된다.

5. 계측, 제어기기의 고장율

동일종류의 계측, 제어기기라고 하더라도 단위화학설비에 따라 온도, 압력 등의 조작조건이 다르게 되면 그 고장율도 현저하게 달라지게 된다.

즉 화학공장 평균 고장율에 대하여 반응기의 경우는 온도계류 2.4배, 압력계류 2.2배, 액위계류 1.04배 등의 높은 고장율을 나타내었으나 저장조의 경우는 온도계류 0.43배, 압력계류 0.24배, 액위계류 0.28배의 아주 낮은 고장율을 나타내었다. 따라서 보다 정확한 안전도 분석을 위하여는 계측, 제어기기류, 회전기기류 등의 고장율을 일괄적으로 평균하여 적용할 것이 아니라 해당 단위설비별로 분류하여 적용하여야 할 것이다.

6. FTA 수행용 컴퓨터프로그램

결함수 해석법(FTA)을 수행하기 위한 컴퓨터 프로그램을 개발하였다. 프로그램에 이용된 알고리즘은 Fussel 알고리즘으로서 BICS을 구하여 이로부터 미니멀 컷 셋트와 패스 셋트를 얻는 방법을 택하였고 사용 언어는 Turbo-C를 사용하였으며 대상기종은 PC-AT급을 기준으로 하였다. 프로그램상 하나의 게이트가 분

할되는 하부 게이트 수를 최대 10개 이내로, 게이트 명의 스트링은 8자 이내로, 또한 기본사상을 제외한 수를 30개 이내이거나 기본사상의 총수가 30개 이내로 수행할 수 있도록 제한하였으나 이는 분석하고자 하는 FT의 복잡성과 사용 컴퓨터의 기종에 따라 수정변경이 가능토록 설계하였다. 분석하고자 하는 FT가 위의 사항을 초과하는 경우에는 몇개의 FT로 분할하여 프로그램을 수행한다면 컴퓨터 기억 용량에 큰 제한을 받지 않을 것이다.

참 고 문 헌

1. 이영섭 외, "FTA를 이용한 재해예방모델 개발 연구" 국립노동과학연구소 (1987)
2. 이근철 역, FTA안전공학, 기전연구사, pp. 118~125(1990)
3. 박경수, 신뢰도공학 및 정비이론, 탑출판사, pp. 427~572(1978)
4. D.J.Allen and M.S.Madhava Rao, "New Algorithms for the Synthesis and Analysis of FT", Ind. Eng. Chem. Fundam., 19, 79(1980)
5. D.J.Allen, "Digraphs and FT", Ind. Emg. Chem. Fundam., 23, 175 (1984)
6. F.D.Stevenson, S.T.Maher, D.R.Sharp and B.D.Sloane, "Process Safety Management of a Fuel Gas Conditioning Facility Using FTA", Can.J. of Chem. Eng., 64, 848(1986)
7. G.A.Martin-Solis, P.K.Andow and F.P.Lees, "FT Synthesis for Design and Real Time Analysis", Trans. IChemE., 60, 14(1982)
8. H.Kumamoto and E.J.Henley, "Automated FT Synthesis by Disturbance Analysis", Ind. Eng. Chem. Fundam., 25, 233(1986)
9. H.K.Fauske and J.C.Leung, "New Eoperimental Technique for Characterizing Runaway Chemical Reactions", CEP, AUG., 39(1985)
10. H.Ulrich, "FTA of a Proposed Ethylene Vaporization Unit", Ind. Eng. Chem. Fundam., 20, 304(1981)
11. H.Ulrich and Y. Javier, "FT Evaluation by Monte Carlo Simulation", Chem. Eng., Jan., 91(1983)
12. H.Ulrich and S.Herminio, "Safety Analysis of a Plant for Production of Vinyl Acetate", J. of Chem. Eng of Jap., 17, 165(1984)

13. M.A.Grolmes, J.C.Leung and H.K.Fauske, "Large Scale Experiments of Emergency Relief Systems", CEP, AUG., 57(1985)
14. 川崎義人, 信賴性 設計, 日科技連出版社 東京 pp 13~107()1985)
15. 鹽見 弘, FMEA, FTAの活用, 日科技連出版社, 東京, pp. 163~188(1985)
16. 黃圭錫・富田重幸, 大島榮次, "化學プラントの 操作手順自動合成 システム開發に關する研究", 東京, 化學工學論文集, 第14卷 第 6 號, pp 728-738(1988)
17. 仲勇治, 黃圭錫, 大島榮次, "ヒートポンプシステムの省エネルギー性とその評價" 東京, 化學-學 第49卷 第12號, pp 942~946(1985)

부 록

Ⅰ. 계측, 제어기기 고장율

Ⅱ. '가'사의 계측 제어기기 고장율

Ⅲ. FTA 수행용 컴퓨터프로그램

여 백

부록 Ⅰ. 계측 제어기기 고장율

(단위 : 기, 회, 회/년)

설비명	회사명	보유수	온도계류		압력계류		액위계류		안전밸브		자료수집일수
			횟수	고장율	횟수	고장율	횟수	고장율	횟수	고장율	
반응기	가	23	62	0.9052	43	0.6278	88	1.2847	4	0.0584	1087
	나	9	4	1.2773	3	0.9580	15	4.7900	2	0.6387	127
	다	12	10	0.5602	14	0.7842	13	0.7282	2	0.1120	543
	라	13	14	0.8346	12	0.7153	16	0.9538	7	0.4173	471
	평 균	-	-	0.8471	-	0.6776	-	1.2423	-	0.1412	-
저장조	가	24	3	0.0420	6	0.0839	5	0.0700	6	0.0839	1087
	나	18	6	0.9580	3	0.4790	21	3.3530	2	0.3193	127
	다	17	4	0.1582	1	0.0395	2	0.00791	-	-	543
	라	23	7	0.2359	-	-	16	0.5391	1	0.0337	471
	평 균	-	-	0.1507	-	0.0753	-	0.3316	-	0.0678	-

부록 Ⅱ. '가'사의 계측, 제어기기 고장율

(단위 : 기, 회, 회/년)

공 장	설비보유수		온 도 계 류		압 력 계 류		액 위 계 류		안 전 밸 브	
			횟수	고장율	횟수	고장율	횟수	고장율	횟수	고장율
1	R	2	13	2.1826	6	1.0074	17	2.8542	0	0.0
	T	4	0	0.0	2	0.1679	0	0.0	0	0.0
2	R	4	15	1.2592	16	1.3431	24	2.0147	0	0.0
	T	16	3	0.630	4	0.0839	5	0.1049	6	0.126
3	R	4	13	1.0913	10	0.8395	21	1.7629	3	0.252
	T	2	0	0.0	4	0.0	0	0.0	0	0.0
4	R	3	7	0.7835	5	0.5596	10	1.1193	1	0.112
	T	–	–	–	–	–	–	–	–	–
5	R	4	5	0.4197	5	0.4197	14	1.1753	0	–
	T	–	–	–	–	–	–	–	–	–
7	R	6	9	0.5037	1	0.0560	2	0.1119	0	0.0
	T	2	0	0.0	0	0.0	0	0.0	0	0.0
평 균	R	23	62	0.9052	43	0.6278	88	1.2847	4	0.0584
	T	24	3	0.0420	6	0.0839	5	0.0700	6	0.0830

R : 반응기, T : 저장조

부록 Ⅲ. FTA 수행용 컴퓨터프로그램

```
/*****************************************************************************

        한글 도큐먼트를 화면에 나타내주는 프로그램

    File :  README.C
    BY   :  KWC
        Purpose:
        readme.doc으로 되어 있는 화일을 화면에 나타내주고
        화일의 내용을 조회할 수 있도록 up/down, page up/down
        기능을 부여한다.
*****************************************************************************/

/*===========================================================================
 *
 * INCLUDE FILES:
 *
 * Delimited by: <> - if from the C Compiler common libraries
 *                    (specified by the environment path INCLUDE).
 *               "" - if in current directory or specified by compiler -I
 *                    switch.
 *===========================================================================
 */
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <graph.h>
#include <dos.h>
#include <datatype.h>
#include <hanio.h>
#include <color.h>
#include <keydef.h>
#include <winlib.h>
/*===========================================================================
 *
 * EXTERNAL REFERENCES:
 *
 * Global Data and Function declarations (not defined in include files).
 * Note: Function declarations are a complete prototype, that is, they
 *       define all passed parameters as well as the function type.
 *===========================================================================
 */
VOID main(INT16, CHAR **);

/*===========================================================================
```

```
 *
 * LOCAL DEFINITIONS:
 *
 * Defines, Typedefs, Structures, etc., local to this file
 *============================================================================
 */
#define MAX_LINES   1000
#define PAGE_SIZE   25
/*============================================================================
 *
 * GLOBALS DEFINED IN THIS MODULE:
 *
/*============================================================================
 *
 * LOCAL (STATIC) DATA AND FUNCTIONS:
 *
 * Static data for this file and functions not to be globally
 * recognized
 *============================================================================
 */
VOID dp_page(INT16 line_cnt, CHAR *line[]);

UCHAR file_name[13]={"README.DOC"};
UCHAR *line[MAX_LINES];
/*============================================================================
 *
 * CODE:
 *
 *============================================================================
 */


/*----------------------------------------------------------------------------
 * FUNCTION:    M A I N
 *
 * RETURN:      NULL
 *
 */

VOID main(argc, argv)
INT16 argc;
CHAR *argv[];
{
    INT16 i, read_cnt, cnt, line_cnt, c, in_key, finish;
    INT16 fresh;
    CHAR *result;
    FILE *fp;
    union {
```

```
    INT16 c;
    UCHAR ch[2];
} key;

_setvideomode(_VRES16COLOR);
color_set(WHITE,BLUE);
fp = fopen(file_name,"r+");
if(fp == NULL) {
    err_msg(" README.DOC 화일을 열 수 없읍니다.");
    exit(0);
}
for(i=0; i<MAX_LINES; i++ ) {
    line[i] = calloc(132,sizeof(CHAR));
    if(line[i] == NULL) {
        err_msg(" MEMORY ALLOCATION FAIL");
        exit(0);
    }
}
cnt = 0;
do {
    result = fgets(line[cnt], 132, fp);
    cnt++;
    read_cnt = cnt;
}while(result != NULL && cnt<MAX_LINES );
finish = 0;
line_cnt = 0;
fresh = 1;
do {
    if(fresh) {
        dp_page(line_cnt, line);
        cursor_set(28,0);
        color_set(BLACK,WHITE);
        hprintf("%80s","     PGUP, PGDN, HOME, END, F10(or ESC)    ");
        color_set(WHITE, BLUE);
        fresh = 0;
    }
    key.c = get_key();
    if( key.ch[0] == 0 ) in_key = key.ch[1]+256;
    else                 in_key = key.ch[0];
    switch(in_key) {
        case PGUP:
                line_cnt -= PAGE_SIZE;
                fresh = 1;
                break;
        case PGDN:
                line_cnt += PAGE_SIZE;
                fresh = 1;
                break;
```

```
        case UP  :
                line_cnt--;
                fresh = 1;
                break;
        case DOWN :
                line_cnt++;
                fresh = 1;
                break;
        case HOME :
                line_cnt = 0;
                fresh = 1;
                break;
        case END :
                line_cnt = MAX_LINES;
                fresh = 1;
                break;
        case ESC :
                finish = 1;
                break;
        case F10 :
                finish = 1;
                break;
        default :
                break;
    }
    if( line_cnt < 0 ) line_cnt = 0;
    if( line_cnt+PAGE_SIZE >= read_cnt ) line_cnt = read_cnt - PAGE_SIZE;
}while(!finish);
   for(i=0; i<MAX_LINES; i++ ) {
       free(line[i]);
   }
   _setvideomode(_DEFAULTMODE);
   argc;
   argv;
}
VOID dp_page(INT16 line_cnt, CHAR *line[])
{
    INT16 i;
    cursor_set(2,0);
    _setcolor(BLUE);
    _rectangle(_GFILLINTERIOR,0,0,639,479);

    for( i=line_cnt; i<line_cnt+PAGE_SIZE; i++ ) {
        if( line[i] != NULL ) {
           hprintf("%s", line[i]);
        }
    }
}
```

```
/*****************************************************************************

        FAULT TREE ANALYSIS 주 프로그램

    File :  FTA.C

*****************************************************************************/

/*===========================================================================
 *
 * INCLUDE FILES:
 *
 * Delimited by: <> - if from the C Compiler common libraries
 *                    (specified by the environment path INCLUDE).
 *               "" - if in current directory or specified by compiler -I
 *                    switch.
 *===========================================================================
 */
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <graph.h>
#include <dos.h>
#include <datatype.h>
#include <hanio.h>
#include <color.h>
#include <menu.h>
#include <keydef.h>
#include <winlib.h>
#include <ftadef.h>
#include <pdfta.h>
/*===========================================================================
 *
 * EXTERNAL REFERENCES:
 *
 * Global Data and Function declarations (not defined in include files).
 * Note: Function declarations are a complete prototype, that is, they
 *       define all passed parameters as well as the function type.
 *===========================================================================
 */
VOID main(VOID);
VOID edit_top_main(VOID);
VOID edit_base_main(VOID);
VOID edit_gate_main(VOID);
```

```
/*=============================================================================
 *
 * LOCAL DEFINITIONS:
 *
 * Defines, Typedefs, Structures, etc., local to this file
 *=============================================================================
 */

#define YES 1
#define NO 0
#define FRESH ((UCHAR *) -1)


#define NMX     0x100           /* 사상의 이름 */
#define FRX     0x300           /* 사상의 형식 */
#define INX     0x400           /* 입력 게이트의 이름 */
#define PRX     0x500           /* 확율 */



/*=============================================================================
 *
 * GLOBALS DEFINED IN THIS MODULE:
 *
 *=============================================================================
 */

INT16   menuflag = 0;

/*=============================================================================
 *
 * LOCAL (STATIC) DATA AND FUNCTIONS:
 *
 * Static data for this file and functions not to be globally
 * recognized
 *=============================================================================
 */
/*
 *   Menu Structures:
 *          These are the menu structures which will be displayed
 *    when the configurator is run
 */
static ME   mmu[] = {
     0,"FAULT TREE ANALYSIS V1.0 "      ,  2, 25, 0, 0, 0,
     F4,"F4 - 정상 사상의 입력 "        ,  5, 25, 1, 0, 0,
     F5,"F5 - 기본 사상의 입력 "        ,  7, 25, 1, 0, 0,
     F6,"F6 - GATE 정보의 입력 "        ,  9, 25, 1, 0, 0,
```

```
    F7,"F7 - FTA의 실행          "        , 11, 25, 1, 0, 0,
    F8,"F8 - 출력화일의 검색     "        , 13, 25, 1, 0, 0,
    F1,"F1 - 도움말"                      , 15, 10, 0, 0, 0,
    F2,"F2 - SAVE"                        , 15, 25, 0, 0, 0,
    F3,"F3 - LOAD"                        , 15, 40, 0, 0, 0,
   F10,"F10 - 빠짐"                       , 15, 55, 0, 0, 0,
    0,""
};

static ME   tmu[] = {
    0,"정상사상 입력화면"                 ,  2, 25, 0, 0, 0,
  NMX,"이  름 : "                         ,  5, 10, 8, 8, 0,
  FRX,"형  식 : "                         ,  6, 10, 8, 8, 0,
INX+ 0,"입력  1: "                        ,  7, 10, 8, 8, 0,
INX+ 1,"입력  2: "                        ,  8, 10, 8, 8, 0,
INX+ 2,"입력  3: "                        ,  9, 10, 8, 8, 0,
INX+ 3,"입력  4: "                        , 10, 10, 8, 8, 0,
INX+ 4,"입력  5: "                        , 11, 10, 8, 8, 0,
INX+ 5,"입력  6: "                        , 12, 10, 8, 8, 0,
INX+ 6,"입력  7: "                        , 13, 10, 8, 8, 0,
INX+ 7,"입력  8: "                        , 14, 10, 8, 8, 0,
INX+ 8,"입력  9: "                        , 15, 10, 8, 8, 0,
INX+ 9,"입력 10: "                        , 16, 10, 8, 8, 0,
M_EXIT,"F10: 전화면"                      , 18, 50, 0, 0, 0,
    0,""
};

static ME   bmu[] = {
    0,"기본 사상 입력화면"                ,  2, 25, 0, 0, 0,
  NMX,"이 름 :"                           ,  8, 10, 8, 8, 0,
  PRX,"확 율 :"                           , 10, 10,13,13, 0,
M_PGUP,"PGUP: 앞사상"                     , 14, 10, 0, 0, 0,
M_PGDN,"PGDN: 뒷사상"                     , 14, 30, 0, 0, 0,
M_EXIT,"F10: 전화면"                      , 14, 50, 0, 0, 0,
    0,""
};

static ME   gmu[] = {
    0,"GATE 정보 입력화면"                ,  2, 25, 0, 0, 0,
  NMX,"이  름 : "                         ,  5, 10, 8, 8, 0,
  FRX,"형  식 : "                         ,  6, 10, 8, 8, 0,
INX+ 0,"입력  1: "                        ,  7, 10, 8, 8, 0,
INX+ 1,"입력  2: "                        ,  8, 10, 8, 8, 0,
INX+ 2,"입력  3: "                        ,  9, 10, 8, 8, 0,
INX+ 3,"입력  4: "                        , 10, 10, 8, 8, 0,
INX+ 4,"입력  5: "                        , 11, 10, 8, 8, 0,
INX+ 5,"입력  6: "                        , 12, 10, 8, 8, 0,
```

```
INX+ 6,"입력  7: "                    , 13, 10, 8, 8, 0,
INX+ 7,"입력  8: "                    , 14, 10, 8, 8, 0,
INX+ 8,"입력  9: "                    , 15, 10, 8, 8, 0,
INX+ 9,"입력 10: "                    , 16, 10, 8, 8, 0,
M_PGUP,"PGUP: 앞사상"                 , 18, 10, 0, 0, 0,
M_PGDN,"PGDN: 뒷사상"                 , 18, 30, 0, 0, 0,
M_EXIT,"F10: 전화면"                  , 18, 50, 0, 0, 0,
    0,""
};

WINDOW status;
UCHAR ftapath[41];
UCHAR datpath[41];
UCHAR file_name[13];
UCHAR full_name[41];
UCHAR out_file[13];
BASE base[MAX_GATE];


GATE gate[MAX_GATE];

CHAR ***bics;
CHAR ***bips;

INT16 x_max, y_max, max_base_num, max_gate_num;
INT16 gate_cnt, base_cnt;
/*============================================================================
 *
 * CODE:
 *
 *============================================================================
 */


/*----------------------------------------------------------------------------
 * FUNCTION:    M A I N
 *
 * PURPOSE:     put up main menu
 *
 * RETURN:      NULL
 *
 */
VOID main(VOID)
{
    INT16 action;
    _setvideomode(_VRES16COLOR);
```

```
color_set(WHITE,BLUE);
strcpy(ftapath,getenv("FTAPATH"));
strcpy(datpath,getenv("DATPATH"));
menuflag = 1;
action = 0;
_setcolor(BLUE);
_rectangle(_GFILLINTERIOR,0,0,639,479);
establish_window(&status, 0, 20, 80, 7, WHITE, BLUE);
set_border(&status, DTOP_SSIDE);
while(1) {
    if( menuflag ) {
        menu_dsf(mmu);
        menu_cur(mmu, dp_main);
        menuflag = 0;
    }
    switch(action) {
        case F1  :
        case M_HELP :
            wprintf(&status,"도움말 선택 \n");
            _setvideomode(_DEFAULTMODE);
            system("README");
            _setvideomode(_VRES16COLOR);
            color_set(lWHITE,BLUE);
            _setcolor(BLUE);
            _rectangle(_GFILLINTERIOR,0,0,639,479);
            set_border(&status, DTOP_SSIDE);
            action = 0;
            menuflag = 1;
            break;
        case F2  :
            save_data();
            if( datpath[0] == 0 ) sprintf(full_name,"%s", file_name);
            else                  sprintf(full_name,"%s\\%s",datpath,file_name);
            cmd_msg(" 저장중... ");
            save_data_file(full_name);
            clrmsg();
            action = 0;
            menuflag = 1;
            break;
        case F3  :
            load_data();
            if( datpath[0] == 0 ) sprintf(full_name,"%s", file_name);
            else                  sprintf(full_name,"%s\\%s",datpath,file_name);
            cmd_msg(" 데이타를 읽고 있읍니다... ");
            read_data_file(full_name);
            clrmsg();
            action = 0;
```

```
        menuflag = 1;
        break;
    case F4   : /* 정상사상 입력 선택 */
        wprintf(&status,"정상사상 입력 선택 \n");
        edit_top_main();
        action = 0;
        menuflag = 1;
        break;
    case F5  :
        wprintf(&status,"기본사상 입력 선택 \n");
        edit_base_main();
        action = 0;
        menuflag = 1;
        break;
    case F6  :
        wprintf(&status,"GATE 정보 입력 \n");
        edit_gate_main();
        action = 0;
        menuflag = 1;
        break;
    case F7  :
        wprintf(&status,"FTA 실행중... \n");
        bld_out_file(file_name,out_file);
        ftamain();
        action = 0;
        menuflag = 1;
        break;
    case F8  :
        wprintf(&status,"출력 화일의 검색 \n");
        _setvideomode(_DEFAULTMODE);
        system("q data\\*.OUT");
        _setvideomode(_VRES16COLOR);
        color_set(IWHITE,BLUE);
        _setcolor(BLUE);
        _rectangle(_GFILLINTERIOR,0,0,639,479);
        set_border(&status, DTOP_SSIDE);
        menuflag = 1;
        action = 0;
        break;
    case F10 :
    case M_EXIT:
        wprintf(&status,"FTA 프로그램을 끝냅니다.\n");
        rstr_window(&status);
        delete_window(&status);
        _setvideomode(_DEFAULTMODE);
        exit(0);
        break;
```

```
            default :
                action = F4;
                action = menu_edit(mmu, action, ip_main, dp_main);
                break;
        }
    }
    _setvideomode(_DEFAULTMODE);
}
VOID edit_top_main(VOID)
{
    int action;
    _setcolor(BLUE);
    _rectangle(_GFILLINTERIOR,0,0,639,479);
    set_border(&status, DTOP_SSIDE);
    menuflag = 1;
    action = 0;
    while(1) {
        if( menuflag ) {
            menu_dsf(tmu);
            menu_cur(tmu, dp_top);
            menuflag = 0;
        }
        switch(action) {
            case F1  :
            case M_HELP :
                err_msg(" 도움말이 없읍니다. ");
                wprintf(&status,"도움말 선택 \n");
                action = 0;
                menuflag = 1;
                break;
            case F10 :
            case M_EXIT:
                _setcolor(BLUE);
                _rectangle(_GFILLINTERIOR,0,0,639,479);
                set_border(&status, DTOP_SSIDE);
                return;
                break;
            default :
                action = NMX;
                action = menu_edit(tmu, action, ip_top, dp_top);
                break;
        }
    }
}
```

```
VOID edit_base_main(VOID)
{
    int action;
    _setcolor(BLUE);
    _rectangle(_GFILLINTERIOR,0,0,639,479);
    set_border(&status, DTOP_SSIDE);
    menuflag = 1;
    action = 0;
    while(1) {
        if( menuflag ) {
            menu_dsf(bmu);
            menu_cur(bmu, dp_base);
            cursor_set(5,60);
            hprintf(" BASE : %d ", base_cnt+1);
            menuflag = 0;
        }
        switch(action) {
            case F1  :
            case M_HELP :
                err_msg(" 도움말이 없읍니다. ");
                wprintf(&status,"도움말 선택 \n");
                action = 0;
                menuflag = 1;
                break;
            case M_PGDN :
            case PGDN :
                if( base_cnt < MAX_GATE ) base_cnt++;
                wprintf(&status, " PAGE DOWN \n");
                action = 0;
                menuflag = 1;
                break;
            case M_PGUP :
            case PGUP :
                if( base_cnt > 0 ) base_cnt--;
                wprintf(&status, " PAGE UP \n");
                action = 0;
                menuflag = 1;
                break;
            case F10 :
            case M_EXIT:
                _setcolor(BLUE);
                _rectangle(_GFILLINTERIOR,0,0,639,479);
                set_border(&status, DTOP_SSIDE);
                return;
                break;
            default :
                action = NMX;
```

```
                action = menu_edit(bmu, action, ip_base, dp_base);
                break;
        }
    }
}


VOID edit_gate_main(VOID)
{
    int action;
    _setcolor(BLUE);
    _rectangle(_GFILLINTERIOR,0,0,639,479);
    set_border(&status, DTOP_SSIDE);
    menuflag = 1;
    action = 0;
    gate_cnt = 1;
    while(1) {
        if( menuflag ) {
            menu_dsf(gmu);
            menu_cur(gmu, dp_gate);
            cursor_set(5,60);
            hprintf(" GATE : %d ", gate_cnt);
            menuflag = 0;
        }
        switch(action) {
            case F1  :
            case M_HELP :
                err_msg(" 도움말이 없읍니다. ");
                wprintf(&status,"도움말 선택 \n");
                action = 0;
                menuflag = 1;
                break;
            case M_PGDN :
            case PGDN :
                if( gate_cnt < MAX_GATE ) gate_cnt++;
                wprintf(&status, " PAGE DOWN \n");
                action = 0;
                menuflag = 1;
                break;
            case M_PGUP :
            case PGUP :
                if( gate_cnt > 1 ) gate_cnt--;
                wprintf(&status, " PAGE UP \n");
                action = 0;
                menuflag = 1;
                break;
            case F10 :
```

```
                case M_EXIT:
                    _setcolor(BLUE);
                    _rectangle(_GFILLINTERIOR,0,0,639,479);
                    set_border(&status, DTOP_SSIDE);
                    return;
                    break;
                default :
                    action = NMX;
                    action = menu_edit(gmu, action, ip_gate, dp_gate);
                    break;
            }
        }
    }
```

```
/*****************************************************************************

        FAULT TREE ANALYSIS FILE HANDLING ROUTINE

    File :  FTAFILE.C

*****************************************************************************/

/*===========================================================================
 *
 * INCLUDE FILES:
 *
 * Delimited by: <> - if from the C Compiler common libraries
 *                    (specified by the environment path INCLUDE).
 *               "" - if in current directory or specified by compiler -I
 *                    switch.
 *===========================================================================
 */
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <graph.h>
#include <dos.h>
#include <datatype.h>
#include <hanio.h>
#include <color.h>
#include <menu.h>
#include <keydef.h>
#include <winlib.h>
#include <ftadef.h>

/*===========================================================================
 *
 * LOCAL (STATIC) DATA AND FUNCTIONS:
 *
 * Static data for this file and functions not to be globally
 * recognized
 *===========================================================================
 */

EXTERN WINDOW status;
EXTERN UCHAR datpath[41];
EXTERN UCHAR file_name[13];
EXTERN UCHAR full_name[41];
EXTERN UCHAR out_file[13];
EXTERN BASE base[MAX_GATE];
```

```
EXTERN GATE gate[MAX_GATE];

EXTERN INT16 max_base_num, max_gate_num;

/*============================================================================
 *
 * CODE:
 *
 *============================================================================
 */

VOID bld_out_file(UCHAR *infile, UCHAR *outfile)
{
    while(*infile != 0 && *infile != '.') *outfile++ = *infile++;
    *outfile++ = '.';
    *outfile++ = 'O';
    *outfile++ = 'U';
    *outfile++ = 'T';
    *outfile++ = 0;
}

VOID load_data(VOID)
{
    WINDOW load;
    UCHAR tmp[20];
    establish_window(&load, 20, 5, 40, 3, BLACK, CYAN);
    save_window(&load);
    set_border(&load, DTOP_SSIDE);
    set_title(&load, "LOAD");
    wprintf(&load, "FILE NAME : %s\\", datpath);
    wgets(&load, tmp);
    tmp[11] = 0;
    strcpy(file_name, tmp);
    rstr_window(&load);
    delete_window(&load);
}
```

```
VOID read_data_file(UCHAR *fname)
{
   FILE *fp;
   CHAR form[10], name[10];
   INT16 i, j;
   DOUBLE val;

   if( (fp = fopen(fname,"r+")) == NULL ) {
      err_msg(" INPUT FILE OPEN ERROR ");
      return;
   }

   for( i=0; i<MAX_GATE; i++ ) {
      fscanf(fp, "%s", name);
      if( strcmp(name, "END") != 0 ) {
         strcpy(gate[i].name, name);
         wprintf(&status,"%s \n", gate[i].name);
         fscanf(fp, "%s", form);
         if( strcmp(form, "AND") == 0 )     gate[i].form = AND;
         else if( strcmp(form, "OR") == 0 ) gate[i].form = OR;
         else                               gate[i].form = -1;
         fscanf(fp, "%d", &(gate[i].input_num));
         for( j=0; j<gate[i].input_num; j++ ) {
            fscanf(fp, "%s", gate[i].input_name[j]);
         }
      }
      else {
         max_gate_num = i;
         i=MAX_GATE;
      }
   }
   for( i=0; i<MAX_GATE; i++ ) {
      fscanf(fp, "%s", name);
      if( strcmp(name, "END") != 0 ) {
         strcpy(base[i].name, name);
         wprintf(&status, " %s \n", base[i].name);
         fscanf(fp, "%le", &val);
         base[i].probability = val;
      }
      else {
         max_base_num = i;
         i=MAX_GATE;
      }
   }
   fclose(fp);
}
```

```
VOID save_data(VOID)
{
    WINDOW save;
    UCHAR tmp[20];
    establish_window(&save, 20, 5, 40, 3, BLACK, CYAN);
    save_window(&save);
    set_border(&save, DTOP_SSIDE);
    set_title(&save, "SAVE");
    wprintf(&save, "FILE NAME : %s\\", datpath);
    wgets(&save, tmp);
    tmp[11] = 0;
    strcpy(file_name, tmp);
    rstr_window(&save);
    delete_window(&save);

}
VOID save_data_file(UCHAR *fname)
{
   FILE *fp;
   INT16 i, j;
   if( (fp = fopen(fname, "w+")) == NULL ) {
      err_msg(" 화일저장 실패 ");
      return;
   }
   for( i=0; i<MAX_GATE; i++ ) {
      if( gate[i].name[0] != 0 ) {
         fprintf(fp, "%-8s", gate[i].name);
         wprintf(&status, "%s \n", gate[i].name);
         if( gate[i].form == AND )
            fprintf(fp, "%-8s", "AND");
         else if( gate[i].form == OR )
            fprintf(fp, "%-8s", "OR");
         else
            fprintf(fp, "%-8s", "NONE");
         gate[i].input_num = 0;
         for(j=0; j<MAX_INPUT; j++ ) {
            if( gate[i].input_name[j][0] != 0 ) gate[i].input_num++;
         }
         fprintf(fp, "%-8d", gate[i].input_num);
         for( j=0; j<gate[i].input_num; j++ ) {
            fprintf(fp, "%-8s", gate[i].input_name[j]);
         }
         fprintf(fp, "\n");
      }
   }
   fprintf(fp, "END\n\n");
   for( i=0; i<MAX_GATE; i++ ) {
```

```
        if( base[i].name[0] != 0 ) {
            fprintf(fp, "%-8s ", base[i].name);
            fprintf(fp, "%le \n", base[i].probability);
            wprintf(&status, "%-8s ", base[i].name);
            wprintf(&status, "%le \n", base[i].probability);
        }
    }
    fprintf(fp,"END\n");
    fclose(fp);
}


/*****************************************************************************

        FAULT TREE ANALYSIS FUCTION

    File :  FTAMAIN.C

*****************************************************************************/

/*===========================================================================
 *
 * INCLUDE FILES:
 *
 * Delimited by: <> - if from the C Compiler common libraries
 *                    (specified by the environment path INCLUDE).
 *               "" - if in current directory or specified by compiler -I
 *                    switch.
 *===========================================================================
 */
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <graph.h>
#include <dos.h>
#include <datatype.h>
#include <hanio.h>
#include <color.h>
#include <menu.h>
#include <keydef.h>
#include <winlib.h>
#include <ftadef.h>
/*===========================================================================
 *
 * EXTERNAL REFERENCES:
 *
```

```
 * Global Data and Function declarations (not defined in include files).
 * Note: Function declarations are a complete prototype, that is, they
 *       define all passed parameters as well as the function type.
 *============================================================================
 */

/*============================================================================
 *
 * LOCAL DEFINITIONS:
 *
 * Defines, Typedefs, Structures, etc., local to this file
 *============================================================================
 */

/*============================================================================
 *
 * GLOBALS DEFINED IN THIS MODULE:
 *
 *============================================================================
 */

/*============================================================================
 *
 * LOCAL (STATIC) DATA AND FUNCTIONS:
 *
 * Static data for this file and functions not to be globally
 * recognized
 *============================================================================
 */


EXTERN WINDOW status;
EXTERN UCHAR datpath[41];
EXTERN UCHAR file_name[13];
EXTERN UCHAR full_name[41];
EXTERN UCHAR out_file[13];
EXTERN BASE base[MAX_GATE];

EXTERN GATE gate[MAX_GATE];

EXTERN CHAR ***bics;
EXTERN CHAR ***bips;

EXTERN INT16 x_max, y_max, max_base_num, max_gate_num;
```

```
/*=============================================================================
 *
 * CODE:
 *
 *=============================================================================
 */

VOID ftamain(VOID)
{
   INT16 x, y, i, finished, gate_num;

   calc_prob(0);
   write_prob_data();

/*      BOOLEAN INDICATED CUT SETS CALCULATION */
   x_max = eval_x_range_bics(0);
   y_max = eval_y_range_bics(0);
   bics =(CHAR ***)calloc( x_max,sizeof(CHAR **));
   if( bics == NULL ) panic(" MEMORY ALLOCATION FAIL");
   for( i=0; i<x_max; i++ ) {
      bics[i] =(CHAR **)calloc(y_max, sizeof(CHAR *));
      if( bics[i] == NULL ) panic(" MEMORY ALLOCATION FAIL");
   }
   bics[0][0] =(CHAR *)&(gate[0].name[0]);
   do {
      finished = search_bics(&x,&y);
      if( finished == FALSE ) {
         switch(find_gate_form(&bics[x][y][0], &gate_num) ) {
            case  AND:
                  wprintf(&status,"\n AND GATE");
                  replace_and_gate_bics(x,y, gate_num);
                  break;
            case  OR :
                  wprintf(&status,"\n OR GATE");
                  replace_or_gate_bics(x,y,gate_num);
                  break;
            default  :
                  wprintf(&status,"\n THE END \n");
                  break;
         }
      }
   } while( finished == FALSE );
   write_bics_set();

/* MINIMAL CUT SETS CALCULATION */
   wprintf(&status,"\nELIMIATE_REPEAT_BICS\n");
   eliminate_repeat_bics();
```

```
   wprintf(&status,"MINIMAL_CUT_SET\n");
   make_minimal_cut_set();
   for( i=0; i<x_max; i++ ) {
      free(bics[i]);
   }
   free(bics);

/*      BOOLEAN INDICATED PATH SET CALCULATION */

   x_max = eval_x_range_bips(0);
   y_max = eval_y_range_bips(0);
   bips = (CHAR ***)calloc( x_max , sizeof(CHAR **));
   if( bips == NULL ) panic(" MEMORY ALLOCATION FAIL ");
   for( i=0; i<x_max; i++ ) {
      bips[i] = (CHAR **)calloc( y_max,sizeof(CHAR *));
      if( bips[i] == NULL ) panic(" MEMORY ALLOCATION FAIL ");
   }
   bips[0][0] = gate[0].name;
   do {
      finished = search_bips(&x,&y);
      if( finished == FALSE ) {
         switch(find_gate_form(bips[x][y], &gate_num) ) {
            case  AND:
                  wprintf(&status,"\n AND GATE");
                  replace_or_gate_bips(x,y, gate_num);
                  break;
            case  OR :
                  wprintf(&status,"\n OR GATE");
                  replace_and_gate_bips(x,y,gate_num);
                  break;
            default  :
                  wprintf(&status,"\n THE END \n");
                  break;
         }
      }
   } while( finished == FALSE );
   wprintf(&status,"\nWRITE_BIPS_SET\n");
   write_bips_set();

/*      MINIMAL PATH SET CALCULATION */
   wprintf(&status,"ELIMINATE_REPEAT_BIPS\n");
   eliminate_repeat_bips();
   wprintf(&status,"MAKE_MINIMAL_PATH_SET\n");
   make_minimal_path_set();
   for( i=0; i<x_max; i++ ) {
      free(bips[i]);
```

```
    }
    free(bips);

}

VOID make_minimal_cut_set()
{
    INT32 *prime_set, *bics_val;
    INT16 i, j, k, max_set, devided;
    FILE *fp;
    prime_set = calloc(max_base_num, sizeof(INT32));
    bics_val  = calloc(x_max, sizeof(INT32));

    prime_set[0] = 2;
    max_set = 1;
    i=2;

    while( max_set < max_base_num ) {
       devided = FALSE;
       for( j=0; j<max_set; j++ ) {
          if( i % prime_set[j] == 0 ) devided = TRUE;
       }
       if( devided == FALSE ) {
          prime_set[max_set] = i;
          max_set++;
       }
       i++;
    }

    for( i=0; i<x_max; i++ ) {
       bics_val[i] = 1;
       for( j=0; j<y_max; j++ ) {
          for( k=0; k<max_base_num; k++ ) {
            if( strcmp(base[k].name, bics[i][j]) == 0 ) {
               bics_val[i] = bics_val[i] * prime_set[k];
            }
          }
       }
    }

    for( i=0; i<x_max; i++ ) {
       if( bics_val[i] > 0 ) {
          for( j=i+1; j<x_max; j++ ) {
             if( bics_val[j] > 0 ) {
                if( bics_val[i] > bics_val[j] ) {
                   if( bics_val[i]%bics_val[j] == 0 ) {
                      bics_val[i] = 0;
```

```
                }
             }
             else {
                if( bics_val[j]%bics_val[i]  == 0 ) {
                   bics_val[j] = 0;
                }
             }
          }
       }
    }
    if(datpath[0] == 0 ) sprintf(full_name,"%s", out_file);
    else                 sprintf(full_name,"%s\\%s", datpath, out_file);
    fp = fopen(full_name,"a+");
    if( fp ==  NULL ) {
       err_msg("출력화일을 열 수 없읍니다. ");
       return;
    }
    fprintf(fp,"\n\n\n");
    fprintf(fp,"#######################################\n");
    fprintf(fp,"#          MINIMAL CUT SETS          #\n");
    fprintf(fp,"#######################################\n");
    for(i=0; i<x_max; i++ ) {
       if( bics_val[i] != 0 ) {
          for( j=0; j<y_max; j++ ) {
             if( bics[i][j] == NULL ) {
                fprintf(fp, "%8s", "\0");
             }
             else {
                fprintf(fp, "%8s", bics[i][j]);
             }
          }
          fprintf(fp,"\n");
       }
    }
    fclose(fp);
    free(bics_val);
    free(prime_set);
}

VOID make_minimal_path_set()
{
    INT32 *prime_set, *bips_val;
    INT16 i, j, k, max_set, devided;
    FILE *fp;
    prime_set = calloc(max_base_num,sizeof(INT32));
    bips_val  = calloc(x_max,sizeof(INT32));
```

```
prime_set[0] = 2;
max_set = 1;
i=2;

while( max_set < max_base_num ) {
   devided = FALSE;
   for( j=0; j<max_set; j++ ) {
      if( i % prime_set[j] == 0 ) devided = TRUE;
   }
   if( devided == FALSE ) {
      prime_set[max_set] = i;
      max_set++;
   }
   i++;
}

for( i=0; i<x_max; i++ ) {
   bips_val[i] = 1;
   for( j=0; j<y_max; j++ ) {
      for( k=0; k<max_base_num; k++ ) {
        if( strcmp(base[k].name, bips[i][j]) == 0 ) {
           bips_val[i] = bips_val[i] * prime_set[k];
        }
      }
   }
}

for( i=0; i<x_max; i++ ) {
   if( bips_val[i] > 0 ) {
      for( j=i+1; j<x_max; j++ ) {
         if( bips_val[j] > 0 ) {
            if( bips_val[i] > bips_val[j] ) {
               if( bips_val[i]%bips_val[j] == 0 ) {
                  bips_val[i] = 0;
               }
            }
            else {
               if( bips_val[j]%bips_val[i]  == 0 ) {
                  bips_val[j] = 0;
               }
            }
         }
      }
   }
}
if(datpath[0] == 0 ) sprintf(full_name,"%s", out_file);
else                 sprintf(full_name,"%s\\%s", datpath, out_file);
```

```
    fp = fopen(full_name,"a+");
    if(fp == NULL) {
       err_msg("출력 화일을 열 수 없읍니다. ");
       return;
    }
    fprintf(fp,"\n\n\n");
    fprintf(fp,"#######################################\n");
    fprintf(fp,"#          MINIMAL PATH SETS          #\n");
    fprintf(fp,"#######################################\n");

    for(i=0; i<x_max; i++ ) {
       if( bips_val[i] != 0 ) {
          for( j=0; j<y_max; j++ ) {
             if( bips[i][j]== NULL ) {
                fprintf(fp, "%8s", "\0");
             }
             else {
                fprintf(fp, "%8s", bips[i][j]);
             }
          }
          fprintf(fp,"\n");
       }
    }
    fclose(fp);
    free(bips_val);
    free(prime_set);
}
VOID write_bics_set()
{
    FILE *fp;
    INT16 i, j;
    if(datpath[0] == 0 ) sprintf(full_name,"%s", out_file);
    else                 sprintf(full_name,"%s\\%s", datpath, out_file);
    fp = fopen(full_name, "a+");
    if( fp == NULL ) {
        err_msg(" BICS.DAT FILE OPEN ERROR");
        return;
    }
    fprintf(fp,"\n\n\n");
    fprintf(fp,"#######################################\n");
    fprintf(fp,"#             BICS SETS               #\n");
    fprintf(fp,"#######################################\n");
    for( i=0; i<x_max; i++ ) {
       for( j=0; j<y_max; j++ ) {
          if( bics[i][j] != NULL ) {
             fprintf(fp,"%8s", bics[i][j]);
          }
```

```
      }
      fprintf(fp, "\n");
   }
   if( fclose(fp) == EOF ) panic(" BICS.DAT CLOSE ERROR");
}
VOID write_bips_set()
{
   FILE *fp;
   INT16 i, j;
   if(datpath[0] == 0 ) sprintf(full_name,"%s", out_file);
   else                 sprintf(full_name,"%s\\%s", datpath, out_file);
   fp = fopen(full_name, "a+");
   if( fp == NULL ) {
       err_msg("출력 화일을 열 수 없읍니다. ");
       return;
   }
   fprintf(fp,"\n\n\n");
   fprintf(fp,"#######################################\n");
   fprintf(fp,"#           BIPS SETS                 #\n");
   fprintf(fp,"#######################################\n");
   for( i=0; i<x_max; i++ ) {
      for( j=0; j<y_max; j++ ) {
         if( bips[i][j] == NULL ) {
            fprintf(fp,"%8s", "\0");
         }
         else {
            fprintf(fp,"%8s", bips[i][j]);
         }
      }
      fprintf(fp, "\n");
   }
   if( fclose(fp) == EOF ) panic(" BIPS.DAT CLOSE ERROR");
}
VOID eliminate_repeat_bics()
{
   INT16 i, j, k, l;

   for( i=0; i<x_max; i++ ) {
      for( j=0; j<y_max; j++ ) {
         for( k=j+1; k<y_max; k++ ) {
            if( bics[i][j] != NULL ) {
               if( strcmp(bics[i][j],bics[i][k]) == 0 ) {
                  for( l= k; l<y_max-1; l++ ) {
                     bics[i][l]= bics[i][l+1];
                  }
                  bics[i][y_max-1] = 0;
                  k--;
```

```
            }
         }
      }
   }
}
VOID eliminate_repeat_bips()
{
   INT16 i, j, k, l;

   for( i=0; i<x_max; i++ ) {
      for( j=0; j<y_max; j++ ) {
         for( k=j+1; k<y_max; k++ ) {
            if( bips[i][j] != NULL ) {
               if( strcmp( bips[i][j],bips[i][k]) == 0 ) {
                  for( l= k; l<y_max-1; l++ ) {
                     bips[i][l] = bips[i][l+1];
                  }
                  bips[i][y_max-1] = 0;
                  k--;
               }
            }
         }
      }
   }
}
VOID write_prob_data()
{
   FILE *fp;
   INT16 i;
   if(datpath[0] == 0 ) sprintf(full_name,"%s", out_file);
   else                 sprintf(full_name,"%s\\%s", datpath, out_file);
   fp = fopen(full_name, "w+");
   if( fp == NULL ) {
      err_msg(" 출력화일을 열수 없읍니다.");
      return;
   }
   fprintf(fp,"\n\n\n");
   fprintf(fp,"########################################\n");
   fprintf(fp,"#              PROBABILITY             #\n");
   fprintf(fp,"########################################\n");
   for( i=0; gate[i].name[0] != 0; i++ ) {
      fprintf(fp, "%8s %le\n", gate[i].name, gate[i].probability);
   }
   if( fclose(fp) == EOF ) panic(" PROB.DAT CLOSE ERROR");
}
DOUBLE calc_prob(INT16 level)
```

```
{
   INT16 i, j;
   DOUBLE tmp_prob, previous;
    tmp_prob = 1.0;
    previous = 0.0;
    for( i=0; i<gate[level].input_num; i++ ) {
        if( gate[level].input_name[i][0] == 'x' ||
            gate[level].input_name[i][0] == 'X' ) {
            previous = find_prob(gate[level].input_name[i]);
        }
        else {
            for( j=0; j<max_gate_num; j++ ) {
                if( strcmp(gate[j].name,gate[level].input_name[i]) == 0  ) {
                    previous = calc_prob(j);
                }
            }
            if( previous == 0.0 ) panic(" CAN'T FIND THE PREVIOUS PROBABILITY!");
        }
        if( gate[level].form == AND ) {
            tmp_prob=tmp_prob*previous;
        }
        else if( gate[level].form == OR ) {
            tmp_prob=tmp_prob*(1.0-previous);
        }
        else {
            panic(" PROBABILITY EVALUATION ERROR !");
        }
    }
    if( gate[level].form == OR )   gate[level].probability = 1.0 - tmp_prob;
    else                gate[level].probability = tmp_prob;
    wprintf(&status," %8s = %le \n", gate[level].name, gate[level].probability);
    return(gate[level].probability);
}

DOUBLE find_prob(CHAR *name)
{
   INT16 i;
   for( i=0; i<max_base_num; i++ ) {
      if( strcmp(base[i].name, name) == 0 ) {
         return(base[i].probability);
      }
   }
   wprintf(&status," %8s ** ", name);
   panic(" CAN'T FIND BASIC GATE PROBABILITY");
}
```

```
INT16 eval_y_range_bics( INT16 level )
{
   INT16 i, maxy, yval, y_range;

   if( gate[level].form == AND ) y_range = 0;
   else                          y_range = 0;

   for( i=0, maxy = 0; i<gate[level].input_num; i++ ) {
      if( gate[level].form == AND ) {
         y_range += search_previous_y_bics( gate[level].input_name[i] );
      }
      else if( gate[level].form == OR ) {
        yval = search_previous_y_bics(gate[level].input_name[i]);
        if( maxy < yval ) maxy = yval;
      }
      else {
         panic(" Y_RANGE EVALUATION ERROR !");
      }
   }
   if( gate[level].form == OR ) y_range = maxy;
   return(y_range);
}

INT16 search_previous_y_bics( CHAR *name )
{
   INT16 i;
   if( name[0] == 'x' || name[0] == 'X' )       return(1);
   for( i=0; i<max_gate_num; i++ ) {
      if( strcmp(gate[i].name,name) == 0  ) {
         return(eval_y_range_bics( i ));
      }
   }
   panic(" SEARCH FAIL IN Y ");
}

INT16 eval_y_range_bips( INT16 level )
{
   INT16 i, maxy, yval, y_range;

   if( gate[level].form == OR ) y_range = 0;
   else                         y_range = 0;

   for( i=0, maxy = 0; i<gate[level].input_num; i++ ) {
      if( gate[level].form == OR ) {
         y_range += search_previous_y_bics( gate[level].input_name[i] );
      }
      else if( gate[level].form == AND ) {
```

```
        yval = search_previous_y_bics(gate[level].input_name[i]);
        if( maxy < yval ) maxy = yval;
      }
      else {
         panic(" Y_RANGE EVALUATION ERROR !");
      }
   }
   if( gate[level].form == AND ) y_range = maxy;
   return(y_range);
}

INT16 search_previous_y_bips( CHAR *name )
{
   INT16 i;
   if( name[0] == 'x' || name[0] == 'X' )       return(1);
   for( i=0; i<max_gate_num; i++ ) {
      if( strcmp(gate[i].name,name) == 0  ) {
         return(eval_y_range_bips( i ));
      }
   }
   panic(" SEARCH FAIL IN Y ");
}

INT16 eval_x_range_bics( INT16 level )
{
   INT16 i, x_range;

   if(gate[level].form == AND ) x_range=1;
   else                         x_range=0;
   for( i=0; i<gate[level].input_num; i++ ) {
      if( gate[level].form == AND ) {
         x_range *= search_previous_x_bics( gate[level].input_name[i] );
      }
      else if( gate[level].form == OR ) {
        x_range += search_previous_x_bics( gate[level].input_name[i] );
      }
      else {
         panic(" X_RANGE EVALUATION ERROR !");
      }
   }
   return(x_range);
}
INT16 search_previous_x_bics( CHAR *name )
{
   INT16 i;

   if( name[0] == 'x' || name[0] == 'X')       return(1);
   for( i=0; gate[i].name[0] != 0; i++ ) {
```

```
      if( strcmp(gate[i].name, name) == 0 ) {
         return(eval_x_range_bics( i ));
      }
   }
   wprintf(&status," %8s ", name);
   panic(" SEARCH FAIL IN X ");
}

INT16 eval_x_range_bips( INT16 level )
{
   INT16 i, x_range;

   if(gate[level].form == OR  ) x_range=1;
   else                         x_range=0;
   for( i=0; i<gate[level].input_num; i++ ) {
      if( gate[level].form == OR ) {
         x_range *= search_previous_x_bips( gate[level].input_name[i] );
      }
      else if( gate[level].form == AND ) {
        x_range += search_previous_x_bips( gate[level].input_name[i] );
      }
      else {
         panic(" X_RANGE EVALUATION ERROR !");
      }
   }
   return(x_range);
}

INT16 search_previous_x_bips( CHAR *name )
{
   INT16 i;

   if( name[0] == 'x' || name[0] == 'X')       return(1);
   for( i=0; gate[i].name[0] != 0; i++ ) {
      if( strcmp(gate[i].name, name) == 0 ) {
         return(eval_x_range_bips( i ));
      }
   }
   wprintf(&status," %8s ", name);
   panic(" SEARCH FAIL IN X ");
}

VOID replace_and_gate_bics(INT16 x, INT16 y, INT16 gate_num)
{
   INT16 i, j;
   CHAR ***new_bics;
```

```
    new_bics =(CHAR ***)calloc(x_max,sizeof(CHAR **) );
    if( new_bics == NULL ) {
       panic(" MEMORY ALLOCATION FAILED");
    }
    for( i=0; i<x_max; i++ ) {
       new_bics[i] =(CHAR **)calloc(y_max,sizeof( CHAR *) );
       if( new_bics[i] == NULL ) panic(" MEMORY ALLOCATION FAIL");
    }
    for( i=0; i<x; i++ ) {
       for( j=0; j<y_max; j++ ) {
          new_bics[i][j]= bics[i][j];
       }
    }
    for( j=0; j<y+gate[gate_num].input_num; j++ ) {
       if( j<y ) new_bics[x][j]=bics[x][j];
       else      new_bics[x][j]= gate[gate_num].input_name[j-y];
    }
    for( j=y+gate[gate_num].input_num; j<y_max; j++ ) {
      new_bics[x][j]=bics[x][j-gate[gate_num].input_num+1];
    }
    for( i=x+1; i<x_max; i++ ) {
       for( j=0; j<y_max; j++ ) {
          new_bics[i][j]=bics[i][j];
       }
    }
    for( i=0; i<x_max; i++ ) {
       for( j=0; j<y_max; j++ ) {
          bics[i][j]=new_bics[i][j];
       }
    }
    for( i=0; i<x_max; i++ ) {
       free(new_bics[i]);
    }
    free(new_bics);
}

VOID replace_or_gate_bics(INT16 x, INT16 y, INT16 gate_num)
{
    INT16 i, j;
    CHAR ***new_bics;
    new_bics = calloc(x_max,sizeof(CHAR **));
    if( new_bics == NULL ) {
       panic(" CAN'T ALLOCATION ");
    }
    for( i=0; i<x_max; i++ ) {
       new_bics[i] = calloc(y_max,sizeof(CHAR *));
       if( new_bics[i] == NULL ) {
```

```
            panic(" MEMORY ALLOCATION FAIL");
        }
    }

    for( i=0; i<x; i++ ) {
        for( j=0; j<y_max; j++ ) {
            new_bics[i][j]=bics[i][j];
        }
    }
    for( i=x; i<x+gate[gate_num].input_num; i++ ) {
        for( j=0; j<y_max; j++ ) {
            if( j==y ) new_bics[i][j]=gate[gate_num].input_name[i-x];
            else       new_bics[i][j]=bics[x][j];
        }
    }

    for( i=x+gate[gate_num].input_num; i<x_max; i++ ) {
      for( j=0; j<y_max; j++ ) {
         new_bics[i][j]=bics[i-gate[gate_num].input_num+1][j];
      }
    }
    for( i=0; i<x_max; i++ ) {
        for( j=0; j<y_max; j++ ) {
            bics[i][j]=new_bics[i][j];
        }
    }
    for( i=0; i<x_max; i++ ) {
        free(new_bics[i]);
    }
    free(new_bics);
}

VOID replace_and_gate_bips(INT16 x, INT16 y, INT16 gate_num)
{
    INT16 i, j;
    CHAR ***new_bips;

    new_bips = calloc(x_max,sizeof(CHAR **) );
    if( new_bips == NULL ) {
        panic(" MEMORY ALLOCATION FAILED");
    }
    for( i=0; i<x_max; i++ ) {
        new_bips[i] = calloc(y_max,sizeof(CHAR *));
        if( new_bips[i] == NULL ) panic(" MEMORY ALLOCATION FAIL");
    }

    for( i=0; i<x; i++ ) {
```

```
      for( j=0; j<y_max; j++ ) {
         new_bips[i][j] = bips[i][j];
      }
   }

   for( j=0; j<y+gate[gate_num].input_num; j++ ) {
      if( j<y ) new_bips[x][j] = bips[x][j];
      else      new_bips[x][j] = gate[gate_num].input_name[j-y];
   }
   for( j=y+gate[gate_num].input_num; j<y_max; j++ ) {
     new_bips[x][j]=bips[x][j-gate[gate_num].input_num+1];
   }

   for( i=x+1; i<x_max; i++ ) {
      for( j=0; j<y_max; j++ ) {
         new_bips[i][j]=bips[i][j];
      }
   }

   for( i=0; i<x_max; i++ ) {
      for( j=0; j<y_max; j++ ) {
         bips[i][j] = new_bips[i][j];
      }
   }
   for( i=0; i<x_max; i++ ) {
      free(new_bips[i]);
   }
   free(new_bips);
}

VOID replace_or_gate_bips(INT16 x, INT16 y, INT16 gate_num)
{
   INT16 i, j;
   CHAR ***new_bips;
   new_bips = calloc(x_max,sizeof(CHAR **));
   if( new_bips == NULL ) {
      panic(" CAN'T ALLOCATION ");
   }
   for( i=0; i<x_max; i++ ) {
      new_bips[i] = calloc(y_max,sizeof( CHAR * ) );
      if( new_bips[i] == NULL ) panic(" MEMORY ALLOCATION FAIL");
   }

   for( i=0; i<x; i++ ) {
      for( j=0; j<y_max; j++ ) {
         new_bips[i][j]=bips[i][j];
      }
```

```
    }
    for( i=x; i<x+gate[gate_num].input_num; i++ ) {
       for( j=0; j<y_max; j++ ) {
          if( j==y ) new_bips[i][j] =gate[gate_num].input_name[i-x];
          else       new_bips[i][j] = bips[x][j];
       }
    }

    for( i=x+gate[gate_num].input_num; i<x_max; i++ ) {
      for( j=0; j<y_max; j++ ) {
         new_bips[i][j]=bips[i-gate[gate_num].input_num+1][j];
      }
    }
    for( i=0; i<x_max; i++ ) {
       for( j=0; j<y_max; j++ ) {
          bips[i][j]=new_bips[i][j];
       }
    }
    for( i=0; i<x_max; i++ ) {
       free(new_bips[i]);
    }
    free(new_bips);
}


INT16 find_gate_form( CHAR *gate_name,  INT16 *gate_num  )
{
    INT16 i;
    for( i=0; i<max_gate_num; i++ ) {
       if( strcmp(gate_name,gate[i].name) == 0 ) {
          *gate_num = i;
          return(gate[i].form);
       }
    }
    wprintf(&status," %s ", gate_name);
    panic(" CAN'T FIND THE GATE FORM");
}

INT16 search_bics(INT16 *x, INT16 *y)
{
    INT16 i, j;
    CHAR a;
    for(j=0; j<y_max; j++ ) {
       for( i=0; i<x_max; i++ ) {
          if( bics[i][j] != NULL ) {
             a = bics[i][j][0];
             if( !(a == 'x' || a == 'X' || a == 0 ) ) {
```

```
                *x = i;
                *y = j;
                return(FALSE);
            }
        }
    }
    return(TRUE);
}

INT16 search_bips(INT16 *x, INT16 *y)
{
    INT16 i, j;
    CHAR a;
    for(j=0; j<y_max; j++ ) {
        for( i=0; i<x_max; i++ ) {
            if( bips[i][j] != NULL ) {
                a = bips[i][j][0];
                if( !(a == 'x' || a == 'X' || a == 0 ) ) {
                    *x = i;
                    *y = j;
                    return(FALSE);
                }
            }
        }
    }
    return(TRUE);
}
```

```
/*****************************************************************************

        FAULT TREE ANALYSIS PARSE / DISPLAY 프로그램

    File :  PDFTA.C

*****************************************************************************/

/*===========================================================================
 *
 * INCLUDE FILES:
 *
 * Delimited by: <> - if from the C Compiler common libraries
 *                    (specified by the environment path INCLUDE).
 *               "" - if in current directory or specified by compiler -I
 *                    switch.
 *===========================================================================
 */
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <graph.h>
#include <dos.h>
#include <datatype.h>
#include <hanio.h>
#include <color.h>
#include <menu.h>
#include <keydef.h>
#include <winlib.h>
#include <ftadef.h>
#include <pdfta.h>

/*===========================================================================
 *
 * LOCAL (STATIC) DATA AND FUNCTIONS:
 *
 * Static data for this file and functions not to be globally
 * recognized
 *===========================================================================
 */

EXTERN BASE base[MAX_GATE];
EXTERN GATE gate[MAX_GATE];
EXTERN INT16 base_cnt, gate_cnt;

static DIC dp_form[] = {
    1, "AND", AND,
```

```
    1, "OR" , OR,
    0, ""   ,  -1
};
/*=============================================================================
 *
 * CODE:
 *
 *=============================================================================
 */
/*-----------------------------------------------------------------------------
 * FUNCTION:     I P _ M A I N
 *
 * RETURN:   none
 *
 */

UCHAR    *ip_main(buf,i)
UCHAR    *buf;
INT16 i;
{
    i;
    buf;
    return(0);              /*default is 'ok' and no screen refresh */
}


/*-----------------------------------------------------------------------------
 * FUNCTION:     D P _ M A I N
 *
 * RETURN:   0
 *
 */
INT16     dp_main(buf,i)
UCHAR    *buf;
INT16 i;
{
    i;
    buf;
    return(0);
}
```

```
/*-----------------------------------------------------------------------
 * FUNCTION:     I P _ T O P
 *
 * RETURN:       none
 *
 */

UCHAR    *ip_top(buf,i)
UCHAR    *buf;
INT16 i;
{
    INT16  item,form, j;
    item = i&0xff00;
    switch(item) {
        case    NMX :
                    strcpy(gate[0].name, strupr(buf));
                    break;
        case    FRX :
                    gate[0].form = xdcwtd(strupr(buf),dp_form);
                    break;
        case    INX :
                    for( j=0; j<MAX_INPUT; j++ ) {
                        if( j != i&0x00ff && buf[0] != 0 ) {
                            if( strcmp(buf,gate[0].input_name[j]) == 0 ) {
                                cmd_msg(" 입력사상이 중복되었읍니다. ");
                                buf[0] = 0;
                            }
                        }
                    }
                    strcpy(gate[0].input_name[i&0x00ff],strupr(buf));
                    clrmsg();
                    break;
    }
        return(0);                              /*default is 'ok' and no screen refresh */
}

/*-----------------------------------------------------------------------
 * FUNCTION:     D P _ T O P
 *
 * RETURN:       0
 *
 */
INT16     dp_top(buf,i)
UCHAR    *buf;
INT16 i;
{
    INT16  item;
    item = i&0xff00;
```

```
    switch(item) {
        case    NMX :
                    strcpy(buf, gate[0].name);
                    break;
        case    FRX :
                    if(gate[0].form == AND)     strcpy(buf,"AND");
                    else if(gate[0].form == OR) strcpy(buf,"OR");
                    else                        strcpy(buf,"---");
                    break;
        case    INX :
                    strcpy(buf,gate[0].input_name[i&0x00ff]);
                    break;
    }
    return(0);
}

/*--------------------------------------------------------------------------
 * FUNCTION:    I P _ B A S E
 *
 * PURPOSE: parse a menu selection from base event menu
 *
 *
 * RETURN:      none
 *
 */

UCHAR    *ip_base(buf,i)
UCHAR    *buf;
INT16 i;
{
    INT16  item;
    item = i&0xff00;
    switch(item) {
        case    NMX :
                    strcpy(base[base_cnt].name, strupr(buf));
                    break;
        case    PRX :
                    base[base_cnt].probability = 0;
                    sscanf(buf,"%le",&(base[base_cnt].probability));
                    break;
        default     :
                    break;
    }
    return(0);              /*default is 'ok' and no screen refresh */
}
```

```
/*-------------------------------------------------------------------------
 * FUNCTION:    D P _ B A S E
 *
 * PURPOSE: display base event menu
 *
 *
 * RETURN:      0
 *
 */
INT16     dp_base(buf,i)
UCHAR    *buf;
INT16 i;
{
    INT16  item;
    item = i&0xff00;
    switch(item) {
        case    NMX :
                    strcpy(buf, base[base_cnt].name);
                    break;
        case    PRX :
                    sprintf(buf,"%1.6le", base[base_cnt].probability);
                    break;
        default     :
                    break;
    }
    return(0);
}

/*-------------------------------------------------------------------------
 * FUNCTION:    I P _ G A T E
 *
 * PURPOSE: parse a menu selection from GATE event menu
 *
 *
 * RETURN:      none
 *
 */

UCHAR    *ip_gate(buf,i)
UCHAR    *buf;
INT16 i;
{
    UINT16  item, form, j;
    item = i&0xff00;
    switch(item) {
        case    NMX :
                    strcpy(gate[gate_cnt].name, strupr(buf));
                    break;
```

```
        case    FRX :
                    gate[gate_cnt].form = xdcwtd(strupr(buf),dp_form);
                    break;
        case    INX :
                    for( j=0; j<MAX_INPUT; j++ ) {
                        if( j != i&0x00ff && buf[0] != 0 ) {
                            if( strcmp(buf,gate[gate_cnt].input_name[j]) == 0 ) {
                                cmd_msg(" 입력사상이 중복되었읍니다. ");
                                buf[0] = 0;
                            }
                        }
                    }
                    strcpy(gate[gate_cnt].input_name[i&0x00ff],strupr(buf));
                    clrmsg();
                    break;
        default :
                    break;
    }
    return(0);              /*default is 'ok' and no screen refresh */
}

/*-------------------------------------------------------------------------
 * FUNCTION:    D P _ G A T E
 *
 * PURPOSE: display GATE information
 *
 *
 * RETURN:      0
 *
 */
INT16     dp_gate(buf,i)
UCHAR    *buf;
INT16 i;
{
    INT16  item;
    item = i&0xff00;
    switch(item) {
        case    NMX :
                    strcpy(buf, gate[gate_cnt].name);
                    break;
        case    FRX :
                    if( gate[gate_cnt].form == AND )    strcpy(buf,"AND");
                    else if(gate[gate_cnt].form == OR ) strcpy(buf,"OR");
                    else                                strcpy(buf,"---");
                    break;
        case    INX :
```

```
                    strcpy(buf,gate[gate_cnt].input_name[i&0x00ff]);
                    break;
        default :
                    break;
    }
    return(0);
}
```

화학공장 계측, 제어계통의 신뢰성평가 및
고장원인 탐색기법
(연구보고서 화학 92-2-21)

발 행 일 : 1992. 12. 31
발 행 인 : 원 장 徐 相 學
연구책임자 : 실 장 정 동 인
연구수행자 : 연 구 원 주 종 대
발 행 처 : 한 국 산 업 안 전 공 단
산 업 안 전 연 구 원
화 학 연 구 실
주 소 : 인천직할시 북구 구산동 34-4
전 화 : (032) 518-6484/6

비매품