	고등급(EAL6 이상) 보안마이크로커널 개발				
	작성자	이혁	소속	고려대학교	연구 책임자	김승주
	작성일	2021-10-15	파일명	2021-기술문서01-SW안전기능설계명세서-v2.hwp	
	제목	소프트웨어 안전기능 설계 명세서			

소프트웨어 안전기능 설계 명세서

수정일자	수정자	버전	추가/수정 항목	내 용
2020-06-15	이혁	0.1		초안작성
2020-10-10	이혁	0.5		
2020-11-12	이혁	1.0		
2021-10-15	이혁	2.0		

목차

<표 차례>

<그림 차례>

I. 목적

본 명세서는 고등급 보안 마이크로 커널(CHAOS)에 필요한 안전기능 설계에 관해서 기술하는 것에 그 목적이 있다. 이것을 통해 개발하는 시스템 안전기능의 설계 수준 의존성과 인터페이스에 관해 설명한다.

II. 범위

본 명세서는 고등급 보안 마이크로 커널(CHAOS)에 필요한 안전기능 중 스케줄 기능과 관련된 모듈을 그 범위로 한다.

III. 참고문서

- IEC, IEC61508. “61508 functional safety of electrical/electronic/programmable electronic safety-related systems.“ International electrotechnical commission (1998).
- 소프트웨어 안전 기능 요구사항 명세서 (2019)
- SW 안전성 공통 개발 가이드. “정보통신산업진흥원.“ (2016).
- ChibiOS/RT The Ultimate Guide (RT 6)
  https://www.chibios.org/dokuwiki/doku.php?id=chibios:documentation:books:rt:start

IV. 용어 및 약어정의

없음

V. SW 안전기능 설계 정보

가. SW 안전기능 설계 개체

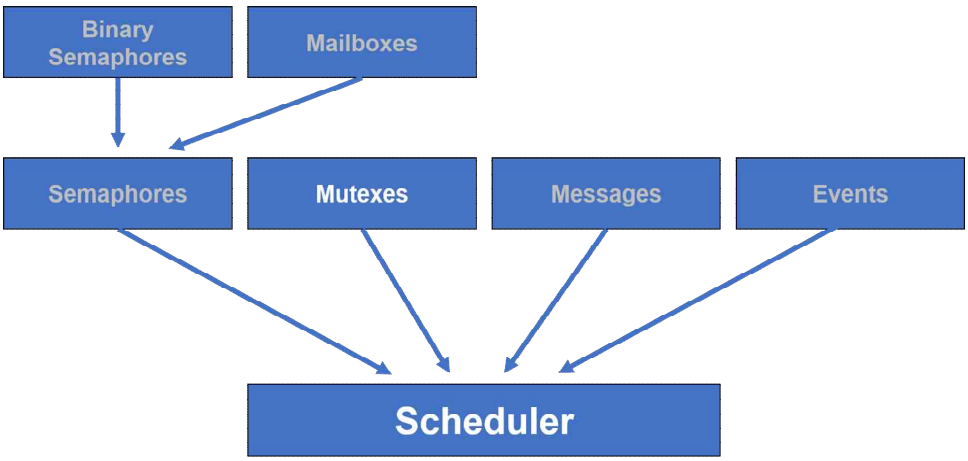


그림 1 CHAOS 커널레벨 시스템 구조

CHAOS의 커널레벨 시스템 구조는 그림2와 같다. 커널 서비스는 기본적으로 스케줄러 모듈을 중심으로 동작을 하며, 그 밖에는 스레드(Thread) 모듈과 가상타이머(Virtual timer) 모듈이 커널의 중심을 구성한다. 안전성 분석을 통해 식별된 여러 안전기능에서 스케줄러와 관련된 안전기능의 안전 목표에 대한 설계 명세에 대해 작성한다.

- SG.1. 자가진단
- SG.2. 도메인 분리
- SG.3. 스케줄링 예측 가능
  - 공정한 자원할당
  - 우선순위 반전문제
  - 스케줄링 가능성 분석 제공
- SG.4. 안전상태 설계
- SG.5. 예외 상황 처리

표 1 개체 목록

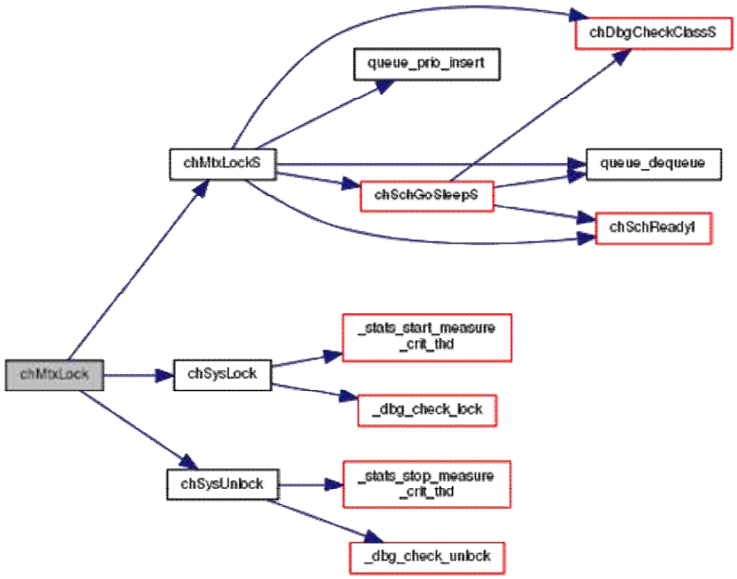
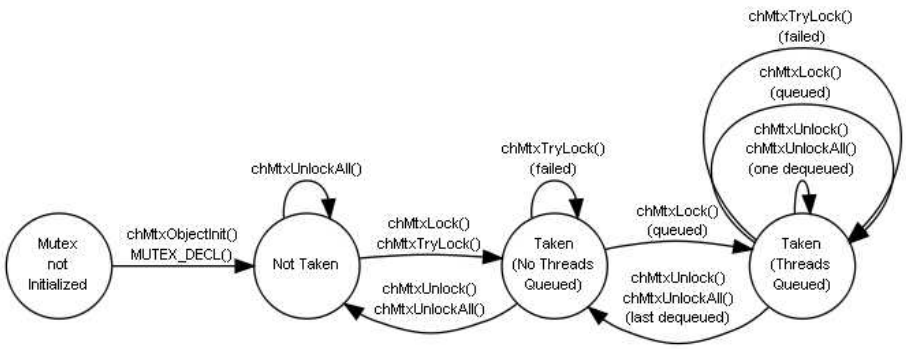
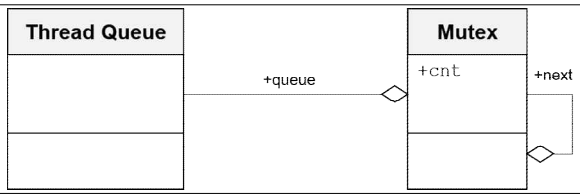
	개체 이름	목적	기능
1	Message	문맥 교환(context switch)이 발생할 때 교환이 이루어지는 스레드 간에 메시지 교환이 한다.	• chMsgSend() • chMsgWait() • chMsgRelease()
2	Events	태스크 간의 통신을 위한 가장 기본적인 방법으로 bit flag의 형태이다.	• chEvtRegisterMaskWithFlags() • chEvtUnregister() • chEvtGetAndClearEventsI() • chEvtGetAndClearEvents()

			• chEvtAddEvents() • chEvtBroadcastFlagsI() • chEvtGetAndClearFlags() • chEvtSignal() • chEvtSignalI() • chEvtBroadcastFlags() • chEvtGetAndClearFlagsI() • chEvtDispatch() • chEvtWaitOne() • chEvtWaitAny() • chEvtWaitAll() • chEvtWaitOneTimeout() • chEvtWaitAnyTimeout() • chEvtWaitAllTimeout()
3	Semaphores	자원에 대한 접근을 제어하는 방법이며, 사용 중인 자원에 대해 표시를 하는 방법으로 사용한다. 태스크는 자원에 대한 접근 권한을 얻기 위해서 세마포어를 획득해야 한다.	• chSemObjectInit() • chSemReset() • chSemResetI() • chSemWait() • chSemWaitS() • chSemWaitTimeout() • chSemWaitTimeoutS() • chSemSignal() • chSemSignalI() • chSemAddCounterI() • chSemSignalWait()
4	Mailboxes	태스크 간의 메시지를 전달하는 목적으로 사용된다. 메시지의 크기는 구현에 의존되며 일반적으로는 고정된 크기를 갖는다.	• chMBObjectInit() • chMBReset() • chMBResetI() • chMBPostTimeout() • chMBPostTimeoutS() • chMBPostI() • chMBPostAheadTimeout() • chMBPostAheadTimeoutS() • chMBPostAheadI() • chMBFetchTimeout() • chMBFetchTimeoutS() • chMBFetchI()
5	Mutexes	상호배제적인 세마포어로 이진 세마포어와 유사하게 동작한다. 임시 소유와 중복 소유를 허락한다.	• chMtxObjectInit() • chMtxLock() • chMtxLockS() • chMtxTryLock() • chMtxTryLockS() • chMtxUnlock() • chMtxUnlockS() • chMtxUnlockAllS() • chMtxUnlockAll()

나. 개체 속성

Mutex 개체의 속성은 다음과 같다.

표 2 개체 속성

개체 속성	내용
개체명	Mutex
개체 타입	모듈
목적	자원에 대한 접근을 제어하는 방법이며, 사용 중인 자원에 대해 표시를 하는 방법으로 사용한다. 태스크는 자원에 대한 접근 권한을 얻기 위해서 뮤텍스를 획득해야 한다.
종속관계	
의존관계	
상호작용	
자원	Priority inheritance protocol 기반으로 상호 뮤텍스락 수행 시 데드락 발생 가능
처리	생성조건: 생성을 위한 포인터가 Null이 아닌 경우 생성순서: chMtxObjectInit() 사용순서: chMtxLock()--> chMtxLockS()--> chMtxUnlockS()--> chMtxUnlock() 또는 chMtxLockS()--> chMtxUnlockS()

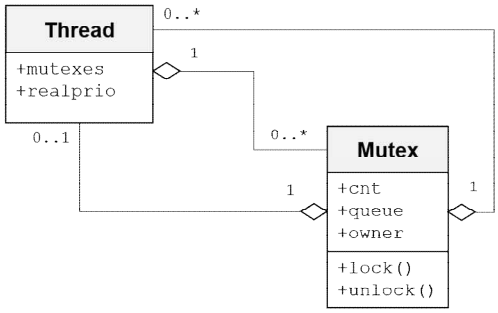
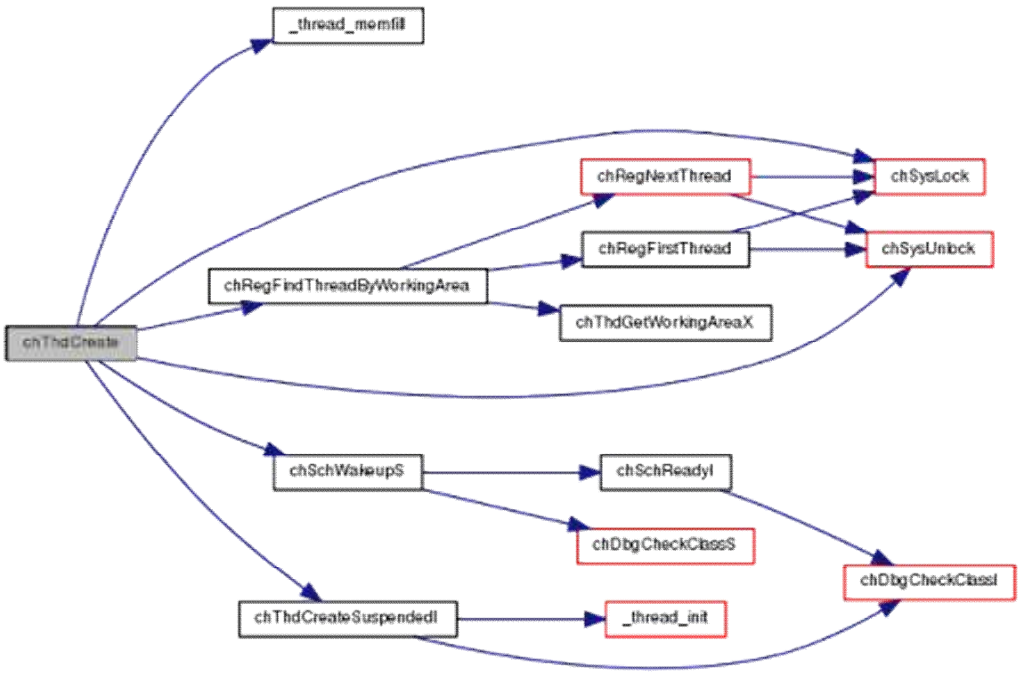
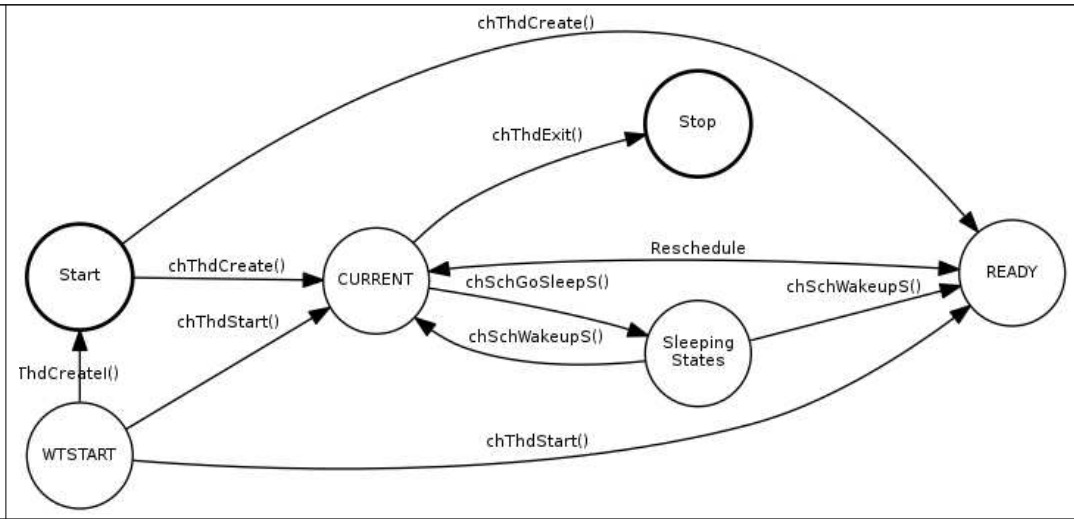
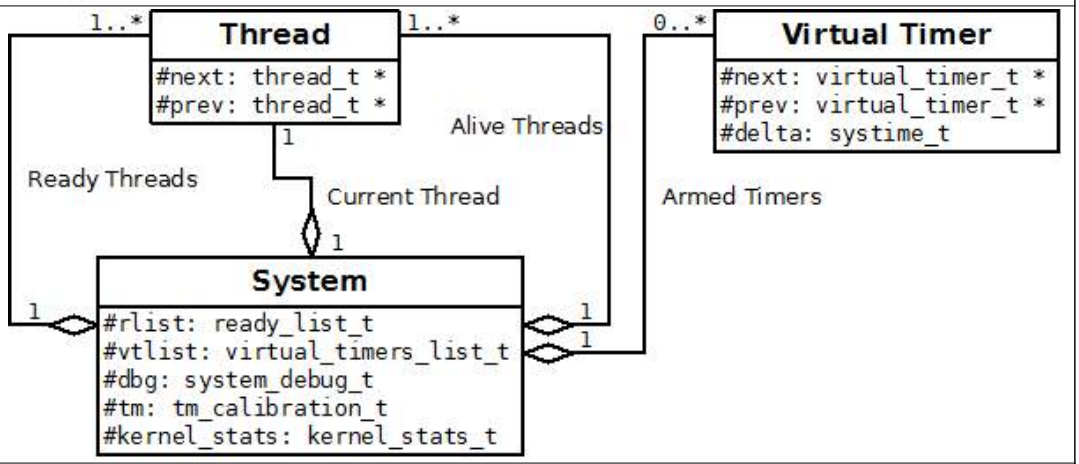
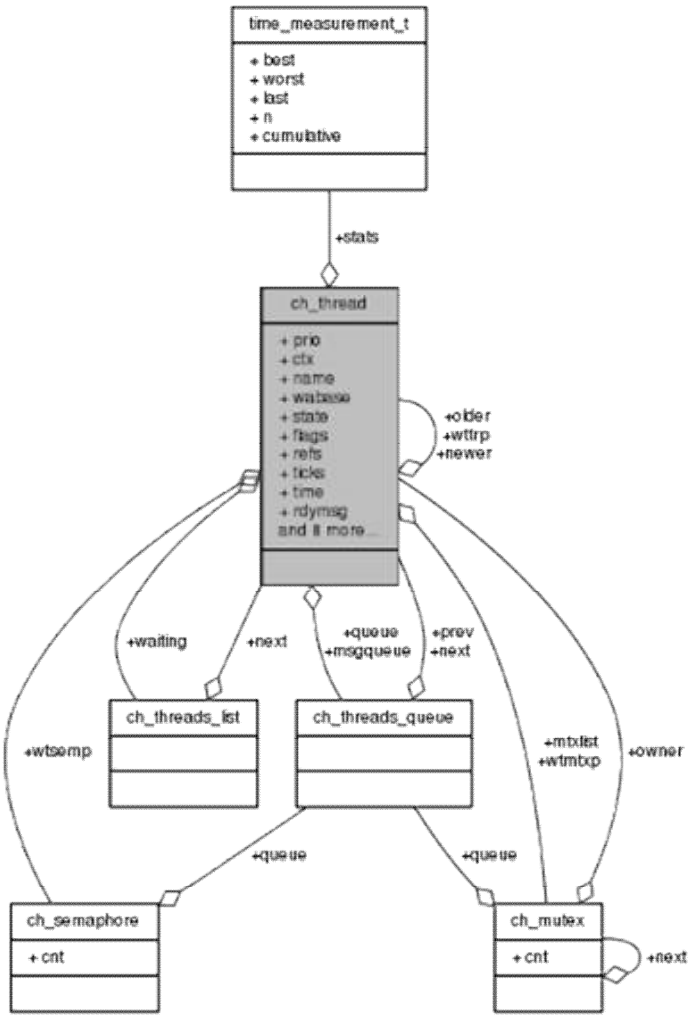
데이터	

표 3 개체 속성

개체 속성	내용
개체명	Thread
개체 타입	모듈
목적	작업을 수행하기 위해 스레드의 생성하여 사용한다. 스레드 생성을 위해서는 스레드의 상태와 WA(Working Area)범위를 설정하여 사용한다.
종속관계	

의존관계	
상호작용	
자원	-
처리	생성조건: 생성을 위한 Working Area가 Null이 아닌 경우 생성순서: chThdCreate() 사용순서: chThdCreate(&idle_descriptor)

데이터



VI. SW 안전기능 설계

Mutex의 컨트롤 플로우 및 상태전이를 나타내는 상태차트는 [그림 1]과 같으며 chMtxLock()을 나타낸다. chMtxLock()을 호출하기 위한 사전조건은 '뮤텍스 초기화'이다. 여기서는 뮤텍스가 초기화되어있는 것으로 가정한다. 뮤텍스의 상태는 어떤 스레드에 의해 '소유된 상태'와 '소유되지 않은 상태'로 나뉘며, '소유된 상태'에서는 같은 스레드가 다시 소유하고자 하는지에 대해서 검사하여 중복 소유 상태를 갖게 된다.

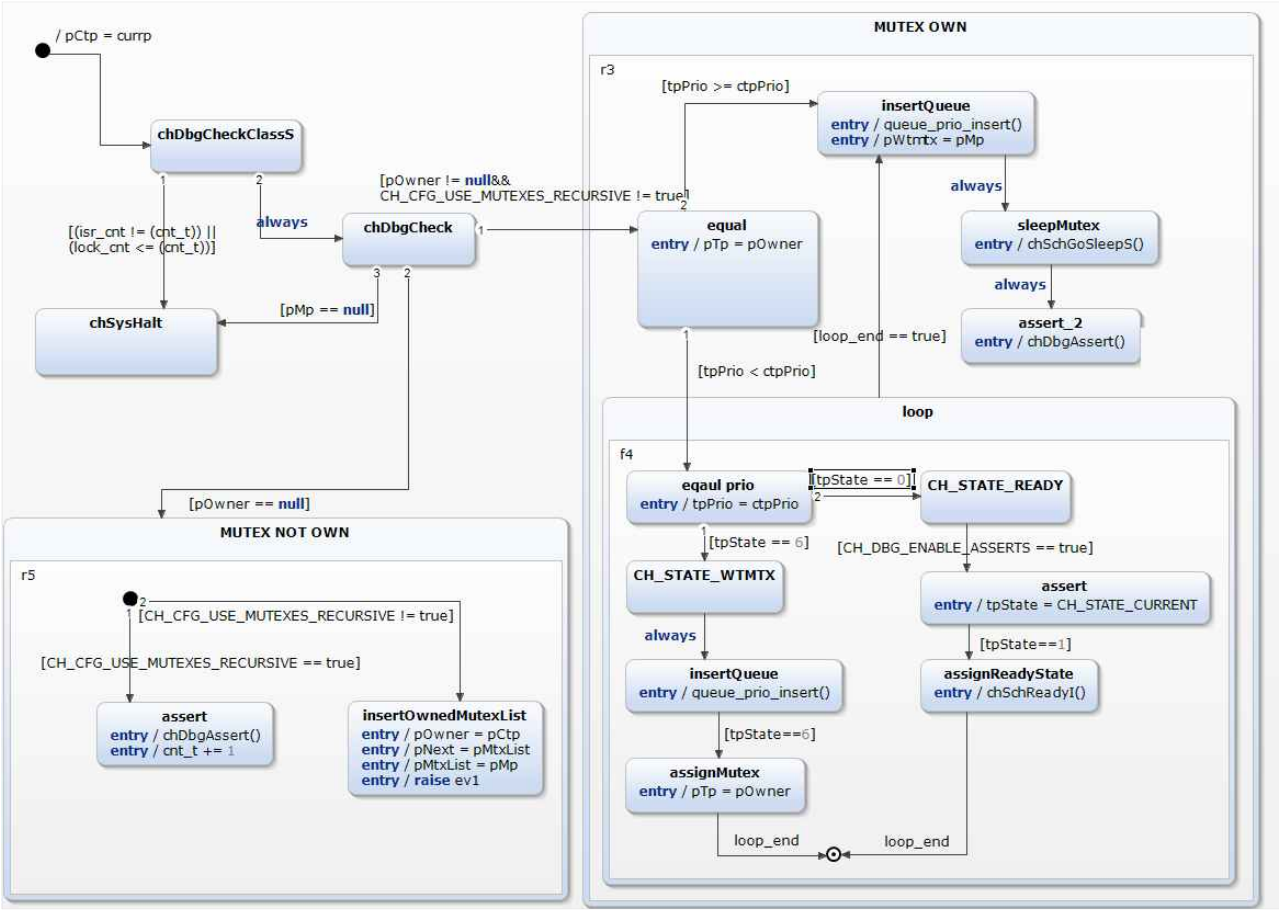


그림 2 뮤텍스 컨트롤 플로우 및 상태전이 설계 명세

1) 안전기능: 자가진단

chDbgCheckClassS() 기능은 '자가진단'을 수행하는 안전기능으로 함수호출 권한을 확인한다. 본 기능은 해당 함수에 대한 호출 권한이 없는 접근에 대해서는 사전에 정의된 예외처리 절차를 수행한다.

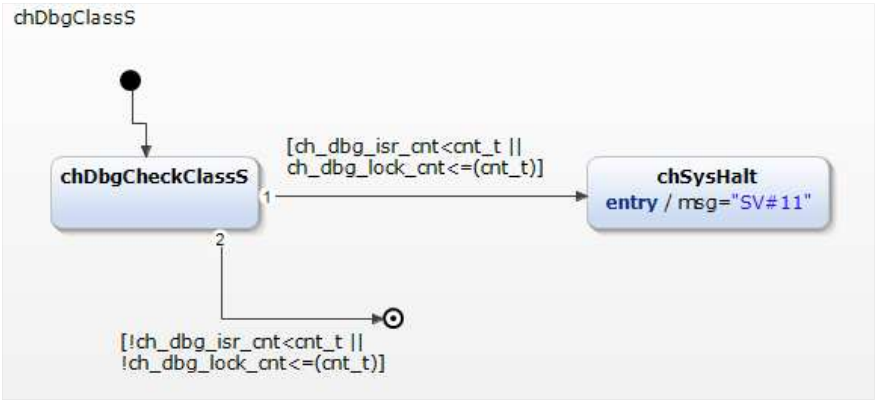


그림 3 chDbgCheckClassS에 대한 설계 명세

chDbgCheck() 기능은 '자가진단'을 수행하는 안전기능으로 생성한 뮤텍스에 대해서 Null 포인터 참조를 확인한다. 본 기능은 해당 함수에 전달된 포인터를 확인함으로써 Null 포인터 역참조를 방지하며, 전달된 포인터가 Null일 경우에는 사전에 정의된 예외처리 절차를 수행한다.

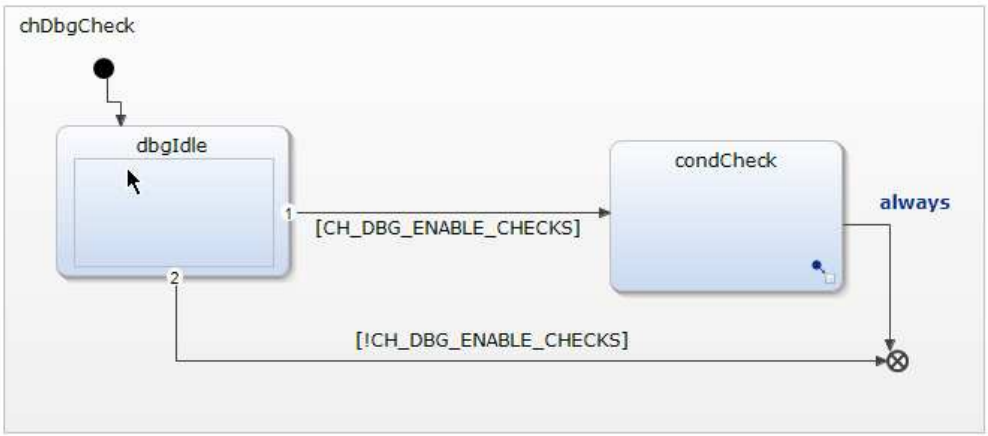


그림 4 chDbgCheck에 대한 설계 명세

chDbgAssert() 기능은 '자가진단'을 수행하는 안전기능으로 원하지 않는 값의 불일치를 확인한다. 본 기능은 여러 가지 값의 일치성을 확인하는데, 첫째로 뮤텍스의 '중복 소유'가 허락된 상태에서 뮤텍스 카운터 값의 음수 여부를 확인한다. 두 번째로는 스레드가 뮤텍스에 대해 오퍼레이션 수행 시, 소유자가 맞는지 확인한다. 만약 원하지 않는 값의 불일치가 발생할 경우, 사전에 정의된 예외처리 절차를 수행한다.

chSysHalt() 기능은 '자가진단'을 수행하는 안전기능으로 진단된 고장에 대해서 예외처리를 한다. 본 기능은 chDbg(디버깅 클래스) 자가진단 함수에 의해 호출되며, 고장이 발생한 원인을 식별하고 커널을 종료시킨다.

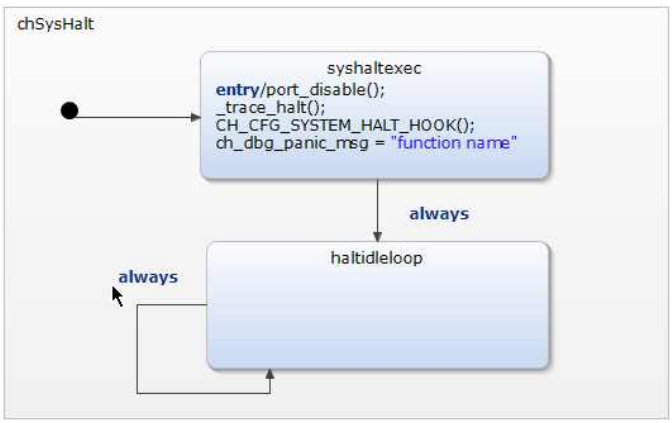


그림 5 chSysHalt에 대한 설계 명세

2) 안전기능: 도메인 분리

도메인 분리는 서로 접근하지 않아야 하는 영역에 대한 접근을 사전에 차단하려는 방법으로, 시간적 분리과 공간적 분리를 할 수 있다. chMtxLock()에서는 시간적 분리 방법으로 **chSchGoSleepS()**을 이용해 뮤텍스에 접근하는 스레드의 실행 시간을 제한한다. 또한, 뮤텍스를 원하는 스레드를 **chSchReadyI()**를 이용해 새로운 우선순위와 함께 실행 대기큐에 넣음으로써 무한정 기다리는 것을 제한한다.

3) 안전기능: 스케줄링 예측 가능

안전한 실행을 위해서 스케줄링 예측이 가능해야 하고, 우선순위 반전문제는 태스크의 예측할 수 없는 행위를 발생시킬 수 있다. CHAOS에서는 공유 자원을 사용하는 뮤텍스에 대해서 우선순위 상속 프로토콜(Priority Inheritance Protocol)을 사용하여 우선순위 반전을 방지한다. 스레드가 이미 사용 중인 뮤텍스를 사용하고자 할 때, 이미 뮤텍스를 소유하고 있는 스레드의 우선순위를 요청 스레드의 우선순위와 같도록 우선순위를 상승시키고 요청 스레드는 뮤텍스의 사용이 끝날 때까지 큐에서 대기한다. [그림 2]는 해당 절차에 대한 설계 명세를 나타낸다.

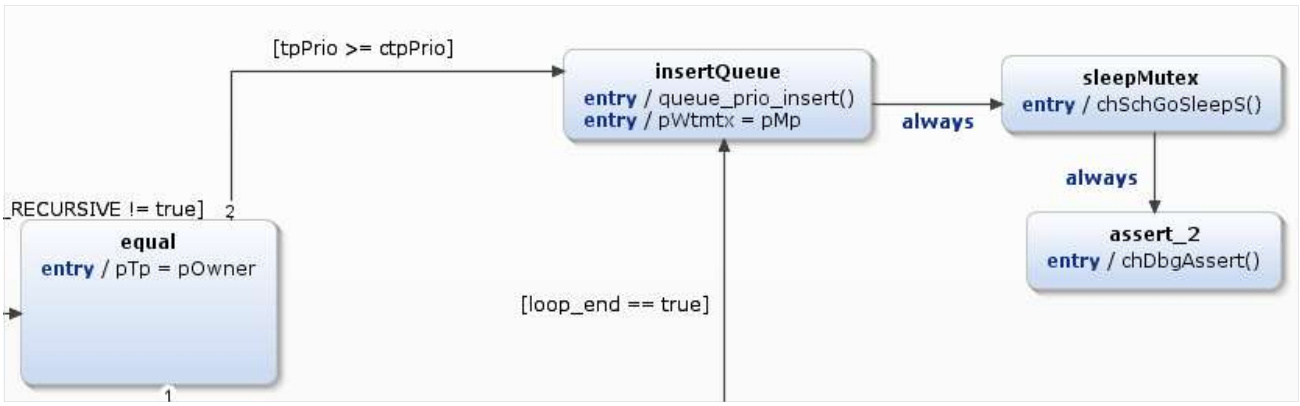


그림 6 우선순위 상속 프로토콜의 설계 명세

4) 안전기능: 안전상태 설계

안전한 커널은 실행 중 예측하지 못한 상황이 일어난다면 안전한 상태로 회복할 수 있어야 한다. 또한, 이런 상황이 발생했을 때, 다른 구성요소에 손상을 주지 않아야 하며, 신속하게 다시 시작할 수 있어야 한다. CHAOS에서는 다음과 같은 원하지 않는 상황을 막기 위해서 안전기능 sysHalt를 수행한다. [그림 5]는 해당 절차에 대한 설계 명세를 나타낸다.

가) 커널이 제어할 수 없는 상황이 발생함
나) 비정상 상태가 발생했을 때 안전한 상태로 회복하지 못함
다) 특정 요소의 안전상태 회복이 다른 요소의 고장으로 전이함
라) 특정 요소의 재시작이 다른 요소의 고장으로 전이함
마) 안전상태로 회복되는 작업보다 다른 작업의 우선순위가 더 높음

5) 안전기능: 예외상황 처리

안전한 실행을 위해서 예외상황이 올바르게 정의되어 있어야 하고, 예외상황이 발생했을 경우 처리가 가능해야 한다. CHAOS에서는 비정상적인 동작을 감지하고 방지한다. 비정상적인 동작은 하드웨어 오류가 발생하거나 소프트웨어 실행이 어려운 경우를 말하며, 이에 안전한 커널은 해당하는 예외처리를 실행한다. 여러 가지 예외상황에 대해서 dbg 클래스 함수를 통해 예외처리를 한다. [그림 3]과 [그림 4]는 해당 절차에 대한 설계 명세를 나타낸다.

chDbgCheckClassS()는 예외상황 처리를 위한 안전기능으로 함수호출 권한을 확인한다. 본 기능은 해당 함수에 대한 호출 권한이 없는 접근에 대해서는 할당된 예외처리 코드와 함께 사전에 정의된 예외처리 절차를 수행한다. chDbgCheck()는 예외상황 처리를 위한 안전기능으로 호출한 함수에 대해서 잘못된 메모리 참조를 방지하기 위한 예외처리로 사용된다. 본 기능은 해당 함수에 전달된 포인터를 확인함으로써 Null 포인터 역참조를 방지하며, 전달된 포인터가 Null일 경우에는 사전에 정의된 예외처리 절차를 수행한다. chDbgAssert()는 예외상황 처리를 위한 안전기능으로 함수의 실행을 위한 조건 만족 여부를 확인한다. 본 기능은 여러 가지 값의 일치성을 확인하는데, 만약 원하지 않는 값의 불일치가 발생할 경우, 사전에 정의된 예외처리 절차를 수행한다.

다음은 예외상황 처리 기능을 통해 방지하고자 하는 위험원들이다.

가) 정의되지 않은 예외상황이 발생함
나) 부적절한 예외상황을 정의함
다) 예외상황에 대하여 대응하지 못함
라) 예외상황에 대하여 부적절하게 대응함
마) 예외상황을 감지하지 못함

