MŰEGYETEM 1782

SZAKDOLGOZAT-FELADAT

Szentpéteri Szabolcs (HCAQUD)
villamosmérnök hallgató részére

AUTOSAR memóriastack tesztelése hibainjektálással

Egy modern gépjármű biztonsági és komfort funkcióit számos *beágyazott vezérlőegység* (ECU) támogatja. Az ezen számítógépeken futó szoftver komplexitása gyakran összemérhető a desktop alkalmazásokéval, pl. egy elektronikus kormányrendszer kb. 150 szoftverkomponensből, több ezer kapcsolatból és félmillió kódsorból áll. Annak érdekében, hogy az ECU a jármű teljes élettartamán át elviselje a fellépő *jelentős fizikai igénybevételt* (szélsőséges hőmérséklet, rázkódás, páratartalom, ingadozó tápellátás stb.), a desktop rendszerektől eltérően adattárolásra nem merevlemezeket, hanem tipikusan EEPROM vagy flash alapú tárolókat használnak. Ezen tárolók kezelőprogramjainak képesnek kell lenni a memóriát érő sérülések detektálására és javítására adatvesztés és a szolgáltatás időleges kiesése nélkül, hiszen akár biztonsági szempontból jelenős funkciók függhetnek tőle. Fontos követelmény tehát, hogy a memóriát kezelő rutinokról tapasztalati úton tudjuk bizonyítani, hogy képesek detektálni és túlélni egyes flash cellák sérülését. Ezen EEPROM vagy flash alapú tárolók kezelő szoftverei az autóiparban, a vezető autógyártók által 2002-ben életre hívott AUTOSAR konzorcium által definiált szoftver modulokból épülnek fel. A konzorcium célja az, hogy ezen szakterületi szabványokra építve specifikáljon egy (i) *alapvető szolgáltatásstruktúrát,* amely eltakarja a hardver sajátosságait és támogatja az alkalmazási szoftver hordozhatóságát (base software stack, BSW), (ii) egy *modellezési nyelvet* az ECU-kon futó alkalmazási szoftver szabványos leírására (software component template), és (iii) az alkalmazások és BSW-k ECU-n belüli és ECU-k közti *transzparens kommunikációját* lehetővé tevő elosztott, futásidejű szolgáltatást (RTE):

- A *base software stack* magában foglalja az alacsony szintű eszközmeghajtókat (pl. EEPROM és Flash driverek), az ezeket eltakaró absztrakciós rétegeket (pl. memória absztrakciós felület) és az ezekre ültetett magas szintű funkciókat (pl. perzisztens adattárolás).
- A *modellezési nyelv* lehetővé teszi, hogy precízen specifikáljuk az adattípusokat, illetve az alkalmazást alkotó komponensek interface-eit és belső felépítését.
- Az *RTE* egy generált glue kód réteg, amely eltakarja az alkalmazáskomponensek elől, hogy az általuk fogadott vagy küldött információ pontosan hogyan jut el a forrástól a célig.

A jelölt feladata egy hibainjektor szoftver kifejlesztése (egy – már létező – teszt keretrendszert felhasználva), amellyel a flash memória különböző sérüléseit lehet szimulálni, ezzel vizsgálva a flash memóriára épülő állománykezelő rendszer hibatűrő képességeit, esetleg a későbbiekben javaslatot adva a hibadetektáló és túlélő képességek javítására. A megvalósítandó tesztszoftver a *bitinverzión alapuló hibainjektálás* módszerét fogja felhasználni (melyet a későbbiekben részletezünk). A feladatot a következő lépésekben hajtsa végre:

- *A szabvány kapcsolódó részeinek megismerése:* (i) ismertesse az AUTOSAR rétegzett BSW struktúráján belül a *nem felejtő memória alacsony szintű kezeléséért (Flash Driver)* és a *nem felejtő memóriában tárolandó adatblokkok kezeléséért felelős modulok (Flash EEPROM Emulation* és *NVRAM Manager) szerepét (*gyűjtőnevükön: *Memory Stack*

Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Méréstechnika és Információs Rendszerek Tanszék
1117 Budapest, Magyar Tudósok krt. 2. I. ép. I.E.444.
Telefon: 463-2057, Fax: 463-4112
http://www.mit.bme.hu • e-mail: mitadm@mit.bme.hu

modulok*),* különös tekintettel az adatblokkok kezelését megvalósító modulra (*NVRAM Manager*), majd (ii) vázolja ezen modulok együttműködését egy olyan *forgatókönyv bemutatásával*, melyben egy (vagy több) adatblokk flash memóriába kiírásának és visszaolvasásának menetét láthatjuk.

- *Memóriastack hibatűrésének analízise:*(i) ismertesse a teljes AUTOSAR memóriastack *hibamodelljét* - az egyes szoftver rétegek hiba detektáló képességein keresztül -, ill. ismertesse, hogy a memóriastack mely hibadetektáló képességeit tudja majd ellenőrizni a megvalósítandó teszt, (ii) fogalmazzon meg követelményeket a *tesztelési eljárásokra, a teszt környezet* és a *teszt eredményeinek kiértékelésére* a konzulens segítségével.
- *A megvalósítás erőforrásigényének vizsgálata:* (i) adjon becslést, az Ön által fejlesztett teszt teljes futási idejére, (ii) adjon becslést a célprocesszorban található beágyazott flash memória várható meghibásodására vonatkozóan, (iii) ismertesse, milyen környezetben érdemes a teszteket futtatnia, hogy az költséghatékony legyen.
- *Szoftvertervezés és -megvalósítás:* (i) modellezze UML-ben a statikus felépítést és a dinamikus viselkedést, (ii) készítsen – a teszt számára kiindulási alapot szolgáltató – különböző adattartalmú, hibamentes flashmemória-képeket (flash image), melyeken a bitinverziós hibainjektálás elvégezhető, (iii) módosítsa a rendelkezésre álló Memory Stack Integration Test projekt szoftver elemeit a megvalósítandó tesztnek megfelelően, (iv) készítsen vizuálisan is könnyen értelmezhető hibariportot a teszteredményekről (pl. HTML formátumban). A feladat megvalósítása a C és JAVA nyelv ismeretére támaszkodik, így a hallgatónak ezen nyelvek ismeretét is – a feladatnak megfelelő szinten – el kell sajátítania.
- *A megvalósítás tesztelése:* A memóriastack hibatűrésének analízise alapján készítsen olyan jellegű teszteseteket – a megvalósított teszthez –, amelyekkel a teszt hibadetektáló képessége az analízisben meghatározott esetekre bizonyítható.

Tanszéki konzulens: Dr. Sujbert László, docens
Külső konzulens: Teveli Zoltán (ThyssenKrupp Presta Hungary Kft.)

Budapest, 2017. október 7.

……
Dr. Dabóczi Tamás
tanszékvezető