Járműinformatika GEIAL34J-B

Számonkérés - főbb témakörök

Alapok, bevezetés (Fóliák: JarmuInfE1.pdf)

A jármű irányító rendszerei kapcsán mi a különbség a vezérlés és a szabályozás között?

- Vezérlés – „nyílt hurkú” nincs visszacsatolás, a beavatkozás tényleges eredményének nincs hatása a beavatkozásra.
- Szabályozás – „zárt hurkú” a szabályozott rendszer tényleges állapota befolyásolja a szabályzón keresztül a beavatkozást.

Jármű elektronikus rendszerei és a környezet „onboard”, „offboard” kommunikációja.

- Mit jelentenek, mi a különbség köztük?

Informatikai feladat „online”, „offline” végrehajtása

- Mit jelentenek, mi a különbség köztük?

Önvezető autó automatizáltsági szintek (SAE 0-5)

- Mit jelentenek, mi a különbség köztük?

A jármű elektronikus rendszerei (Fóliák: JarmuInfE2.pdf)

A jármű főbb informatikai alrendszerei

- Hajtáslánc, futómű, karosszéria (komfort és passzív biztonsági), multimédia alrendszer
- Az alrendszerek főbb összetevői, elektronikus rendszerei, érzékelőik, beavatkozóik (pár példa mindegyikre)

Minek a rövidítése az „ECU”?

Az elektronikus rendszer és szoftver fejlesztés folyamata (Fóliák: JarmuInfE3.pdf)

Járműinformatikai „logikai rendszer architektúra”, „technikai architektúra”

- Mit jelentenek, mi a különbség köztük?

Az elektronikus rendszer és szoftver fejlesztés folyamatának „V-modellje”:

Lefelé: (lebontás és a legalján implementáció)

(Rendszer nézet, Rendszer fejlesztés)

- Felhasználói követelmény analízis / Logikai rendszer architektúra specifikáció
- Logikai rendszer architektúra analízis / Technikai architektúra specifikáció

(Komponens nézet, Szoftver fejlesztés)

- Szoftver követelmény analízis / Szoftver architektúra specifikáció

- Szoftver komponens specifikáció / Szoftver komponens tervezés és implementáció (a legalja, itt készülnek el ténylegesen a komponensek)

Felfelé (tesztelés és integráció és a tetején a kész rendszer)

(Komponens nézet, Szoftver fejlesztés)

- Szoftver komponensek tesztelése
- Szoftver komponensek integrálása (összeépítése)
- Szoftver integrációs tesztek

(Rendszer nézet, Rendszer fejlesztés)

- Rendszer komponensek integrálása (összeépítése)
- Rendszer integrációs tesztek
- Kalibráció (a valódi rendszer tényleges paramétereinek beállítása)
- Elfogadhatósági teszt / Rendszer teszt (a legteteje, a teljes rendszer kész)

A rendszer és szoftver fejlesztést támogató folyamatok:

- Követelmény menedzsment (követelmények azonosítása és dokumentációja)
- Konfiguráció menedzsment (hibakezelés, változtatási igény kezelés, verzió követés, archiválás)
- Projekt menedzsment
- Beszállító menedzsment
- Minőség biztosítás

Rendszermodellezés (Fóliák: JarmuInfE4.pdf)

Mi a célja a rendszermodellnek?

- leírja a rendszer logikai és fizikai tulajdonságait
- a specifikációt készítők és a tervezők bemutathatják elképzeléseiket
- a kommunikáció megfigyelhetővé kiértékelhetővé válik

Mik a modellezési folyamat főbb lépései?

- a lényeges rendszer elemek és kapcsolataik meghatározása
- majd egy jól definiált formában leírása

Mi a célja a modellezési nyelveknek:

- jól érthető, világos (pl. grafikus) leírásmód
- pontosság (tiszta szemantika)
- mindenre kiterjedő
- futtatható (tesztelhető)

A modell alapú szoftver fejlesztés kapcsán mit értünk „software in the loop” és „hardware in the loop” tesztelésen? Mi a különbség köztük?

A modell alapú szoftver fejlesztés folyamatában mit értünk „ellenőrzésen” és „igazoláson”? Mi a különbség a céljaikban?

Vezérlő és ellenőrző rendszerek (Fóliák: JarmuInfE5.pdf)

Mik a szabályzás logikai rendszer architektúra blokk diagramjának főbb elemei?

- Alapjel (Referencia) ennek az állításával szeretnénk módosítani a szabályozott szakasz állapotát (az állapotot a szabályozott jelek mérésével az érzékelő figyeli)
- Különbségképző: (alapjel)-(ellenőrző jel)=(rendelkező jel („hibajel”))
- Szabályzó: rendelkező jelből beavatkozó jelet készít
- A beavatkozó szerv (Actuator) a beavatkozó jel alapján változtatja a szabályozott szakasz paramétereit
- Szabályozott szakasz (Plant), ez a rendszer, ennek a működését szeretnénk szabályozni
- Szabályozott (irányított) jel, a szabályozott szakasz működését jellemzi, ezt szeretnénk valamilyen értékre állítani
- Érzékelő (Sensor), méri a szabályozott jelet és ellenőrző jelként (visszacsatolt jel) továbbítja a különbségképzőbe.

Mik a PID szabályzó főbb összetevői?

- arányos (Proportional) tag
- integráló (Integral) tag
- differenciáló (Differential) tag

A vezérlő és ellenőrző rendszerek kapcsán mit értünk azon, hogy

- egy rendszer „diszkrét idejű”
- egy jel „diszkrét értékű”
- egy jel „diszkrét idejű” és „diszkrét értékű”

Beágyazott rendszerek (Fóliák: JarmuInfE6.pdf)

Mit nevezünk beágyazott rendszernek?

Mi a lényeges különbség egy „beágyazott rendszer” és egy általános célú számítógép feladatai és felépítése között?

Milyen központi vezérlőegység típusok fordulhatnak elő beágyazott rendszerekben? (Pár példa, nagyjából mik azok)

Milyen főbb egységekből épül fel egy mikrokontroller alapú vezérlő egység?

Mikrokontrollerek perifériák:

- Mi a funkciója az analóg-digitális átalakítónak (ADC)?
- Mi a funkciója a digitális-analóg átalakítónak (DAC)?

- Mi a funkciója a kommunikációs interfészeknek?

Mi a lényeges különbség a csak olvasható (ROM), valamint az írható és olvasható memóriák működése és felhasználása között?

Mik a „nem felejtő” memóriák működésének főbb jellemzői?

Mik egy utasítás végrehajtásának főbb lépései?

- Fetch: az utasítás betöltődik a memóriából az utasításregiszterbe (lehívás)
- Decode: utasítás értelmezése (azonosítja az utasítást)
- Execute: a dekódolt utasítás végrehajtása az adatokon, majd az eredmény visszaírása a memóriába (WriteBack)

Mi a lényeges különbség a Neuman és a Harvard architektúrájú processzor memória kezelése között?

- Neumann: közös program és adat memória ugyanazon a sínen.
- Harvard: külön program és adat memória két független a sínen

Az utasítás készleteket alapján a mikrokontrollerek milyen két fő csoportba sorolhatók?

- Csökkentett utasításkészletű eszköz (Reduced Instruction Set Computer (RISC))
- Összetett utasításkészlettel rendelkező eszköz (Complex Instruction Set Computer (CISC))

Mik a főbb jellemzőik a RISC mikrokontrollereknek?

Mik a főbb jellemzőik a CISC mikrokontrollereknek?

Autóipari kommunikációs rendszerek (Fóliák: JarmuInfE7.pdf)

Mik a központosított szabályozó rendszerek előnyei, hátrányai?

Mik az elosztott szabályozó rendszerek előnyei?

Járműinformatikai szempontból mi az előnye a kommunikációs buszok alkalmazásának az egységek közötti független kapcsolatok építéséhez képest?

Kommunikációs szempontból mi a lényeges különbség a buszon használt „recesszív” és „domináns” bit működése között? (Pl. I2C, CAN, LIN)

Milyen bit fog megjelenni a CAN buszon, ha egy időben az egyik CAN mester „recesszív” egy másik pedig „domináns” bit küld?

Hogyan oldható fel a busz hozzáférés (arbitráció) versenyhelyzete adatveszteség nélkül „recesszív” és „domináns” bitek használatával (pl. CAN arbitráció)?

Mit nevezünk „adatátviteli sebességnek”?

- Időegység alatt átvitt bitek száma [bit/secundum]

Mit nevezünk „jelzés sebességnek”?

- Időegység alatt átvitt jelzések száma [baud]

Mik a SAE (Society of Automotive Engineers) autóipari adatátviteli alkalmazás osztályok?

- A osztályú alkalmazások: a karosszéria-elektronika olyan kevésbé intelligens eszközeinek kommunikációja, mint pl. kapcsolók, fényszórók, tükörbeállítás, ülés pozicionálás, elektromos ablakemelő, központi zár. (≤10Kbit/s)
- B osztályú alkalmazások: magasabb szintű információk cseréjére szolgálnak, pl. információátvitel a műszerfal, vagy a légkondicionáló vezérlése felé. (≤125Kbit/s)
- C osztályú alkalmazások: valós idejű (real time critical) információátvitel, 1-10ms-os ciklusidővel, és 1ms-nál rövidebb késleltetéssel. (≤1Mbit/s)
- D osztályú alkalmazások: ebben az esetben nagyobb mennyiségű adat továbbítására van szükség, az adatblokkok mérete elérheti a néhány kbyte-ot is, mint pl. a hangrendszer, telefon vagy GPS (Global Positioning System) kommunikációja során. (1Mbit/s–10Mbit/s)

Egy járműinformatikai rendszerben tipikusan milyen célokra használják a LIN buszt? Mi ennek az oka?

Mit jelent az, hogy a LIN „egy-mester” (Single master) protokoll?

Hogyan jelzi a LIN mester az üzenet küldésének kezdetét?

- 1 Start bit (domináns), 13 domináns bit, majd 1 Stop bit (recesszív)

Hogyan szinkronizálja a LIN mester a szolgák óráját?

- Az üzenetkezdet jelzést követően 0x55-öt (55 hexa) küld (01010101)

Mik a CAN protokoll főbb jellemzői?

Mit jelent az, hogy a CAN „több-mester” (Multimaster) protokoll?

Mit jelent a CAN protokoll „üzenetközpontúsága”?

Hogyan működik CAN protokoll „nem-destruktív” arbitrációs mechanizmusa?

Mit jelent az, hogy a CAN protokoll „üzenetszórást” (Broadcast) alkalmaz?

Mit jelent az, hogy a CAN protokoll „eseményvezéreltsége”?

Mit jelent az, hogy a CAN protokoll „rugalmassága”?

- A csomópontokat dinamikusan rákapcsolhatjuk, illetve leválaszthatjuk a buszról anélkül, hogy a többi csomópont kommunikációját zavarnánk

Hogyan szolgálja a CAN protokoll a valós idejű alkalmazásokat?

- Az üzenetek rövidek, a késleltetési idő maximálva van, az arbitráció pedig gyors

Hogyan szolgálja a CAN protokoll a megbízhatóságot?

- 15 bites CRC (Cyclic Redundancy Check) hibajelző kódolás
- Nem rendszeres hibák helyreállítása a hibás üzenetek automatikus újraküldésével
- Ismétlődő hibák kiküszöbölése a hibás csomópont kikapcsolásával, ami determinisztikussá teszi a rendszer esetleges hibák utáni helyreállásának idejét

- Az elektromágneses interferenciákra alacsony az érzékenysége (csavart érpár).
- A rendszer garantálja, hogy a küldő-csomópont által elküldött adatok megegyeznek a fogadó-csomópontok által fogadott adatokkal.

Hogyan történik a CAN protokollban a nyugtázás?

- Az üzenetben lévő nyugtázó mezőben (Acknowledgement field) a vevő jelzi a küldő csomópontnak, hogy az üzenetet hibátlanul megkapta

Mi szolgálja a CAN protokollban a konzisztens üzenetátvitelt?

- Konzisztencia: minden egyes csomópont megkapja ugyanazt az információt ugyanabban az időpillanatban, vagy egyik sem
- Ha legalább egy vevő hibát észlel a fogadás során, akkor egy hibaüzenettel (Error frame – domináns bitek) rögtön megszakítja az átvitelt, és jelzi a többi fogadó állomásnak, hogy hagyják figyelmen kívül az üzenetet, a küldőnek pedig, hogy küldje el azt ismét.

A CAN protokoll a hibakezelése kapcsán egy CAN csomópont milyen események hatására kerül:

„aktív” hibaállapotból „passzív” hibaállapotba?

- Vételi hibák száma > 127, vagy küldési hibák száma > 127

„passzív” hibaállapotból „aktív” hibaállapotba?

- Vételi hibák száma < 127, és küldési hibák száma < 127

„passzív” hibaállapotból „leválasztott” hibaállapotba?

- Küldési hibák száma > 255

„leválasztott” hibaállapotból „aktív” hibaállapotba?

- Újraindítás, vagy 128-szor előforduló 11 recesszív bit detektálás (nyugta határoló bit+üzenet vége mező+üzenet utáni szünet)

A CAN protokoll a hibakezelése kapcsán egy csomópont működése szempontjából mi a különbség ha az „aktív”, „passzív”, vagy „leválasztott” hibaállapotban van?

- „aktív” hibaállapot: Hiba érzékelése esetén domináns bitekkel jelzi a hibát. Ez a normális alapműködés. Ha úgy romlik el, hogy mindig hibát érzékel, akkor ezzel megakasztja a többiek forgalmát.
- „passzív” hibaállapot: Úgy működik, mint az „aktív” hibaállapotban, csak hiba érzékelése esetén passzív bitekkel jelzi a hibát (azaz nem jelez a többieknek hibát). Ezt akkor teszi, ha megnő a hibaszám és gyanús, hogy rossz a vétele. Így a többiek vételét nem rontja el, de elvész a konzisztencia (az ő részéről).
- „leválasztott” hibaállapot: Ha túl sok a hiba, akkor mindenből kiszáll, nem ad, nem vesz semmit, csak arra vár, hogy a busz működése kellő ideig helyreálljon.