

Beágyazott és Ambiens Rendszerek (vimiac06)

Témakörök

1. Beágyazott rendszerek általános felépítése: tipikus érzékelők (nagy komplexitású eszköz választása vs. egyedi fejlesztés előnye/hátránya), jelkondicionálási feladatok, ADC és DAC típusok és felhasználási területeik. Tipikus feldolgozó egységek (μ P, μ C, DSP, FPGA) egymáshoz képesti viszonya az eszközök teljesítménye, tervezési idő és megoldható feladat komplexitása szempontjából (milyen feladatot melyik eszközzel oldanánk meg).
2. Giant Gecko (EFM32-STK3700) fejlesztői kártya: tipikus szenzorok (fényerősségmérő, LC fémérzékelő, érintésérzékelő) ismertetése (működési elv, milyen tervezési irányelvek vannak, milyen μ C perifériát használnak). Órajel-menedzsment alapelve a Giant gecko processzoron, hogyan szolgálja a fogyasztás csökkentését.
3. Fejlesztői környezetek és fordítók: fordítás folyamata (.c $\rightarrow$ obj $\rightarrow$ link). Néhány tipikus fordítási csatoló ismerése (-D, -O0...-O3, -o, -mcpu, -I, -Wall, -std). make program és makefile formátum ismerése: makefile szabályok (cél, előfeltétel, utasítás szintaktikája, egyszerűbb minták értelmezése)
4. Szoftverarchitektúrák: szoftverarchitektúra tervezésének szempontjai. A tipikus szoftverarchitektúrák jellegzetes felépítése (mintakód, ütemezési diagram, válaszdíő...) és tulajdonságaik: ciklikus programszervezés (és aletei), megszakítással kiegészített, ütemezett függvények.
5. Megszakításkezelés: Megszakítás inicializálásának és kezelésének általános elvei. A megszakítás-engedélyezés általános hierarchiája, vektoros megszakításkezelés elve, Cortex-M3, ATmega128 és ADSP-BF357 megszakítás-kezelő felépítése (elvi szinten, bonyolult ábrák nem kellenek). C nyelvű megszakítás-kezelő függvények megadásának néhány módja (Cortex-M3, ATmega128, ADSP-BF537, ADSP21364). Hogyan kerülnek a függvények a vektortáblába, és hogyan jelezzük a fordítónak, hogy ők megszakításfüggvények?
6. Megosztott változók: alapprobléma megfogalmazása, milyen változók esetén kritikus, mintapélda, megoldási lehetőségek. Kettős bufferelés. Dinamikus memóriahasználat beágyazott rendszerekben (malloc függvény). Stack túlszordulás. Robusztus programozás (timeout, biztonságos kódolás, strukturált programszervezés, típushasználat, redundancia).
7. Speciális C nyelvi elemek: inline függvények, bitmező struktúrák, union adattípus, regisztertömbök strukturált kezelése (memóriaterületek közvetlen elérése), __attribute__ (pl.: interrupt, always_inline, weak) és #pragma (pl.: once, interrupt, align) kulcsszavak, idiom recognition, #define specialítások. Példák.
8. Hordozható kód: mit jelent, miért fontos, integer adattípusok (stdint.h), könyvtári függvények (mire kell figyelni, blokkoló/nem blokkoló). Virtualizáció: 1-es és 2-es működési modell, hypervisor feladata, vele szemben támasztott követelmények.
8. Hibakeresés. Beágyazott rendszer debuggolási sajátosságai. Hibakeresésre használt eszközök: debugger, tracer, profiler, watchpoint. JTAG port segítségével felépített debug rendszer blokkdiagramja, JTAG segítségével megvalósított tipikus feladatok. Tipikus alapszintű debug lehetőségek (GPIO, UART: printf átirányítása). Futási idő mérésének módszerei.