Miskolci Egyetem
Gépészmérnöki és Informatikai Kar
Automatizálási és Kommunikáció-Technológiai Tanszék

2019/2020. tanév
I. félév

Beágyazott Rendszerek és Architektúrák GEVAU 218M c. tantárgy

Előadásának és gyakorlatainak ütemterve

MSC szintű mérnökinformatikus hallgatók részére.

Tárgynév:	**Beágyazott Rendszerek**.					
Rövid név:	BeáR_MSC			*Kód*	**GEVAU 218M**	
Angol név:	Embedded Systems					
Tanszék:	Villamosmérnöki Intézet, Automatizálási és Infokommunikációs Intézeti Tanszék					
Tárgyfelelős:	Dr. Vásárhelyi József egyetemi docens, tel: (46) 565 111 /1753 vajo@mazsola.iit.uni-miskolc.hu					
Előtanulmányok:	nincs			*Kódja:*	GEVAU	
Kredit:		*Követelmény:*		gyakorlati jegy		
Heti óraszámok	*Előadás:*	2	*Gyakorlat:*		*Labor:*	2
Oktatási cél:	A villamosmérnöki ismeretekhez a digitális-technikai alapok elsajátítása.					
Tárgy tartalom:	A tantárgy célja bemutatni a beágyazott rendszerek tervezési platformját képező rendszerelemeket, kibővíti az alapvető általános szoftveres ismereteket a beágyazott rendszerek szoftvertervezési ismeretekkel (esemény és ideve-zérelt programozás, tervezési minták, szoftverarchitektúrák, modell alapú szoftverfejlesztés). Ismerteti a legelterjedtebb rendszer architektúrákat.					
Irodalom:	A.N.Sloss, D. Symes, C. Wriht: *ARM System Developer's Guide, Designing and Optimizing System Software*, Elsevier, ISBN: 978-1-55860-874-0					
Ajánlott Irodalom	1. Labrossse J.J et all: *Embedded Software know it all*, Newnes, ISBN 978-07506-8582-5, 2008, pp.770. 2. Labrosse J.J: *MicroC/OS-II The real-time kernel*, CMP Books, ISBN 1-57820-103-9, 2002, pp. 606. 3. Scott Hauck, Andree Dehon ed. *Reconfigurable Computing The Theory and Practice of FPGA-Based Computation*, Elsvier, ISBN 978-0-12-370522-8, 2008, pp. 945					

Mintatantervi elhelyezkedés szakok szerint				
Szak	*Szakirány/sáv*	*Tantervi modul-tantervi kód*	*Mintatantervi félév*	*Választhatóság*
Villamosmérnöki Szak	minden	MV	1	kötelező

Jellemző oktatási módok	
Oktatási nyelv:	Magyar, angol
Előadás:	Tábla + számítógépes vetítés
Gyakorlat:	
Labor:	laboratórium gyakorlat egyéni feladatokkal

FÉLÉVKÖZI ZH BEÁGYAZOTT RENDSZEREK ÉS ARCHITEKTÚRÁK

SOFTWARE DESIGN BASICS

1. Melyek a beágyazott rendszerek tervezési paraméterei?

Méret, végrehajtási ciklus, egyszeri tervezési költség, piacra kerülési idő, disszipált teljesítmény, karbantarthatóság, stb.

2. Mit jelent a tervezési paraméterek optimálása?

A tervezési paraméterek egymás ellen hatása miatt ha valamelyik paraméter értékét csökkentjük (pl. méret), akkor a másik paraméter növekszik (pl disszipált teljesítmény növekedés ha nem gondoskodunk hűtésről).

3. Mit értünk a beágyazott rendszer végrehajtási ciklusa alatt?

Azt a szükséges időt ami a feladat egyszeri végrehajtásához szükséges.

4. Mi az utasítás ciklus?

Az utatsítás ciklus áll: fecth, decode, execute

5. Miért melegedik a magas frekvencián működő processzor?

Mivel a processzor tranzisztorai kapcsoló üzemben működnek ezért nagy a disszipált teljesítmény, ai 50% kitöltésű jelnél hővé alakul.

6. Adott egy automata mosógép (beágyazott rendszer). Határozza, meg milyen feladatokat kell ellátnia és melyek lehetnek a bemeneti és kimeneti mennyiségek milyen perifériákat kell tartalmaznia ha mikroporceszoros vezérlést alkalmazunk?

   Bemeneti mennyiségek: a szennyes ruha tömege, a víz mennyisége (áztatás, mosás, öblítés), programozási gombok, program kiválasztó, hőmérő, stb.
   kimeneti mennyiségek: kijelző, motor, vízmelegítő, relék;
   A mikroporceszoros vezérlést alkalmazva konfiguráljuk a megfelelő protokat, egyéb periféria: időzítő.

7. Rajzolja meg a konkurens hardware és software műveletek időbeni lefolyását.
   Útmutatás: A konkurens műveletek esetében ugyanaz a hardver/szoftver végzi a műveleteket, de időosztással

CORTEX-M0+ PROCESSOR CORE

8. A Cortex-M0+ processzor mag regiszterei közül melyek az általános célú regiszterek?

R0-R12

9. A Cortex-M0+ processzor mag regiszterei közül melyik a link regiszter?

R14

10. A Cortex-M0+ processzor mag regiszterei közül melyik a verem mutató (stack pointer)?

R13

11. Milyen méretű adatokat kezel a load és store utasítás?

    8, 16, 32, 64 bites adatokat kezel a load/store;

FÉLÉV VÉGI VIZSGA BEÁGYAZOTT RENDSZEREK ÉS ARCHITEKTÚRÁK

SOFTWARE DESIGN BASICS

1. Melyek a beágyazott rendszerek tervezési paraméterei?

Méret, végrehajtási ciklus, egyszeri tervezési költség, piacra kerülési idő, disszipált teljesítmény, karbantarthatóság, stb.

2. Mit jelent a tervezési paraméterek optimálása?

A tervezési paraméterek egymás ellen hatása miatt ha valamelyik paraméter értékét csökkentjük (pl. méret), akkor a másik paraméter növekszik (pl disszipált teljesítmény növekedés ha nem gondoskodunk hűtésről).

3. Mit értünk a beágyazott rendszer végrehajtási ciklusa alatt?

Azt a szükséges időt ami a feladat egyszeri végrehajtásához szükséges.

4. Mi az utasítás ciklus?

Az utatsítá ciklus áll: fecth, decode, execute

5. Miért melegedik a magas frekvencián működő processzor?

Mivel a processzor tranzisztorai kapcsoló üzemben működnek ezért nagy a disszipált teljesítmény, ai 50% kitöltésű jelnél hővé alakul.

6. Adott egy automata mosógép (beágyazott rendszer). Határozza, meg milyen feladatokat kell ellátnia és melyek lehetnek a bemeneti és kimeneti mennyiségek milyen perifériákat kell tartalmaznia ha mikroporceszoros vezérlést alkalmazunk?

   Bemeneti mennyiségek: a szennyes ruha tömege, a víz mennyisége (áztatás, mosás, öblítés), programozási gombok, program kiválasztó, hőmérő, stb.
   kimeneti mennyiségek: kijelző, motor, vízmelegítő, relék;
   A mikroporceszoros vezérlést alkalmazva konfiguráljuk a megfelelő protokat, egyéb periféria: időzítő.

7. Rajzolja meg a konkurens hardware és software műveletek időbeni lefolyását.
   Útmutatás: A konkurens műveletek esetében ugyanaz a hardver/szoftver végzi a műveleteket, de időosztással

CORTEX-M0+ PROCESSOR CORE

8. A Cortex-M0+ processzor mag regiszterei közül melyek az általános célú regiszterek?

R0-R12

9. A Cortex-M0+ processzor mag regiszterei közül melyik a link regiszter?

R14

10. A Cortex-M0+ processzor mag regiszterei közül melyik a verem mutató (stack pointer)?

R13

11. Milyen méretű adatokat kezel a load és store utasítás?

    8, 16, 32, 64 bites adatokat kezel a load/store;

Évközi feladatok, zárthelyik:	1
Lezárási feltételek:	A Tanulmányi és Vizsgaszabályzat szerint. Az Előadások legalább 60%-ának látogatása, a gyakorlatok legalább 75%-ának teljesítése. Gyakorlatokon aktív részvétel; az előírt feladatok teljesítése; a két évközi zárthelyi dolgozat eredményes megírása (legalább elégséges); az évközi (házi) feladatok elfogadható szintű elkészítése. A lezáráshoz írásbeli- és szóbeli vizsgát kell tenni. Az évközi teljesítmény40%-a és az aláírás 60% összege a tárgyat lezáró jegy.

Ütemterv

1.	EA: Bevezetés a beágyazott rendszerek tervezésébe; Tervezési paraméterek versenyhelyzete. Gyak: Ismerkedés az ARM fejlesztőrendszerével és az oktatási kártyával
2.	EA: RISC processzor tervezési módszerek, ARM tervezési módszerek Gyak: Terv létrehozása, program fejlesztés, fordítás, letöltés, stb.
3.	EA: ARM processzor ismeretek Gyak: Terv hibakeresés, követés futtatás
4.	EA: ARM processzor utasítás készlet és THUMB utasítás készlet ismertetése Gyak: Egyéni feladat beadás 45 héten
5.	EA: Hatékony C program fejlesztése Gyak: Egyéni feladat beadás 45 héten
6.	EA: ARM ASM program írása és optimalizálása Gyak: Egyéni feladat. beadási határidő: 47. hét
7.	EA: Tervezési szempontok PLD áramköröknél. Időzítési modell.
8.	EA: Digitális jelfeldolgozás ARM processzorokkal Gyak: Egyéni feladat. beadási határidő: 47. hét
9.	EA: Megszakításkezelés, Firmware Gyak: Egyéni feladat. beadási határidő: 47. hét
10.	EA: ARM perifériák I. Gyak: Egyéni feladat. beadási határidő: 47. hét
11.	Zárthelyi dolgozat Gyak:Feladat beadási
12.	EA: ARM perifériák II Gyak: Egyéni feladat II . beadási határidő: 49. hét
13.	EA: Rendszer a chipen megoldások ARM processzorokkal Gyak: Feladat beadás, Gyakorlat pótlás
14.	EA: Memória kezelés, Beágyazott operációs rendszerek Gyak: Feladat beadás, Gyakorlat pótlás
15.	EA: Többmagos architektúrák. Gyak: Gyakorlatok pótlása

Miskolc, 2019. szeptember. 1.

Dr. Trohák Attila
tanszékvezető egyetemi docens

Dr. Vásárhelyi József
egyetemi docens

12. A Cortex-M0+ processzor mag regiszterei közül melyik a programszámláló (program counter)?
    R15
13. Milyen méretű adatokat kezel a push és pop utasítás?
    32/64 (a processzor függvénye)
14. Milyen módon jelenik meg a 0x3b85fea3 szó a memóriában “little-endian” és “big-endian” memória rendszerben?

Cím	Little-Endian	Big-Endian
n	a3	3b
n+1	fe	85
n+2	85	fe
n+3	3b	a3

15. Milyen típusú az STM F4 sorozatú ARM Cortex processzor: little-endian vagy big-endian?
    Little endian
16. Milyen irányban nő az STM F4 veremtár mutató (R15) tartalma (csökken vagy nő)?
    A veremtár mutató csökken minden egyes tárolással (pusH) 4 bájttal, és nő 4 bájttal pop utasítás esetén.

12. A Cortex-M0+ processzor mag regiszterei közül melyik a programszámláló (program counter)?
    R15
13. Milyen méretű adatokat kezel a push és pop utasítás?
    32/64 (a processzor függvénye)
14. Milyen módon jelenik meg a 0x3b85fea3 szó a memóriában “little-endian” és “big-endian” memória rendszerben?

Cím	Little-Endian	Big-Endian
n	a3	3b
n+1	fe	85
n+2	85	fe
n+3	3b	a3

15. Milyen típusú az STM F4 sorozatú ARM Cortex processzor: little-endian vagy big-endian?
    Little endian
16. Milyen irányban nő az STM F4 veremtár mutató (R15) tartalma (csökken vagy nő)?
    A veremtár mutató csökken minden egyes tárolással (pusH) 4 bájttal, és nő 4 bájttal pop utasítás esetén.
17. Milyen regisztereket ment el az ARM kivételkezeléskor?
    R0,R1, R2, R3, R4, R12, LR, PC, xPSR

18. Feltételezve, hogy az alapállapotban SP = 0x0000_2034. Milyen értéke less az SP-nek a következő utasítás végrehajtása után POP {r0-r4,PC}?

    4 regiszter * 4 byte mindenik = 16 bytes (0x10)
    SP=0x0000_2034 + 0x10 = 0x0000_2044

19. Milyen utasításokat használunk szubrutinhíváshoz? Válasz: BL, BX
20. Milyen utasítást használunk szubrutinból való visszatéréshez?? Válasz: BX

21. C AS IMPLEMENTED IN ASSEMBLY LANGUAGE

22. Adott a fordító által egy adott C fügvényből létrehozott aszambler kód. Magyarázza meg a kijelölt utasításokat és írja le a regiszterek tartalmát.

```
                find_in_list PROC
;;;20    int find_in_list(int key) {
00002e  4601        MOV     r1,r0
;;;21        unsigned int i;
;;;22        for (i=0; i<NUM_ELS; i++) {
;;;23            if (list[i] == key)
000030  4a0b        LDR     r2,|L1.96|
000032  2000        MOVS    r0,#0           ;22
                |L1.52|
000034  0083        LSLS    r3,r0,#2
000036  58d3        LDR     r3,[r2,r3]
000038  428b        CMP     r3,r1
00003a  d004        BEQ     |L1.70|
00003c  1c40        ADDS    r0,r0,#1        ;22
00003e  280a        CMP     r0,#0xa         ;22
```

```
000040  d3f8            BCC     |L1.52|
;;;24                   return i;
;;;25       }
;;;26       return -1;
000042  2000            MOVS    r0,#0
000044  43c0            MVNS    r0,r0
              |L1.70|
;;;27     }
000046  4770            BX      lr
;;;28
                    ENDP
```

a) LDR R2, |L1.96|

23. LDR loads betölti az R2 regiszterbe a tömb kezdő címét, amelyet az |L1.96| cím tartalmaz..

a) MOVS r0,#0

MOVS i változó alapállapotba állítás i=0.

MEGSZAKÍTÁSOK ÉS KIVÉTEL KEZELÉS

24. Mi a link regiszter szerepe?

Tartalmazza a visszatérési címet.

25. Hogyan tilthatjuk le a kivételek kiszolgálását a konfigurálható prioritássokkal?

Beállítjuk a PRIMASK regiszter's PM bitjét.

26. Sorolja fel a memória tartalomban és a regiszterekben történt változásokat egy kivételkezelés kéréskor de még a kivételkezelés első utasításának végrehajtása előtt.

Veremtár tartalma csökken 32 bájttal, mivel a processzor menti az XPSR, visszatérési cím LR, R12, R3, R2, R1, R0 kerül a verem memóriába.
A processzor átkapcsol handler/privileged üzembe
A kivétel kezdőcíme bekerül a PC
A LR-be a visszatérési cím kerül EXC_RETURN code
Az xPSR értéke megváltozik: IPSR-be kerül a kivétel kezelés (megszakítás) száma

27. Adott az alábbi memória és regiszter tartalom. A POP {r5,r4, pc} utasítás végrehajtása a kivételkezelésből való visszatérés előtt történik meg. Mi lesz a következő utasítás címe, amit a processzor végrehajt? Magyarázza meg miért pont az az utasítás?
28. Given the following memory and register contents, the instruction POP {r5,r4, pc} executes in order to return from this interrupt handler. What is the address of the instruction which will execute next? Explain why.

Register	**Contents**
R13 (SP)	0x2000_0000

Register	**Contents**
R13 (SP)	0x2000_0000
Memory Address	**Contents**
0x2000_0000	0x0000_5133
0x2000_0004	0x4500_ffff
0x2000_0008	0xffff_fff9
0x2000_000c	0x0000_4800
0x2000_0010	0x0000_5000
0x2000_0014	0x0000_5080
0x2000_0018	0x0000_0009
0x2000_001c	0x0000_0087
0x2000_0020	0x0000_1840
0x2000_0024	0x0000_2380
0x2000_0028	0x0100_0000

Az címen 0x0000_2380 lévő usatítást hajta végre a processzor will execute next. Előbb az r5, r4 pc kikerül a veremből. POP 0xffff_fff9 érték (the EXC_RETURN code) thread üzemből visszatér a processzor és MSP regisztert használja. Majd POP az elmentett 8 regiszter visszaállítása beleértve a PC (a címről: 0x20000_0024). Tehát a PC=0x0000_2380

29. ÁTLALÁNOS CÉLÚ I/O

30. Hogyan változtatják a port adatot az alábbi regiszterek?
    a) PDOR

    PDOR regiszterben megváltozik a megfelelő prot bitje egy kiírt X érték esetében

    b) PTOR

    1 írása PTOR komplementálja a megfelelő port bitjét 0 írása nincs hatással.

c) PSOR

1 írása a PSOR adott bitjére beállítja a port megfelelő bitjét 1-re

d) PCOR

1 írása a PCOR adott bitjére törli a port megfelelő bitjét.

31. Írjon C programot az A port 7-es bitjének bemenetként és az 1 bit kimenetként történő konfigurálására

```
PORTC->PCR[7] = PORT_PCR_MUX(1);
PTC->PDDR |= 1<<7;
PORTA->PCR[1] = PORT_PCR_MUX(1);
PTA->PDDR &= ~(1<<1);
```

32. Írjon C programciklust ami az A port 1-es bitjét vezérli a 7-es bit komplemensével

```
while (1) {
    if (PTC->PDIR & (1<<7))
        PTA->PCOR = 1 << 1;
    else
        PTA->PSOR = 1 << 1;
}
```