Institutionen för data- och informationsteknik
CHALMERS TEKNISKA HÖGSKOLA

Hemtentamen - Exempel

Maskinorienterad programmering

avsedd att ge exempel på hur hemtentamen är utformad, det är inte meningen att ni ska arbeta med uppgifterna (använd gamla salstentamina i stället).

Examinator
Roger Johansson

Kontaktpersoner under tentamen:
Roger Johansson, epost (Canvas)

Tillåtna hjälpmedel

Alla hjälpmedel är tillåtna så länge uppgifterna löses på individuell basis utan att konsultera någon annan än examinator under skrivningstiden.

Lösningar
Anslås senast dagen efter tentamen via kursens hemsida.

Granskning
Tid och plats anges på kursens hemsida.

Allmänt
Svar kan avges på svenska eller engelska.

Tänk på att disponera din tid väl.

Börja med att läsa igenom alla uppgiftstexter.

Försök avge lösningsförslag på samtliga uppgifter.

Betygsättning
Maximal poäng är 60 och tentamenspoäng ger betyg enligt: (EDA/DAT):

$30p \leq$ betyg 3 $< 40p \leq$ betyg 4 $< 50p \leq$ betyg 5.

respektive (DIT):
$30p \leq$ betyg G $< 45p \leq$ VG

Som **slutbetyg** på kursen ges det lägre av betyg från hemtentamen respektive betygsbedömd laboration.

Uppgift 1

Ett program, givet i C, ska översättas till ARMv6 assemblerspråk.

```
unsigned int meanvalue( int n, unsigned short *v )
{
   unsigned int sum; int i;
   if( n <= 0 )
      return 0xFFFF;
   sum = 0;
   for( i = 0; i < n; i++ )
   {
      sum += (unsigned int) v[i];
   }
   return sum/n;
}

unsigned short vector[]={12,800,649,100,10};
unsigned int sum;

void main(void)
{
  sum = meanvalue( sizeof(vector)/sizeof(unsigned short), vector);
  for( int i = 0; i < sizeof(vector); i++ )
    vector[i] = sum;
}
```

Speciella anvisningar:

- Assemblerkoden ska kommenteras på sådant sätt att det tydligt framgår vilka delar av C-koden som översatts och genererat just dessa sekvenser.
- Användningen av register (registerallokering) ska beskrivas.
- Antag att instruktionsuppsättningen även innehåller en divisionsinstruktion: `DIVU Rx,Rx,Ry`, som utför heltalsdivision (tal utan tecken) Rx ← Ry/Rx.

Bedömningskriterier:

- Lösningen kan ge maximalt 15p.
- En lösning som saknar, eller bara innehåller mycket bristfälliga, kommentarer kan ge maximalt 5 p.
- Din lösning ska vara tydlig, fullständig och följa anvisningarna ovan, annars görs poängavdrag.
- Kodningen ska följa de råd och anvisningar som getts under kursen.

Uppgift 2

En periferienhet med ett 8 bitars gränssnitt ska anslutas till ett MD407-system.

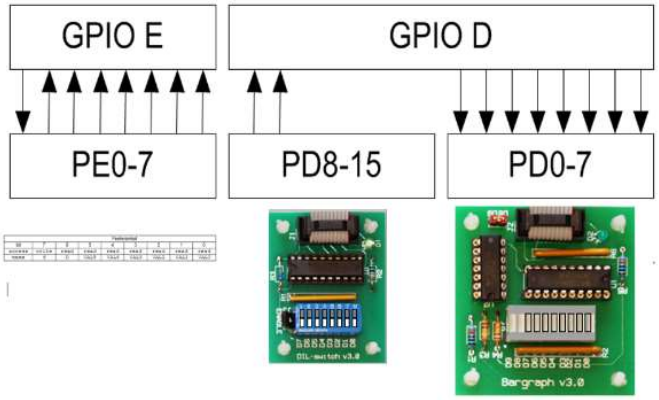
Periferienhet								
bit	7	6	5	4	3	2	1	0
access	read/ write	read	read	read	read	read	read	read
namn	S	D	VAL5	VAL4	VAL3	VAL2	VAL1	VAL0

Från RESET är bit S=0, medan bit D och VAL-bitarna är odefinierade.

Periferienheten styrs och fungerar på följande sätt:

- Programvaran sätter bit S till 1 för att starta operationen. Kretsen behöver nu 100 ms innan statusbiten D är giltig. Då resultatet är färdigt sätts bit D till 1 av periferienheten. Nu är värdet VAL giltigt och kan läsas av från dessa bitar i gränssnittet. Ett giltigt värde är alltid skilt från 0, observera dock att då D är 0 kan VAL innehålla ett värde skilt från 0 trots att detta är ogiltigt.
- Programvaran ska nu läsa av värdet VAL.
- Slutligen återställs periferienheten genom att bit S sätts till 0 av programvaran. Periferikretsen nollställer då bit D vilket indikerar att värdet VAL inte längre är giltigt.
- Periferikretsen är nu klar för nästa operation.

Konstruera en applikation som kontinuerligt läser ett giltigt värde från periferikretsen och skriver detta till en diodramp. Dessutom ska 2 bitar läsas från en strömställare och skrivas till diodrampen. Port E (0-7) ska användas för periferikretsen, medan port D (bit 15 och bit 14) ska användas för strömställare och port D (0-7) till diodrampen. Eftersom de två bitarna från strömställaren ska uppdateras kontinuerligt utan onödig fördröjning kan inte fördröjningsfunktionen utformas som en blockerande funktion.



Speciella anvisningar:

Följande dellösningar ska ingå i applikationen:

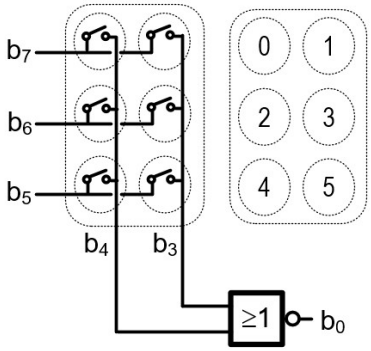
- En initieringsfunktion `init_app`, där du visar hur portarna D och E ska initieras, alla utgångar ska vara *push-pull*, alla ingångar ska vara *pull-down*. IO-pinnar som inte används kan anses vara odefinierade.
- Använd SYSTICK för att implementera en icke blockerande fördröjning.
- Föreslå och specificera C-funktioner och data och beskriv dessa funktioners gränssnitt (parametrar/returvärden). Beskrivningen ska vara kortfattad men tillräckligt detaljerad för att utgöra grund för en implementering.
- Implementera systemet enligt din beskrivning, i programspråket C.

Bedömningskriterier:

- Lösningen kan ge maximalt 15p.
- En lösning som saknar beskrivning/kommentarer, eller bara innehåller mycket bristfälliga, kommentarer kan ge maximalt 5 p.
- Din lösning ska vara tydlig, fullständig och följa anvisningarna ovan, annars görs poängavdrag.
- Kodningen ska följa de råd och exempel som getts under kursen, speciellt att du använt lämpliga makrodefinitioner för GPIO-portarnas adresser.

Uppgift 3

Ett tangentbord för inmatning av sex olika tecken ska konstrueras. Sex stycken återfjädrande omkopplare ansluts därför till port E hos *MD407*, enligt figuren till höger. En nedtryckt tangent ska representera ett binärt värde 0-5.

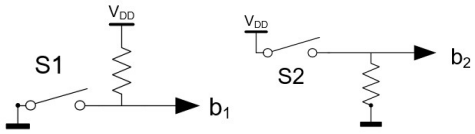


Kretsen kan också användas för att känna av en tangentnedtryckning med hjälp av avbrott. För detta kopplas kolumnerna till en NOR-grind, vars utgång är kopplad till bit 0 hos port E.

Anm: Vi påminner om NOR-grindens funktion: utsignalen är '1' så länge alla ingångar är '0'. Då någon av ingångarna blir '1' blir utsignalen '0'.

Nedtryckta tangenter kan detekteras genom att '1' skrivs till någon av bit 7, bit 6 eller bit 5, därefter avläses bit 4 respektive bit 3. För att detta ska vara tillförlitligt måste dessa bitars ingångar förses med "*pull-down*" samtidigt som utgångarna ska vara "*push-pull*".

Utöver tangentbordet ansluts också två återfjädrande strömställare, S1 ("Unlock keyboard") med "pull-up"-resistor och S2 ("Lock keyboard") med "pull-down"-resistor enligt figuren till höger. Dessa används för att generera ytterligare avbrott.



Visningsenhet

Utöver detta ansluts en utmatningsenhet för visning av två hexadecimala siffror till port E b8-b15.

Speciella anvisningar:

- Bit 0, bit 1 och bit 2 ska kopplas till avbrottssystemet hos *MD407* via EXTI-modulen. Avbrott ska ske då omkopplarna sluts, betrakta kopplingarna i figurerna ovan och fundera på vilken flank avbrotten ska ske på. De tre avbrottsfunktionerna namnges `KeybIRQ`, `S1IRQ`, `S2IRQ`. Vektortabellen börjar på adress 0x2001C000
- En tangentbordsfunktion returnerar tangentbordets tillstånd. Detta kan vara 0-5 om en tangent tryckts ned, eller 0xF om tangentbordet är låst.
- En tangentnedtryckning ska resultera i ett avbrott via b0. Strömbrytare S2 används för att 'låsa' tangentbordet med ett avbrott via b2. I låst tillstånd ignoreras alla tangentnedtryckningar. Strömbrytare S1 används för att 'låsa upp' tangentbordet med ett avbrott via b1. I upplåst tillstånd registreras och sparas den senast nedtryckta tangentens kod. Visa de tre avbrottsfunktionerna `KeybIRQ`, `S1IRQ`, `S2IRQ` och ange dessas avbrottsvektorer.
- En enkel utmatningsenhet för visning av två hexadecimala siffror, i form av en `char` används som indikator och är ansluten till port E, bit 8-15. Ett huvudprogram skriver kontinuerligt ut information från tangentbordet. Om tangentbordet är upplåst ska senast nedtryckta tangent (0-5) visas, om tangentbordet är låst ska siffran 0xF visas. Tangentbordet ska initialt vara låst.
- Beskriv, med ord, ett program med de delar (funktioner och data) som realiserar datainsamlingssystemet. Beskrivningen ska vara kortfattad men tillräckligt detaljerad för att utgöra grund för en implementering.
- Implementera datainsamlingssystemet enligt din beskrivning, i programspråket C.

Bedömningskriterier:

- Lösningen kan ge maximalt 15p.
- En lösning som saknar beskrivning/kommentarer, eller bara innehåller mycket bristfälliga, kommentarer kan ge maximalt 5 p.
- Din lösning ska vara tydlig, fullständig och följa anvisningarna ovan, annars görs poängavdrag.
- Kodningen ska följa de råd och exempel som getts under kursen, speciellt att du använt lämpliga makrodefinitioner för GPIO-portarnas adresser.

Uppgift 4

Antag att en dator (inte nödvändigtvis en MD407) används för enkel tidtagning vid en idrottstävling. Till datorn finns kopplat två givare och en klockkrets. (Dessutom finns en sifferindikator, men den behöver inte programmeras i denna uppgift.) Den första sensorn känner när startskottet går och den andra när den tävlande passerar mållinjen.

De båda givarna är kopplade till *samma* 16-bitars styrregister. Detta ligger på adressen 0x1234. Styrregistret aktiveras och inaktiveras genom att bit nr 0 i det sätts till 1 resp. 0. Om registret är inaktiverat påverkas det inte av inkommande signaler, men om det är aktiverat gäller följande:

- När en signal kommer från någon av de två sensorerna sätts bit nr 7 i registret till 1.
- Om man har satt bit nr 6 i registret till 1 genereras då även en avbrottssignal.
- Styrregistret skall återställas efter ett avbrott genom att man sätter bit 7 till 0.

Klockkretsen är kopplad till ett annat 16-bitars styrregister, vilket ligger på adressen 0x1230. Detta styrregister har samma konfiguration och fungerar på samma sätt som styrregistret för givarna med skillnaden att de inkommande signalerna kommer från klockkretsen istället för givarna. Klockkretsen genererar 1000 signaler per sekund.

Uppgiften är att skriva en C-applikation som gör en tidsmätning. När programmet startar skall det visa tiden 0 på en display och vänta tills startskottet går. När detta sker skall klockan aktiveras och tiden skall visas fortlöpande på displayen. Displayen skall visa tiden uttryckt i hundradels sekunder och den visade tiden skall uppdateras hundra gånger per sekund. När den tävlande passerar mållinjen skall klockan stoppas och sluttiden visas konstant på displayen. Programmet behöver bara klara en tidsmätning. (Vill man göra en ny får man starta om programmet genom att trycka på reset-knappen.) Du får anta att tiden för instruktionsexekvering är försumbar.

Speciella anvisningar:

- Du får förutsätta att det finns en färdigskriven C-funktion `void display( unsigned long )`. När den anropas visar den parameterns värde på en sifferindikator.
- Du får också förutsätta att avbrottsmekanismer är initierade och att en funktion:
  `void clocktrap(void)` anropas vid avbrott från klockan samt en funktion:
  `void sensortrap(void)` anropas vid avbrott från någon sensor
  du måste dock skriva dessa båda funktioner.
- Föreslå och specificera resterande C-funktioner och data och beskriv dessa funktioners gränssnitt (parametrar/returvärden). Beskrivningen ska vara kortfattad men tillräckligt detaljerad för att utgöra grund för en implementering.
- Implementera systemet enligt din beskrivning, i programspråket C.

Bedömningskriterier:

- Lösningen kan ge maximalt 15p.
- För full poäng krävs att dina lösningar är tydliga, fullständiga och att du följt anvisningar ovan.
- En lösning som saknar beskrivning/kommentarer, eller bara innehåller mycket bristfälliga, kommentarer kan ge maximalt 5 p.
- Kodningen ska följa de råd och exempel som getts under kursen.