EKSAMENSOPPGAVE (Deleksamen 1)

Emne: **IRE12011 / Programmering og mikrokontrollere**
Lærer: **Åge T Johansen**

Grupper: **Alle i 1. årskurs Elektro**	Dato: **16.12.2015**	Tid: **09.00 – 12.00**
Antall oppgavesider: **5** (forsiden inkludert)	Antall vedleggsider: **7** **Vedlegg 1:** 12 flervalgspørsmål (3 sider) **Vedlegg 2:** MBED oversikt (4 sider)	
Sensurfrist: 15.01.2015		
Hjelpemidler: **Lærebøker:** "Mike McGrath: C Programming", "Mike McGrath: C++ Programming" **PC** med blant annet Code::Blocks finnes tilgjengelig på eksamensrommet. **Kalkulator**		
KANDIDATEN MÅ SELV KONTROLLERE AT OPPGAVESETTET ER FULLSTENDIG Alle deloppgaver teller likt ved bedømming, bortsett fra i flervalgoppgaven, der 3 spørsmål teller som 1 deloppgave. PC uten nettilknytning kan benyttes som hjelp for å løse oppgavene. Alle besvarelser skal leveres på papir (utleverte eksamensark med gule og blå kopier). Ønsker du å levere programmer du har laget på utlevert minnepenn, kan du gjøre dette i en konvolutt der du påfører eksamensnummeret. Ved tvil/klage er det papirkopien som gjelder.		

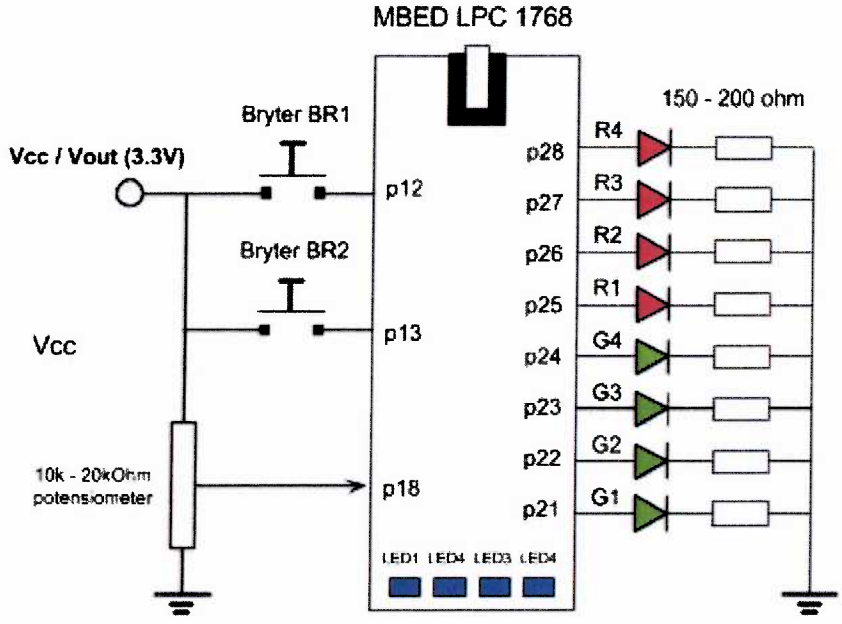
Oppgave 1

Vedlegg 1 inneholder en flervalgoppgave med **12** spørsmål. Du skal kun velge **ett** svaralternativ på hvert spørsmål. Velg alternativet du mener **passer best**. Presenter svarene med samme referanser som i vedlegget, med nummer på spørsmålet og bokstav på svaralternativet. Ordne besvarelsen i en tabell som vist nedenfor. (NB! Svaralternativene som er skrevet inn, er kun eksempler.)

Spørsmål	Svar	Spørsmål	Svar	Spørsmål	Svar
1	*C*	5	...	9	...
2	*A*	6	...	10	...
3	*E*	7	...	11	...
4	...	8	...	12	...

Oppgave 2

Figuren nedenfor viser en MBED-modul med tilkoblinger. Modulen er også forbundet med en PC via en USB-kabel, men dette er ikke vist i figuren. Alle deloppgaver i oppgave 2 er relatert til denne figuren.



a) Hvilken diode / eventuelt hvilke dioder lyser **etter** at dette programmet er kjørt, og hvilken verdi skrives til en tilkoblet PC med *Tera Term*? Forklar **kort** hvordan programmet virker og navngi dioden(e) som lyser i henhold til figuren.

main.cpp

```
#include "mbed.h"

Serial pc(USBTX, USBRX);
BusOut leds(p21, p22, p23, p24, p25, p26, p27, p28);

int main()
{
    int n;
    for (n = 5; n < 17; n++)
    {
        leds = n;
    }
    pc.printf("%d\n", n);
    return 0;
}
```

b) Samme spørsmål som i deloppgave a), men med følgende program:

main.cpp

```
#include "mbed.h"
Serial pc(USBTX, USBRX);
BusOut leds(p21, p22, p23, p24, p25, p26, p27, p28);

int main()
{
    for (int n = -300; n < 300; n++)
    {
        if (n > 127) {
            leds = n;
            pc.printf("%d\n", (n / 200) * 200);
            break;
        }
    }
    return 0;
}
```

c) Skriv et fullstendig program som skal utføre følgende oppgave:

- ❑ Diode R4 skal alltid lyse.
- ❑ Diodene G1 og LED1 skal begge blinke med en hastighet på ½ Hz, men slik at G1 vil lyse i ½ sekund og LED1 i 1 sekund per blink.

d) Skriv et nytt program:

- ❑ Diodene G1, G2, G3, G4, R1, R2, R3, R4 skal lyse i sekvens slik at bare en diode lyser om gangen. (Løpelys.)
- ❑ Hver diode skal lyse 1 sekund om gangen hvis bryter 1 er trykket, hvis ikke skal diodene lyse 2 sekunder om gangen.
- ❑ Når sekvensen er ferdig, starter den på nytt igjen.

e) Skriv et nytt program. Potensiometeret kan kobles mot en analog inngang og leses da av som en flyttallsverdi mellom 0.0 og 1.0.

- ❑ LED1 skal lyse når spenningen inn mindre enn 1V.
- ❑ G1 skal lyse når spenningen er mellom 1 og 2V.
- ❑ R1 skal lyse når spenningen er størren enn 2V.

Oppgave 3

a) Funksjonen vist nedenfor er ment å beregne arealet av et trapes, der ***a*** og ***b*** er de parallelle sidene og ***h*** er avstanden mellom dem. Dessverre har det sneket seg inn noen syntaksfeil. Finn og beskriv feilene. Referer til linjenumrene.

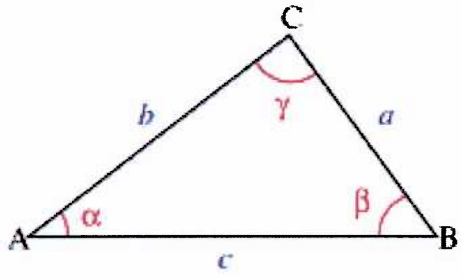
```
43  float trapez(float a, float b, float h)
44      float area;
45      area = [a + b] * h;
46      area = area : 2.0;
47      return area;
48  }
```

I den videre delen av denne oppgaven gjelder figuren nedenfor, som viser en trekant med sider og vinkler. Sammenhengen mellom sider og vinkler er gitt av følgende formel:

$$c^2 = a^2 + b^2 - 2ab \cdot cos(\gamma)$$

Det forutsettes at *Tera Term*, *putty* eller et lignende program kjører på en PC som er koblet til MBED via USB-forbindelsen.

b) Skriv et program med *main*() der brukeren blir bedt om å taste inn lengdene til alle 3 sidene i en trekant - a, b, og c. Programmet skal beregne ***vinkelen*** - γ - mellom de 2 første sidene som er skrevet inn - dvs. *a* og *b* i figuren. Til slutt skal den beregnede vinkelen skrives ut.



For å beregne vinkelen, skal du i denne deloppgaven benytte en ***ferdigskrevet*** funksjon, *vinkel*. Funksjonen *vinkel* har en funksjonsprototype og en spesifikasjon som vist nedenfor:

```
float vinkel(float aa, float bb, float cc);
/*  aa, bb, cc: Kjente sider
    Returverdi: Den beregnede verdien til vinkelen
    mellom sidene aa og bb i radianer. */
```

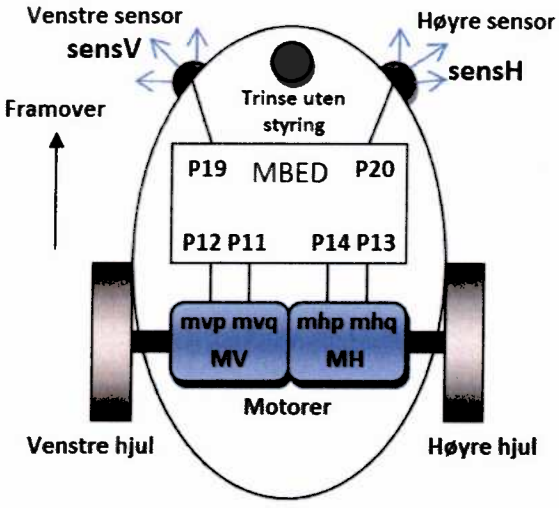
*NB! Du skal **ikke** utvikle funksjonen i denne deloppgaven, bare benytte den i programmet.*

c) Skriv koden for funksjonen *vinkel* som ble beskrevet i deloppgave b). Som en forenkling kan du anta at vinkelen ligger i 1. kvadrant.

NB! Den matematiske standardfunksjonen - acos() - vil være aktuell å benytte i denne oppgaven.
NB! Husk ingen innlesning fra tastatur eller utskrift til skjerm fra funksjonen. Nødvendige verdier for beregningene fås gjennom funksjonsargumentene.

Oppgave 4

Figuren til høyre viser en skisse (sett ovenfra) av et lite robot-kjøretøy med 2 motorer (**MV** og **MH**) med hvert sitt hjul og en **MBED**-modul til å styre bevegelsene. En fritt opplagret trinse foran gjør at kjøretøyet holder balansen. Motorens driverkretser er tilpasset de digitale utgangene til **MBED** og kalles henholdsvis **mvp**, **mvq**, **mhp og mhq**. Se figur for tilkoblinger.



Se tabellen nedenfor for sammenhengen mellom bevegelse og motorpådrag. Det er i oppsettet ingen turtallsregulering av motorene (kun av/på).

*NB! Selv om de ikke er aktuelle i denne oppgaven, er det i figuren også vist to analoge optiske sensorer (**sensV** og **sensH**) som er plassert framme på hver side av kjøretøyet. Se bort fra disse her.*

Forutsett at roboten kan bevege seg framover med en hastighet på ***10 cm per sekund***, og at den vil kunne dreie med en vinkelhastighet på ***10 grader per sekund***.

FUNKSJON	Motorkontroller			
	mvp	mvq	mhp	mhq
STOPP	0	0	0	0
FRAMOVER	1	0	1	0
BAKOVER	0	1	0	1
DREI VENSTRE	0	1	1	0
DREI HØYRE	1	0	0	1

a) Skriv et komplett ***mbed***-program med *main*() som får roboten til å bevege seg 3 meter rett fram for så å dreie 60 grader mot venstre og kjøre 1 meter i denne retningen før den stanser. (*Det kan lønne seg å organisere motorstyringen som et BusOut-objekt.*)

b) Skriv en generell funksjon, ***robotkontroll***, for å styre roboten i en gitt retning i en valgt tid. (*Funksjonen er tiltenkt å kunne benyttes i deloppgave **a)** for å få en enklere løsning her.*)

Funksjonsprototypen til denne funksjonen er:

```
void robotkontroll(int bevegelse, float tid);
```

bevegelse :styrer bevegelsen etter følgende regler:
0 - stopp, 1 - framover, 2 - bakover, 3 drei venstre, 4 - drei høyre.
Tallene representerer verdien til parameteren ***bevegelse***.

tid : bestemmer hvor lenge aktiviteten skal pågå (i antall sekunder).

VEDLEGG 1

	Spørsmål		**Svaralternativer**
1	Ved programmering i C++ står begrepet vektor (*vector*) for:	A	Alternativt navn på datatypen *double*
		B	Samme som *array* generelt
		C	*Array* av koordinater
		D	C++ utvidelse for å kunne gjøre geometriske beregninger
		E	*Arrays* som inneholder tekst
		F	C++ klasse for å danne arraylignende strukturer med utvidet funksjonalitet
2	I et C/C++-program må man "si i fra" hvor data kan plasseres ved å benytte egendefinerte navn. En slik navngitt datalagringsplass kalles:	A	class
		B	adresse
		C	konstant
		D	funksjon
		E	struct
		F	Ingen av alternativene ovenfor
3	Overloading er et begrep som i C++ blant annet angir at:	A	en beregningen har ført til overflow
		B	CPU-en er overopphetet
		C	flere ulike funksjoner kan ha samme navn
		D	en gammel verdi i en variabel er skrevet over med en ny verdi
		E	en divisjon med 0
		F	en regneoperasjon med inkompatible datastørrelser
4	I C++ benyttes try-catch konstruksjonen for å:	A	generere et tilfeldig tall med prøving og feiling
		B	fange opp runtime-feil i en blokk og behandle disse
		C	fange en nøyaktig frekvens
		D	fange høyeste verdi i en rekke av beregninger
		E	fange syntaksfeil ved kompilering
		F	lage grafiske dialogvinduer

	Spørsmål		**Svaralternativer**
5	Hva skrives ut her? `printf("xxx%6.4fxxx", 6.4);`	A	`xxx%6.4xxx`
		B	`xxx6.4000xxx`
		C	`xxx 6.4xxx`
		D	`xxx000006.4xxx`
		E	`6.4000`
		F	Ingen av alternativene ovenfor
6	Følgende deklarasjon er gitt: `short int w;` Hva er største og miste verdi variabelen ***w*** kan ha? (Anta at en ***short int*** har 16 bit.)	A	-2^{16}, $+2^{16}$
		B	-2^{15}, $+2^{15}$
		C	0, 2^{16}
		D	-2^{15}, $+(2^{15}-1)$
		E	$-(2^{15}-1)$, $+(2^{15}-1)$
		F	Ingen av alternativene ovenfor
7	I C++ er en klasse (class):	A	en utvidelse av *struct* der også funksjoner kan legges inn som medlemmer
		B	alternativ til *arrays*
		C	klassifisering av funksjoner som for eksempel *int* eller *void*
		D	en slags *union* der første datamedlem overstyrer de andre
		E	en måte å skille mellom *løkker* og *tester*
		F	Ingen av alternativene ovenfor
8	Hvilken verdi har variabelen ***x*** etter at denne programlinjen er utført? `int x = (1>0) * 3 + (1<0);`	A	0
		B	1
		C	2
		D	3
		E	4
		F	Ingen av alternativene ovenfor

	Spørsmål		**Svaralternativer**
9	En datatype som kan benyttes når en variabel skal kunne inneholde tall med desimaler er:	A	long int
		B	double
		C	short int
		D	char
		E	unsigned char
		F	unsigned long int
10	En datatype som kan benyttes når en variabel skal kunne inneholde et heltall med verdier i intervallet [-1000, +1000]:	A	int
		B	unsigned long int
		C	char
		D	unsigned char
		E	Alle alternativene A- D
		F	Ingen av alternativene ovenfor

11	Hvilke er de 3 siste tallene som skives ut når programutsnittet nedenfor utføres:
	`int p = 10000;`
	`while(p >= 50) {`
	`  printf("%d ", p / 2);`
	`  p--;`
	`}`
		A	48 49 50

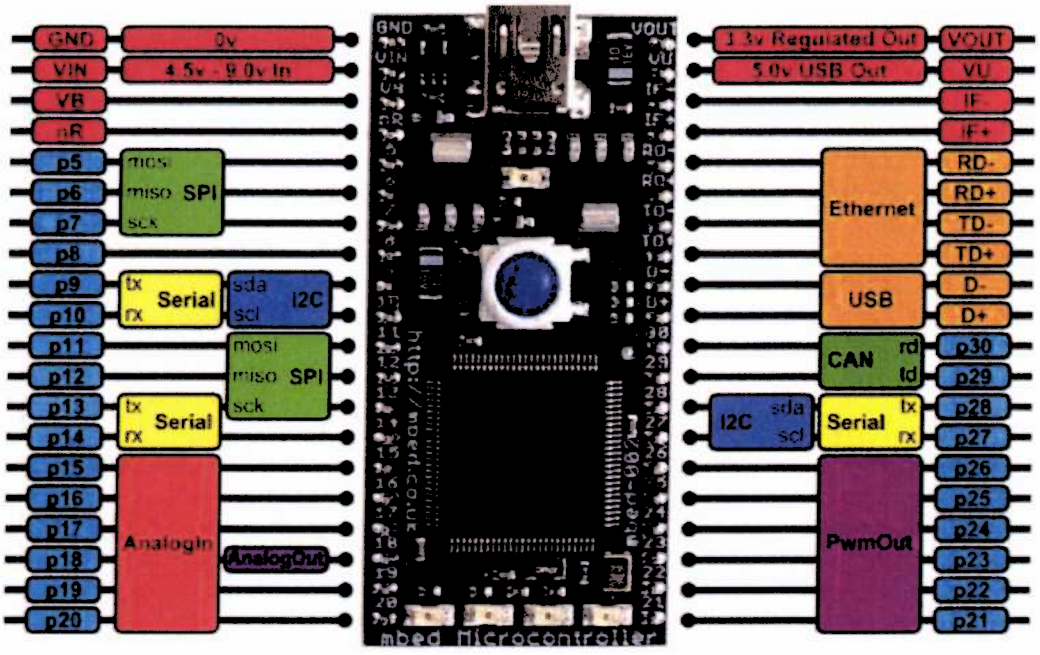
		B	26 25 25
		C	25 24 25
		D	25 25 25
		E	27 24 24
		F	Ingen av alternativene ovenfor

12	Hva blir skrevet ut etter at disse tre setningene er utført i et C++ program?
	`int a = 23;`
	`float b = 3;`
	`std::cout << a/2 << " + " << b*1.25;`
		A	a/2 + b*1.25

		B	12 + 4
		C	11 + 3.75
		D	11 + 3
		E	11.5 + 3.75
		F	Ingen av alternativene ovenfor

VEDLEGG 2

Oversikt over ofte benyttede ressurser for ***mbed***, som beskrevet i "http://mbed.org/handbook". (Den engelske teksten er uforandret, men formatet kan være redigert.)



DigitalOut

A digital output, used for setting the state of a pin.

Functions

DigitalOut	Create a DigitalOut connected to the specified pin
write	Set the output, specified as 0 or 1 (int)
read	Return the output setting, represented as 0 or 1 (int)
operator=	A shorthand for write
operator int()	A shorthand for read

Interface

The DigitalOut Interface can be used on mbed pins p5-p30, and also on-board LED1-LED4 .

The DigitalOut Interface can be used to set the state of the output pin, and also read back the current output state. Set the DigitalOut to zero to turn it off, or 1 to turn it on.

Details

The pin output is 0v and 3.3v (0 and 1), and can source or sink a maximum of 40mA.

DigitalIn

A digital input, used for reading the state of a pin.

Functions

DigitalIn	Create a DigitalIn connected to the specified pin
read	Read the input, represented as 0 or 1 (int)
mode	Set the input pin mode
operator int()	An operator shorthand for read()

Details

The pin input is logic '0' for any voltage on the pin below 0.8v, and '1' for any voltage above 2.0v. By default, the DigitalIn is setup with an internal pull-down resistor.

BusOut

The BusOut interface is used to create a number of DigitalOut pins that can be written as one value.

Example.

```
#include "mbed.h"
BusOut myleds(LED1, LED2, LED3, LED4)

int main() {
    while(1) {
        for(int i=0; i <;16; i++) {
            myleds = i;
            wait(0.25);
        }
    }
}
```

BusIn

The BusIn interface is used to create a number of DigitalIn pins that can be read as one value. Any of the numbered mbed pins can be used as a DigitalIn in the BusIn.

Example:

```
#include "mbed.h"
BusIn nibble(p5, p6, p18, p11);

int main() {
    while(1) {
        switch(nibble) {
            case 0x3: printf("Hello!\n"); break; // p5 and p6 are 1
            case 0x8: printf("World!\n"); break; // p11 is 1
        }
    }
}
```

AnalogIn

An analog input, used for reading the voltage on a pin.

Functions

AnalogIn	Create an AnalogIn, connected to the specified pin
read	Read the input voltage, represented as a float in the range [0.0, 1.0]
read_u16	Read the input voltage, represented as an unsigned short in the range [0x0, 0xFFFF]
operator float	An operator shorthand for read()

Details

The AnalogIn Interface can be used on *mbed* pins p15-p20.

The 0.0v to 3.3v range of the AnalogIn is represented in software as a normalized floating point number from 0.0 to 1.0.

AnalogOut

An analog output, used for setting the voltage on a pin

Functions

AnalogOut	Create an AnalogOut connected to the specified pin
write	Set the output voltage, specified as a percentage (float)
write_u16	Set the output voltage, represented as an unsigned short in the range [0x0, 0xFFFF]
read	Return the current output voltage setting, measured as a percentage (float)
operator=	An operator shorthand for write()
operator float()	An operator shorthand for read()

Details

The AnalogOut Interface can be used on mbed pin p18.

The AnalogOut Interface can be used to set the voltage on the analog output pin somewhere in the range of 0.0v to 3.3v.

The 0.0v to 3.3v range of the AnalogOut can be represented in software as a normalized floating point number from 0.0 to 1.0, or directly as volts or millivolts.

wait

Generic wait functions.

Functions:

```
void wait(float s);
```

Waits for a number of seconds, with microsecond resolution (within the accuracy of single precision floating point).

Variables

s - number of seconds to wait

Timer

A general purpose timer

Example:

```
// Count the time to toggle a LED
#include "mbed.h"
Timer timer;
DigitalOut led(LED1);
int begin, end;
int main() {
    timer.start();
    begin = timer.read_us();
    led = !led;
    end = timer.read_us();
    printf("Toggle the led takes %d us", end - begin);
}
```

Functions:

void start(); //Start the timer

void stop(); // Stop the timer

void reset(); // Reset the timer to 0. If it was already counting, it will continue

float read(); // Get the time passed in seconds

int read_ms(); // Get the time passed in mili-seconds

int read_us(); // Get the time passed in micro-seconds

Ticker

The Ticker interface is used to setup a recurring interrupt to repeatedly call a function at a specified rate. Any number of Ticker objects can be created, allowing multiple outstanding interrupts at the same time. The function can be a static function, or a member function of a particular object.

A Ticker is used to call a function at a recurring interval

Functions

attach()Attach a function to be called by the Ticker, specifying the interval in seconds
Arguments: function-name, interval.

attach_us() Attach a function to be called by the Ticker, specifying the interval in micro-seconds

detach() Detach the function