MALMÖ HÖGSKOLA

Inbyggda system och signaler
Styrteknik

Inlämningsuppgift 1601e

Utlämning: 2016-12-04
Deadline: 2016-12-14

Namn: ……

Namn: ……

Gion Koch Svedberg
december 2016

Elektroniskt lås

Syfte med uppgiften är att tillämpa tillståndsbeskrivningen på programmeringen av ett elektroniskt lås, såsom de t.ex. är vanliga på dörrlås.
Tillståndsmodeller (eller modeller av tillståndsmaskin eller eng. ”finite state machines”) används som universell metod för att beskriva sekvenser av händelser, till exempel i styrsystem.
Teorin illustreras i denna laboration med en lite mer konkret uppgift där modellen består i tillämpningen av automatteorin i form av en tillståndsmaskin.

Den praktiska delen utgår från en given tillståndsmodell (FSM) av ett elektroniskt lås, (eng. ”code-lock”), och uppgiften består i att implementera tillståndsmodellen som Matlabprogram med hjälp av knappar och lysdioder.

Samma utrustning ska en vecka senare användas igen i programmeringslabbet, så riv inte sönder det ni har byggt upp!!

Anvisningar

Skriv inte ut detta dokument utan ha det öppet på datorn och besvara frågorna direkt i dokumentet. Det är viktigt att ni följer denna anvisning noggrant i rätt ordning steg för steg, annars finns det risk att ni missar viktiga delar och behöver börja från början eller att utrustning blir förstörd!!
Läs hela uppgiften innan handledningstillfällen och lös de förberedande teoretiska uppgifterna innan ni påbörjar med den praktiska delen i labbsalen!

Inlämningen av detta fullständigt ifyllda dokument samt andra filer som ni ska generera för att dokumentera vissa delar av er lösning ska ske på its learning.

Uppgiften genomförs som vanligt i par dvs. ni jobbar två och två. Vid inlämningen på Its learning och på detta dokuments titelsida anges vem som jobbat ihop.
Ni behöver dessutom en digitalkamera eller en mobiltelefon med kamera för att kunna dokumentera slutresultatet.

Krav för godkänd

- **Fullständigt ifyllt dokument med korrekta svar till alla frågor, uppladdad till its learning som pdf-fil.**
- **Fil(er) med sparade matlabsession(er), uppladdad på its learning**
- **Inspelad video av fungerande elektroniska låset enligt anvisningen i respektive uppgift, uppladdad som videofil (obs storleken av filer är begränsat på its learning) eller som länk till (personliga) youtube.**

Inlämningsuppgifter som lämnas in innan deadline blir granskade av läraren som återkopplar med skriftliga kommentarer, beroende på rapportens utförande:

- **Rapporten blir godkänd**
- **Komplettering krävs**
- **Rapporten blir underkänd**

Efter eventuell komplettering som görs under kurstillfället:

- **Rapporten blir godkänd**
- **Rapporten blir underkänd**

Inlämningsuppgifter som lämnas in efter deadline men under kurstillfället blir granskade av läraren som återkopplar med skriftliga kommentarer, eroende på rapportens utförande:

- **Rapporten blir godkänd**
- **Rapporten blir underkänd**

I händelse av underkänd rapport eller om komplettering inte inkommer under kurstillfället ges nya inlämningstillfällen i anslutning till omtentamenstillfällena. Vid dessa ges dock inga kompletteringsmöjligheter, dvs rapporten blir godkänd eller underkänd.

Lägg märke till att laborationsutrustningen normalt endast finns tillgänglig under kurstillfället!

Lyckas man inte få godkänt för alla labbar under kursens gång, måste alla labbarna göras om nästa gång kursen ges igen! Anledningen är att vi inte kan tillhandahålla utrustningen utanför kursen samt att vi kontinuerligt anpassar och förändrar labbarna i våra kurser.

Tillgängliga referenser på its learning

[1] Mathworks, Matlab primer, R2015b

Beskrivning av uppgiften

Uppgiften är att programmera Matlab så att man måste trycka på knapparna på elektroniska låset i en viss följd för att öppna låset, vilket symboliseras av att den stora lampan tänds.

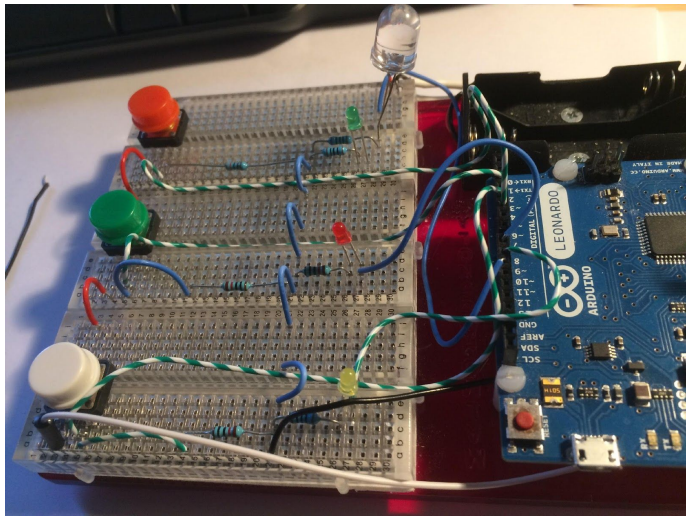


Bild av hur utrustning kan se ut som ska användas för att programmera ett elektroniskt lås. Kopplingen med tre knappar och fyra lysdioder ingår som del av uppgiften.

A) Teoretiska delen (att genomföra *innan* den praktiska delen)

A.1 Modelleringsprocessen

I föreläsningarna tittade vi närmare på systembeskrivningar och systembegreppet. Om man vill beskriva ett system, eller vissa egenskaper av ett system, behöver man aspekterna som man är intresserad av. Resultatet av modelleringen blir en som, jämfört med systemet, är en förenklad avbildning av relevanta aspekter av systemet. Modellen beror på problemområdet. Modeller är mer eller mindre användbara för ett visst syfte. Exempel av olika typer av modeller:

- **Tankemodell**
- **Fysisk modell/ funktionsmodell**
- **Arkitekturmodell**
- **Simuleringsmodell**
- **Matematisk modell**
- **…**

På föreläsningen visades en modelleringsprocess som skisserar hur man kommer från systemet till en modell med två steg: steg 1) konceptualisering och steg 2) representation, se bilden nedan. Inom styrteknik kan vi tolka det som att man till exempel kan använda systembegreppet för att avgränsa systemet. Tankemodeller, som konceptualiserar en dynamisk modell, kan beskrivas med hjälp av tillståndsmaskiner (FSM). Går vi ett steg vidare kan själva modellbygge i form av ett inbyggt system med hårdvara och mjukvara också anses som en form av representation.

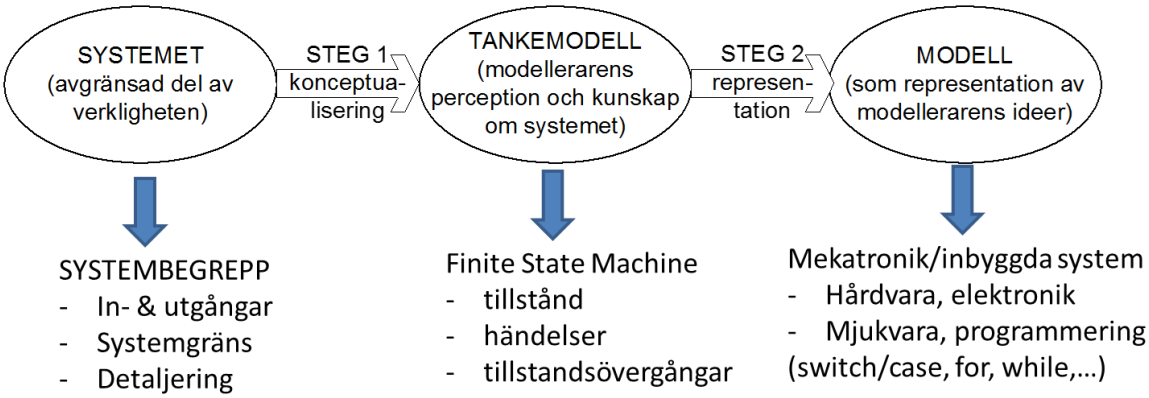


Bild: Modelleringsprocessen

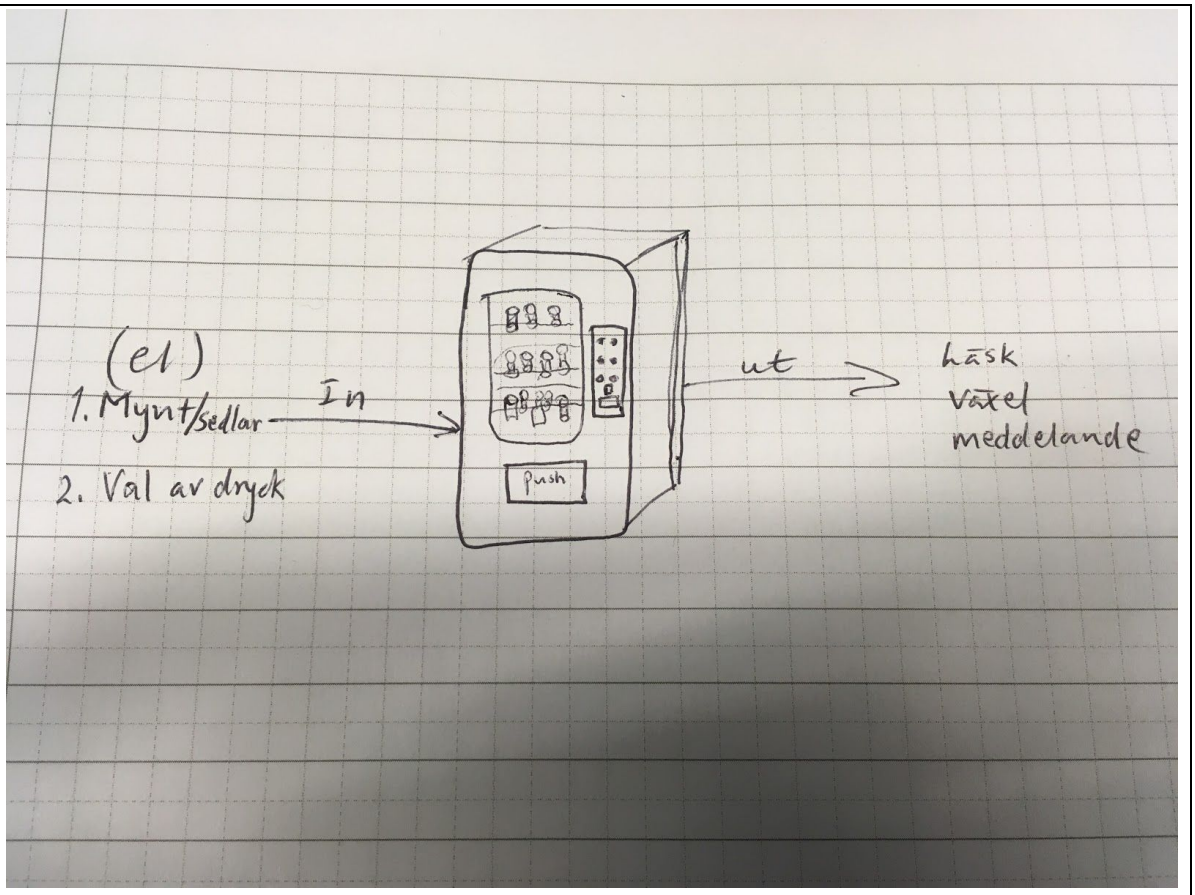
Bild: Läskautomat

För att själv testa modelleringsprocessen ska ni i uppgiften nedan tillämpa den på ett konkret exempel. Modellen som ska tas fram ska vara en som beskriver systemet som en automat. För att detta känns användbart ska systemet faktiskt vara en . Vi tar därför prototypen av en automat: !

Uppgiften nedan börjar med att beskriva läskautomaten som ett system, dvs som en avgränsad del av verkligheten. Genom konceptualiseringen ska man sedan ta fram en tankemodell. Det är framför allt automatdelen som vi vill modellera. Tankemodellen ska därför strukturera upp vilka tillstånd, och tillståndsövergångar (händelser) som modellen ska innehålla. Nästa steg blir att representera denna tankemodell som tillståndsmodell i form av en switch/case programmering.

A.1.1 Beskriv läskautomaten som ett system, enligt systembegreppet.

Kopiera in en ritning (går bra med infogad kort på whiteboard eller papper)



In och utgångar :
in:Pengar, knappval, (el)
ut: läsk, växel och meddelande

Förklara med egna ord vad man ser i ritningen:
Genom inmatning av pengar och val av dryck kan vi få ut från automaten läsk, Skulle det vara slut på en specifik dryck får vi istället ett felmeddelande. Skulle man mata in lite pengar kan man också få ut ett meddelande. Lägger man in för mycket pengar returnerar maskinen växel.

A.1.2 *Konceptualisering*: Tänkt dig systemet som en automat enligt automatteorin. Vilka tillstånd och vilka händelser finns? Beskriv tillstånden och händelserna med ett verb och ett substantiv (t.ex. ”kasta in mynt”, ”läsken är slut”, osv), förklara mer om du anser att det behövs för att kunna förstå vad du menar.

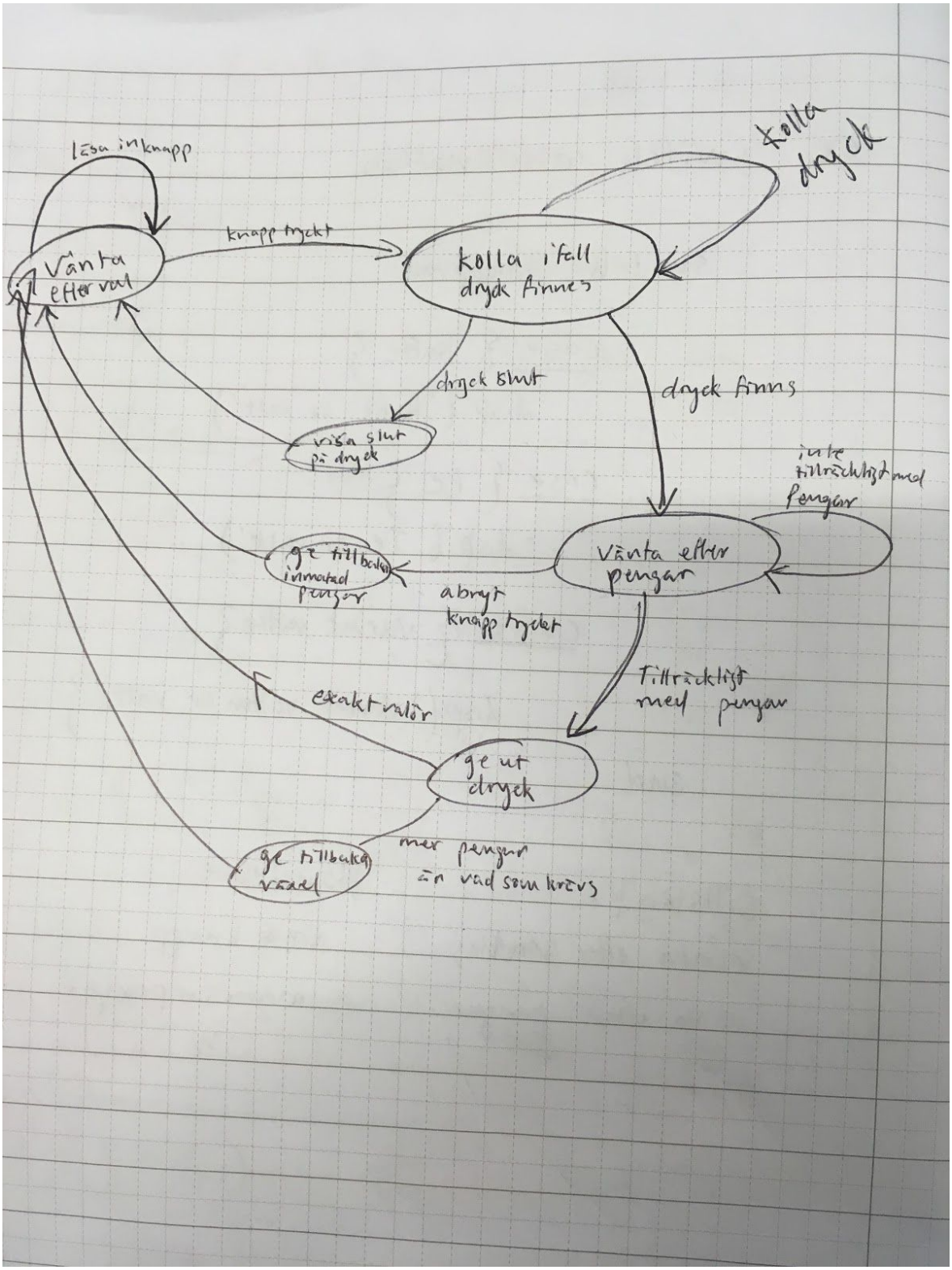
Tillstånd:
Vänta efter val,kolla dryck,kolla valör,ge ut dryck,ge ut växel, avbryt

Händelser:

knapp trycks, läsk slut, läsk finns,pengar matas in, för lite pengar, ge tillbaka växel, vänta

A.1.3 Rita nu automaten som tillståndsmodell.

Kopiera in en ritning (går bra med infogad kort på whiteboard eller papper)



A.1.4 Skriv pseudo-koden som innehåller en switch/case anvisning och som motsvarar tillståndsmodellen från A.1.3:

nextstate = vila
currentstate=nextstate;

While true{
switch currentstate:

<u>case vila:</u>
kanpp tryckt?
ja-->nextstate är kolla_dryck
nej-->nextstate är vila

<u>case kolla_dryck:</u>
finns dryck?
ja-->nextstate är vänta_efter_inmatning
nej→ nextstate är error

<u>case vänta_efter_inmatning:</u>
tillräckligt med pengar?
ja-->nextstate är ge_dryck
nej-->nextstate är vänta_efter_inmatning

<u>case ge_dryck:</u>
ge ut dryck
exakt valör?
ja-->nextstate är vila
nej-->nextstate är ge_växle
avbryt knapp tryckts???
nextstate är vila

<u>case ge_växel:</u>
ge ut växel
nextstate är vila

<u>case error:</u>
visa felmeddelande
nextstate är vila

currentstate=nextstate;

slut while}

A.2 Tillståndsmodell av ett elektroniskt låswtillståndsmaskiner (eng. ”finite state machine”). Tillämpningsområdet är en fysisk funktionsmodell av ett elektroniskt lås med tre knappar, tre små lysdioder och en större lysdiod.

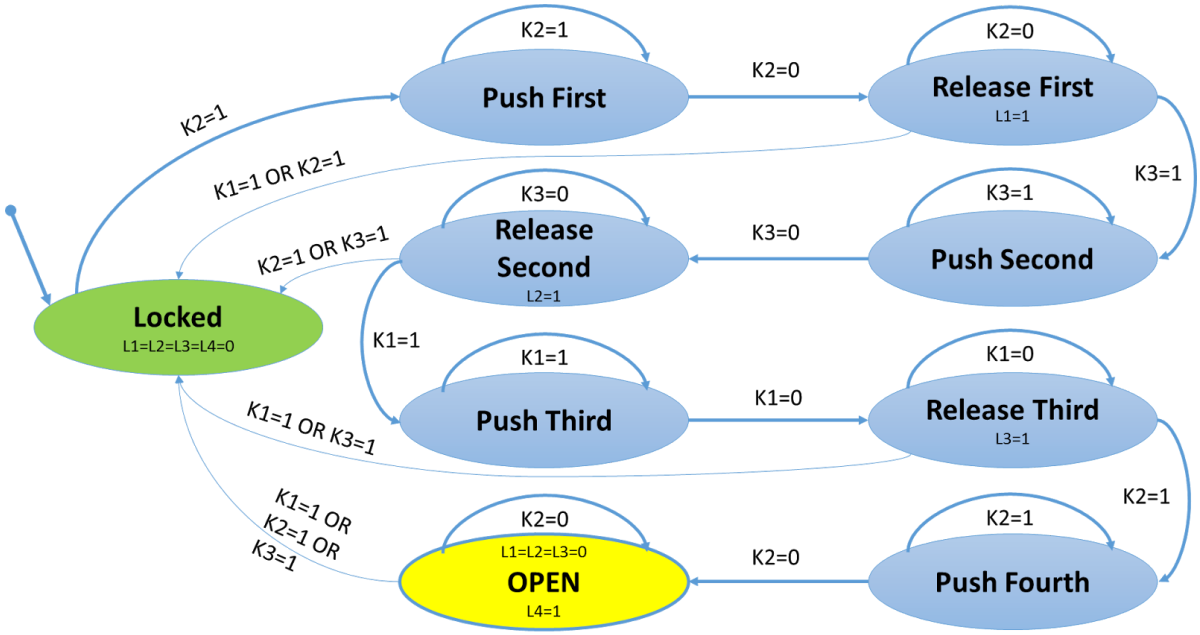


Fig. A.2: Tillståndsdiagram (FSM) av systemet som ska utvecklas

A.2.1) Systembegrepp

A.2.1.1 Vilka av följande egenskaper gäller för systemet ”elektroniskt lås”? Förklara varför eller varför inte!

begrepp	***analys och förklaring:***
Fysisk	sånt man kan ta på, exempelvis kretskort, knappar, lampor och sladdar. Något system vars storhet är fysiskt mätbar vilket är vårt kretskort,knappar etc.
Abstrakt	sånt man inte kan ta på, exempelvis matlab-kod. Storhet vilket inte är fysiskt mätbar
Tidskontinuerlig	Vår system är tidskontinuerilig på ett sätt att det körs om och om igen men mer tidsdiskret då vi vid olika punkter gör något.
Tidsdiskret (samplad/händelsestyrd)	Låset fungerar tills man matat in fel kombination 3 gånger. Vilket betraktar vissa diskrets tidspunkter då systemet tänkt förändra sitt tillstånd, exempelvis när vi trycker på knapp.
Dynamisk	Vårt system är dynamiskt då eftersom vid olika händelser vid vissa tillstånd kan vi hamna i flera olika exempelvis trycker

	man fel knapp hamnar man i låst annars rätt knapp hamnar man ett steg närmre att låsa upp låset
Linjär	Vårt system är inte linjärt eftersom systemet kommer utföra vissa operationer enbart vid olika tillfällen. Vårt system är händelsstyrt och därför är mer ett olinjärtsystem
Olinjär	Det är olinjärt om man slår in fel kod. Riktningen bland cases byts, och man går baklänges istället för framåt.
Deterministisk	Vi kan bara välja ett state åt taget därför är det deterministiskt. Vi vet vad som kommer hända vid dem olika händelserna
Stokastisk	Kan inte vara stokastiskt eftersom vi vet vad som kommer uppstå då man trycker ner en viss knapp. Vi vet vad som kommer hända

A.2.2) Automatteori, FSM-finite state machine/tillståndsdiagram

Följande frågor beträffar FSM enligt figur A.2.

A.2.2.1 Vilken är mängden X av tillstånd?

X={Locked, Push First, Release First, Push Second, Release Second, push Third, release Third, Push Fourth, Open}

A.2.2.2 Vilken är mängden E av händelser?

E={ k1=0, k1=1, k2=0, k2=1, k3=0, k3=1 }

A.2.2.3 Vilka är övergångsfunktionerna, (eng. ”transition functions”) **f:X x E -> X ?**

f(locked,k2)=pushfirst
f(locked,k1)=locked
f(locked,k3)=locked
f(pushfirst,~k2)=releaedfirst
f(pushfirst,k2)=pushfirst
f(releasedfirst,~k2)=releasedfirst
f(releasedfirst,k3)=pushsecond
f(releasedfirst,k1)=locked
f(releasedfirst,k2)=locked
f(pushsecond,k3)=pushsecond
f(pushsecond,~k3)=releasedsecond
f(releasedsecond,~k3)=releasedsecond
f(releasedsecond,k2)=locked
f(releasedsecond,k3)=locked
f(releasedsecond,k1)=pushthird
f(pushthird,k2)=pushthird
f(pushthird,~k2)=releasedthird
f(releasedthird,~k1)=releasedthird
f(releasedthird,k1)=locked

f(releasedthird,k3)=locked
f(releasedthird,k2)=pushedfouth
f(pushfourth,k2)=pushedfourth
f(pushfourth,~k2)=open
f(open,~k2)=open
f(open,k1)=locked
f(open,k2)=locked
f(open,k3)=locked

A.2.2.4 Hur kan man utifrån FSM (t.ex. enligt fig.A.2) lättast bestämma hur många övergångsfunktioner det ska vara?

Man kan genom att se antalet tillstånd bygga upp i tanken en sorts matric där vi har tillstånde in y-led och händelser i x-led. Vi har 9 tillstånd och i nästan alla tillstånd har vi 3 händelser knapp1,2 och 3 detta ger oss ungefär 3*9=27 olika transitionfunctions, man kan även se genom antalet pilar som går över till ett state.

Stämmar antalet överens med dem som ni anger under A.2.2.3?

Ja eftersom vi fick 27 stycken verkar det stämma
3 events*9 states=27 transition functions

A3) Programmering av tillståndsdiagram i Matlab

A.3.1 Ta reda på hur ”switch” konstruktet kan användas för att programmera tillståndsmaskiner i Matlab. Ge ett exempel hur de första tre tillstånd i fig. A.2 med rätta övergångar kan programmeras i Matlab med hjälp av ”switch”.

case 1: kolla knappnedtryckning. Om rätt knapp är nedtryckt, gå vidare till nästa case. Om fell knapp är nedtryckt, gå till locked och öka tries-variabeln med 1.

case2: stanna kvar i case2 tills knappen släpps. När knappen släpps, gå till case3.

case3: kolla knappnedtryckning. Om rätt knapp är nedtryckt, gå vidare till nästa case. Om fell knapp är nedtryckt, gå till locked och öka tries-variabeln med 1.

A.3.2. Programmera elektroniska låset som matlabprogram som följer tillståndsdiagrammet i fig A.2 i uppgiftsbeskrivningen ovan. Den stora lampan ska vara släckt så länge som koden inte är korrekt och tändas när rätta knapparna har trycks i rätt ordning. De små lamporna ska tändas och släckas enligt informationen i tillståndsdiagrammet.

```
function lock( a )
%LOCK Summary of this function goes here
%   Detailed explanation goes here
nextState = 'locked';
currentState = nextState;
```

```
a.pinMode(2,'INPUT');%btn 1
a.pinMode(3,'INPUT');%btn 2
a.pinMode(4,'INPUT');%btn 3

a.pinMode(6,'OUTPUT');% led 1
a.pinMode(7,'OUTPUT');% led 2
a.pinMode(8,'OUTPUT');% led 3
a.pinMode(9,'OUTPUT');% Bigled 4

a.digitalWrite(6,0);
a.digitalWrite(7,0);
a.digitalWrite(8,0);
a.digitalWrite(9,0);
tries = 0;



%   k1=~a.digitalRead(2);
%  k2=~a.digitalRead(3);
%   k3=~a.digitalRead(4);


while(1)


   if (tries >=3)
      a.digitalWrite(6,0);
      a.digitalWrite(7,0);
      a.digitalWrite(8,0);
      a.digitalWrite(9,0);


      disp('too many failed tries');

      break;
   end

   k1=~a.digitalRead(2);
   k2=~a.digitalRead(3);
   k3=~a.digitalRead(4);

   if k1 | k2 | k3
     buttonPressed = 1;
   end



   if k1
      disp('knapp1')
   end

   if k2
      disp('knapp2')
   end

   if k3
      disp('knapp3')
   end
```

```
switch currentState
   case {'changepass'}

   case{'locked'}
      %  disp('locked');
      a.digitalWrite(6,0);
      a.digitalWrite(7,0);
      a.digitalWrite(8,0);
      a.digitalWrite(9,0);

      if k2 & ~(k1 | k3)
         %  disp('ska bli push1');
         nextState = 'push1';
      end

      if k1 | k3
         nextState = 'go2locked';
      end
   case{'push1'}
      %  disp('push1');

      if k2 & ~(k1 | k3)
         nextState = 'push1';

      elseif ~k2 & ~(k1 | k3)
         nextState = 'release1';
      end

      if k1 | k3
         nextState = 'go2locked';
      end
   case{'release1'}
      %  disp('release1');
      if ~k2 & ~(k1 | k2)
         a.digitalWrite(6,1);
      end

      if k1 | k2
         nextState = 'go2locked';
      end

      if k3 & ~(k1 | k2)
         nextState = 'push2';
      end
   case{'push2'}
      disp('push2');
      if k3 & ~(k1 | k2)
         nextState = 'push2';
      end

      if ~k3 & ~(k1 | k2)
         nextState = 'release2';
      end

      if k1 | k2
         nextState = 'go2locked';
      end
```

```
case{'release2'}
   %   disp('release2');
   if ~k3 & ~(k1 | k2)
      a.digitalWrite(7,1);
   end



   if k1 & ~(k2 | k3)
      nextState = 'push3';
   end

   if k2 | k3
      nextState = 'go2locked';
   end


case{'push3'}

   if k1 & ~(k2 | k3)
      nextState = 'push3';
   end

   if ~k1 & ~(k2 | k3)
      nextState = 'release3';

   end

   if k2 | k3
      nextState = 'go2locked';
   end

case{'release3'}

   if ~k1 & ~(k2 | k3)
      a.digitalWrite(8,1);
      nextState = 'release3';
   end

   if k2 & ~(k1 | k3)
      nextState = 'push4';

   end

   if k1 | k3
      nextState = 'go2locked';
   end


case{'push4'}
   if k2 & ~(k1 | k3)
      nextState = 'push4';

   end

   if ~k2 & ~(k1 | k3)
      nextState = 'open';

   end
```

```
        if k1 | k3
            nextState = 'go2locked';
        end

    case{'open'}
        if ~(k2 | k1 | k3)
            nextState = 'open';
        elseif (k2 | k1 | k3)
            nextState = 'go2locked';
        end

        a.digitalWrite(6,0);
        a.digitalWrite(7,0);
        a.digitalWrite(8,0);
        a.digitalWrite(9,1);


    case{'go2locked'}

        if k2 | k1 | k3
            nextState = 'go2locked';
        end

        if ~(k2 | k1 | k3)
            tries = tries+1;
            nextState = 'locked';
        end
  end
  currentState=nextState;
end



end
```

A.3.3 Utveckla ditt program så att programmet slutar efter tre felförsök. Beskriv med hjälp av tillståndsmodellen i fig. A.2 vilka anpassningar som krävs.

Vid varje övergång till locked ökar vi int-variabeln “tries” med 1. när tries är större än eller lika med 3 kör vi “break;”

A4) Utbyggnad av Matlabprogrammet

A.4.1 Fundera på hur ditt program kan byggas ut med så få ändringar som möjligt så att man lätt kan byta koden. Använd gärna switch/case delen för att komma in i en ”byt kod tillstånd”. Beskriv dina idéer och vilka ändringar i programmet som det skulle krävas.

Man kan skapa en matrix med 4 platser, där man lagrar varje knapp i. Med en variabel kan man öka när man matar in rätt och använda för att kolla varje knapp med varje plats i matrixen. Ifall samma så låses låset upp annars så är det låst. Man kan skapa 2 cases pushed och released och läsa av knappen på samma sätt

B) Praktiska delen (att genomföra i labbsalen)

VIKTIG INFORMATION: När ni startar upp datorn i labbsalen ska ni välja ”Elektroniklabb Windows 7”. Välj sedan MATLAB 2015b – versionen!

B.1 Bygga upp utrustningen

Ni behöver först bygga upp utrustningen med tre knappar, tre små LED och en stor LED. Se till att välja rätt resistorerna så att knapparna och lamporna fungerar med Arduino Dues 3,3V.

B.1.1 Testa labbutrustningen med ett testprogram som läser av knapparna och sedan tänder eller släcker lamporna.
Kopiera in programmet här:

```
a.pinMode(2,'INPUT');%btn 1
a.pinMode(3,'INPUT');%btn 2
a.pinMode(4,'INPUT');%btn 3

a.pinMode(6,'OUTPUT');% led 1
a.pinMode(7,'OUTPUT');% led 2
a.pinMode(8,'OUTPUT');% led 3
a.pinMode(9,'OUTPUT');% Bigled 4

a.digitalWrite(6,0);
a.digitalWrite(7,0);
a.digitalWrite(8,0);
a.digitalWrite(9,0);

k1=0;
k2=0;
k3=0;

while(1)

    k1 = ~a.digitalRead(2);
```

```
    k2 = ~a.digitalRead(3);
    k3 = ~a.digitalRead(4);


    if k1
        a.digitalWrite(6,1);
    end

    if k2
        a.digitalWrite(7,1);
    end

    if k3
        a.digitalWrite(8,1);
    end

    if k1 & k2 & k3
        a.digitalWrite(9,1);
    end
end
```

B2 Matlabprogrammering

B.2.1 Kör matlabprogrammet som ni har programmerat i A.3.2. och A.3.3
Fungerade programmet som ni hade tänkt? Om inte vad behövdes rätta till? (Beskriv ändringarna och hur ni tänkte fel) (

Det fungerade bra. Genom att skriva kod för diagrammet som det var fungerade det jättebra. Blev lite fel med antalet försök som skulle bryta programmet men löste sig eftersom vi räknat en gång för mycket, fungerar jättebra nu!

B.2.2 Gör ändringarna i programmet för att kunna först spara en villkorlig kombination av fyra knappar och sedan kunna köra låset med kombinationen enligt era funderingar från A.4.1.
Kopiera in hela den fungerande matlabkoden här:

```
function dynamiclock( a )

%--------------------------------------------------------------------------------
%   Dynamiclock
%   Ett lås vilket styres med hälp av en arduino
%   Systemet består av 4 knappar
%   varav 3 av dem är för kombination och 1 för att byta lösen
%   Vid rätt kombination tänds den storalampan
%   Slår man fel kombination 3 gånger bryts Systemet
%   Kombinationen väljs vid start av programmet och kan bytas enbart när öppet tillstånd
% Authors: Leonard Holgersson & Hadi Deknache
%--------------------------------------------------------------------------------
```

```
nextState = 'changepass';
currentState = nextState;

%Initerar alla knappar som ingågnar för att läsa in värde
a.pinMode(2,'INPUT');%btn 1
a.pinMode(3,'INPUT');%btn 2
a.pinMode(4,'INPUT');%btn 3
a.pinMode(5,'INPUT');%btn change pass

%Initerar alla led som utgångar för att skriva ut värde
a.pinMode(6,'OUTPUT');% led 1
a.pinMode(7,'OUTPUT');% led 2
a.pinMode(8,'OUTPUT');% led 3
a.pinMode(9,'OUTPUT');% Bigled 4

%Sätter alla led till avstängda
a.digitalWrite(6,0);
a.digitalWrite(7,0);
a.digitalWrite(8,0);
a.digitalWrite(9,0);

%Vår försöksvariabel som håller reda på antalet fel man tryckt
tries = 0;
i = 1;

btnPressed = 0;
k = 0;

%Matrix med kombinationer som ska lagras
b = zeros(2,4);

while(1)

   % Kollar ifall man försökt öppna låset mer än 3 gånger
   % Stoppar programmet ifall tries är 3 eller större

   if (tries >=3)
      a.digitalWrite(6,0);
      a.digitalWrite(7,0);
      a.digitalWrite(8,0);
      a.digitalWrite(9,0);

      disp('too many failed attempts');

      break;
   end
```

```
%Läser in knapparna

k1=~a.digitalRead(2);
k2=~a.digitalRead(3);
k3=~a.digitalRead(4);
passBtn = ~a.digitalRead(5);

%Om man trycker någon av dem 3 knapparna sätts btnPressed till1
%Annars är den 0
if k1 | k2 | k3
   btnPressed = 1;
elseif ~(k1 | k2 | k3)
   btnPressed = 0;
end

%Ser vilken knapp som tryckts ifall btnPressed
%Samt kontrollerar så att man inte trycker flera samtidigt
if btnPressed
   if k1 && ~(k2 | k3)
      k = 1;
   end
   if k2 && ~(k1 | k3)
      k = 2;
   end
   if k3 && ~(k1 | k2)
      k = 3;
   end

end

%Switch case sats som kollar state samt bytar ifall en ny händelse inträffar
switch currentState
   %Case för att byta kombination
   case {'changepass'}
      if btnPressed
         nextState = 'changepress';
         btn = k;
      end

   %Kollar vilken knapp som tryckts
   case {'changepress'}

      if btnPressed
         btn = k;
         nextState = 'changepress';
      end

      if ~btnPressed
         nextState = 'changerelease';
```

```
    end

%Case för när man släpper knappen som skall lagras
case {'changerelease'}
    %Säkerhetsställer att man släppt knappen
    %Och att i mindre än 4 då kombination ska vara 4 lång
    %Lagrar knapp i Matrix och ökar i
    if ~btnPressed && i <4
        b(1,i) = btn;
        disp(b(1,i));
        nextState = 'changepass';
        i = i+1;

    % Annars ifall i =4 eller större samt ingen knapp tryckts
    %Lagrar knapptryck och går till nästa state låst
    elseif ~btnPressed && i >=4
        b(1,i) = btn;
        disp(b(1,i));
        nextState = 'locked';
        disp(b(1,1:4));
        tries = 0;
    end

%Case låst stänger den av alla led
%Läser av första rätta knapp i matrixen

case{'locked'}

    i = 1;
    a.digitalWrite(6,0);
    a.digitalWrite(7,0);
    a.digitalWrite(8,0);
    a.digitalWrite(9,0);

    b(2,i) = k;

    %Kollar ifall knapp motsvarade samma som i matrixen
    %Går till nästa state
    if btnPressed && k == b(1,i)
        nextState = 'push1';
    end

    % ifall inte samma så fortsätter den i go2locked
    if btnPressed && k ~= b(1,i)
        nextState = 'go2locked';
    end

% Fastnar här sålänge man trycker knapp
case{'go2locked'}
```

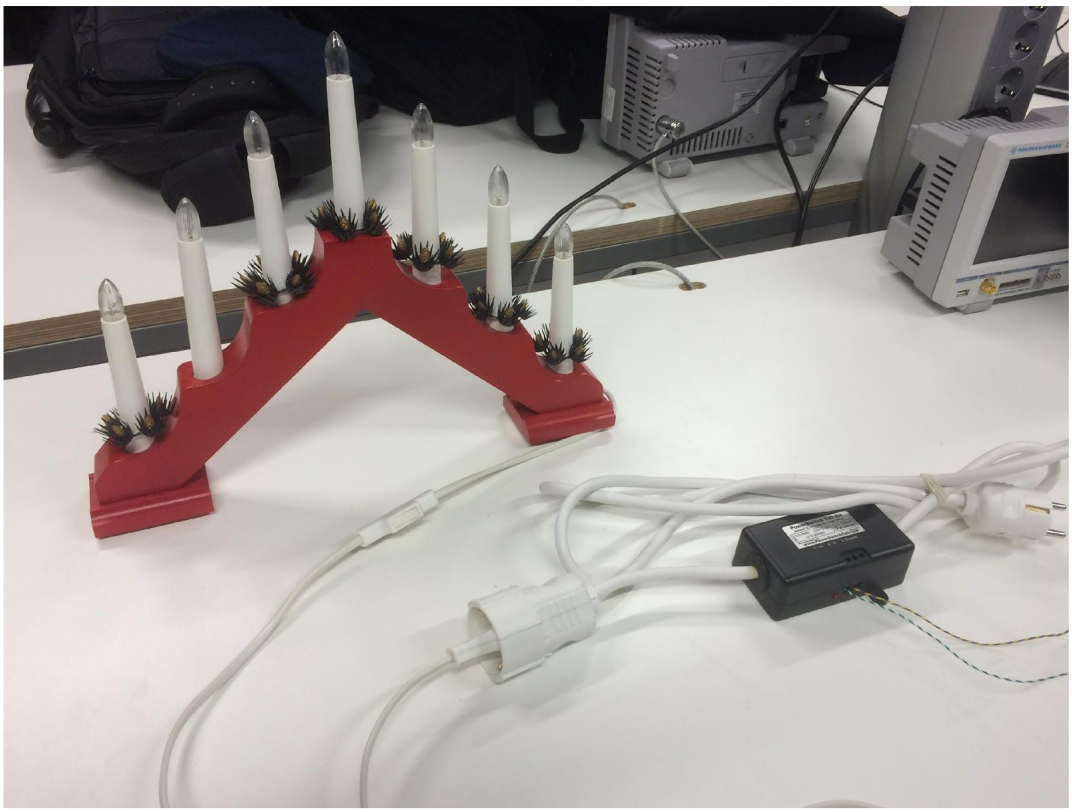
```
    if k2 | k1 | k3
        nextState = 'go2locked';
    end
    %Ökar tries ifall man trycker fel knapp
    if ~(k2 | k1 | k3)
        tries = tries+1;
        nextState = 'locked';
    end

%case för första rätta kombination
case{'push1'}

    % Medan rätt knapp nedtryckt hamnar den i samma state
    if btnPressed && k == b(1,i)
        nextState = 'push1';

        %Om fel knapp trycks låses den igen
        if btnPressed && k ~= b(1,i)
            nextState = 'go2locked';
        end

    % Annars rätt knapp släpps går den till release1
    elseif ~btnPressed
        nextState = 'release1';
        i=2;
    end

case{'release1'}

    %Tänder lampa för första rätt knappkombination
    if ~btnPressed
        a.digitalWrite(6,1);
    end
    %Om fel knapp trycks låses den igen
    if btnPressed && k ~= b(1,i)
        nextState = 'go2locked';
    end
    %Ifall rätt knapp trycks för kombination 2 går den till nästa state
    if btnPressed && k == b(1,i)
        nextState = 'push2';
    end

case{'push2'}
    % Medan rätt knapp nedtryckt hamnar den i samma state
    if btnPressed && k == b(1,i)
        nextState = 'push2';
    end
    % Annars rätt knapp släpps går den till release2
```

```
        if ~btnPressed
            nextState = 'release2';
            i=3;

        end
        %Ifall fel knapp trycks ner istället låses det
        if btnPressed && k ~= b(1,i)
            nextState = 'go2locked';
        end

    case{'release2'}
        %kollar så att knapp släppt
        if ~btnPressed
            a.digitalWrite(7,1);
        end

        %Kollar nästa kombination att det är rätt
        if btnPressed && k == b(1,i)
            nextState = 'push3';
        end
        %Låses ifall fel knapp trycks
        if btnPressed && k ~= b(1,i)
            nextState = 'go2locked';
        end


    case{'push3'}
        %Ifall rätt knapp trycks går den till push 3
        if btnPressed && k == b(1,i)
            nextState = 'push3';
        end

        %Ifall knapp släpps går den till release3 och ökar steget i matrixen
        if ~btnPressed
            nextState = 'release3';
            i=4;

        end
        %Ifall fel knapp låses låset
        if btnPressed && k ~= b(1,i)
            nextState = 'go2locked';
        end

    case{'release3'}
        %Ifall ingen knapp trycks tänder den lampan för rätt kombination
        if ~btnPressed
            a.digitalWrite(8,1);
            nextState = 'release3';
        end
```

```
        %ifall rätt knapp trycks så hoppar den till nästa state
        if btnPressed && k == b(1,i)
            nextState = 'push4';

        end
        %ifall fel knapp låses systemet
        if btnPressed && k ~= b(1,i)
            nextState = 'go2locked';
        end


    case{'push4'}
        %Kollar om knapp tryckts och rät knapp i matrixen motsvarar
        if btnPressed && k == b(1,i)
            nextState = 'push4';

        end
        %Ifall knapp släpps går den till öppet
        if ~btnPressed
            nextState = 'open';

        end
        %Ifall fel knapp trycks låses det
        if btnPressed && k ~= b(1,i)
            nextState = 'go2locked';
        end

    case{'open'}
        %Nollställer försöken
        tries = 0;
        %Kollar så att ingen knapp trycks
        if ~btnPressed
            nextState = 'open';
        %Annars ifall någon knapp trycks så låses låset
        elseif (k1 | k2 | k3)
            nextState = 'go2locked';
        end
        a.digitalWrite(6,0);
        a.digitalWrite(7,0);
        a.digitalWrite(8,0);
        a.digitalWrite(9,1);


        %Ifall man trycker på byt kombinationknappen
        %Hoppar den in i byt kombination
        if passBtn
            disp('Changepass')
            nextState='changepass';
            i = 1;
```

```
            a.digitalWrite(9,0);

        end

    end

    %Sätter nästa state till nuvarande state
    currentState=nextState;
end

end
```

B.2.3 Anslut julstaken med hjälp av ”power switch tail kit” (http://www.powerswitchtail.com/Documents/PSSRTK%20Instructions.pdf) så att ni kan tända julljuset när låset öppnar!



B.2.6 Spela in en video med ert fungerande Matlabprogram. Videon ska dokumentera följande i bild och med ***era muntliga kommentarer***:
- vid rätt knappkombinationen tänds stora LED och julstaken
- visa hur man kommer in i ”byt kod”-tillståndet.
- byt koden till en ny kombination.
- visa att den nya koden fungerar och tänder LED och julstake.
- visa att LED och julstake inte tänds om man väljer fel kod.
- visa att programmet stänger ner efter tredje felet.

Youtube länk:
https://youtu.be/WtLQ_wIOpRE

C) Reflektion och utvärdering över det egna lärandet

Tidigare studenter har framfört olika åsikter om denna tredje del. För att förklara vad den går ut på hänvisar jag till kapitel om ”Reflektionsdokumentet” i Andersen och Schwenckes bok ”Projektarbete – en vägledning för studenter”, Studentlitteratur. Författarna beskriver syftet med reflektion över sitt lärande sammanfattningsvis ungefär på följande sätt:
Det handlar om att öka medvetenheten om den egna inlärningsprocessen. Genom denna reflektion lär du känna dig själv, och hur du förhåller dig till dina medstudenter. Du blir också tryggare i dig själv och det du har gjort och står för. Du blir också mer medveten om hur du lär, så att du kan styra inlärningen och studierna i din egen riktning och ta ansvar för din egen inlärning. Du utvecklar medvetenhet om inlärning och stärker din egen förmåga att arbeta effektivt på längre sikt.
Vidare är det ett viktigt syfte att ge feedback, både till läraren, kursansvarig och till skolan. Därmed blir det möjligt att göra förbättringar för senare studentkullar.

I fall att ni känner att ni inte kan komma på något vettigt att skriva om, så finns en lista med nyckelfrågor i bilagan som ni kan titta på och få inspiration ifrån.

C.1 Vad tycker du/ni var lärorik med uppgiften? (Minst 5 meningar och minst 75 ord!)
vi har fått lära oss om hur man bygger upp en switch case sats i Matlabmiljön på ett korrekt sätt. Vi har även lärt oss hur man genom vardagslivet med diverse maskiner kan tänka sig hur det fungerar med hjälp av tillståndsmodeller. Vi har även lärt oss att utifrån en tillståndsdiagram, kunna skriva ett program i Matlab. Vi har även lärt oss om tillståndsmaskiner att implementera inom matlab och använda det på ett korrekt sätt. Vi har lärt oss att inom ett kodsystem kunna byta en kombination på ett bra sätt.

C.2 På vilket sätt har ni fördjupat er i något nytt? Vad kände ni från tidigare och på vilket sätt har ni lärt er något nytt utifrån det ni redan kunde? (Minst 5 meningar och minst 75 ord!)
Det här var första gången vi körde switch/case i MATLAB. Tidigare har vi bara kört en massa if-satser. I föregående labbar har vi kört massor med ifsatser. Vi märkte nu att det kanske skulle bli bättre med switch case satser och snyggare dessutom. Vi har fått en lite mer fördjupad kunskap om hur ett systems struktur är uppbyggt genom att studera teoridelen med läskautomaten. Genom att kolla på ett system kan man tänka sig hur det är uppbyggt med olika händelser och tillstånd som det kan hamna i.

C.3 Vad var det svåraste med uppgiften? (Minst 5 meningar och minst 75 ord!)

Att MATLAB tycker om att byta plats på knapparna. Vi tror att det beror på att vi avbrutit programmet med Ctrl+C. Bortsett från det var systembegreppen svåra. Ifall det inte hade varit problem med Matlab så hade det definitivt tagit mindre tid. Vi förstod senare när det blivit problem att matlab läser in fel knappar att det var dags att starta om för att det skulle fungera. Genom att det uppstått fel tidigare kan vi nu nästan direkt se ifall det skulle vara fel någonstans om det beter sig på samma sätt vilket kanske är bra på sitt sätt.

Hur mycket tid totalt har ni lagt ner på att lösa uppgiften och hur mycket av denna tid har ni lagt på det som ni anser var det svåraste?

12 timmar! ungefär 3h av tiden gick åt teoridelen att förstå och skissa. Sedan resterande 9 timmar gick åt att implementera state machine och sedan skriva om lite med att kunna ändra kombination.

C.4 Synpunkter, förslag, kommentarer? (Minst 5 meningar och minst 75 ord!)

Väldigt rolig labb man fick lära sig lite av allt, samt i slutändan fick vi ett spel som fungerade vilket man kunde spela. Flera labbar med spel uppskattas och är väldigt roliga. Eftersom att vi haft bra föreläsningar med genomgång av det som krävdes har det varit relativt enkelt att skriva tillstånds maskinerna samt veta ungefär hur man ska skriva dem, detta uppskattas verkligen. Sista biten med att byta kombination var väldigt intressant och man lärde sig en del hur man ska lösa detta problemet.

Bilaga:

Översikt över Matlab instruktioner

Matlab-instruktion	argument	Beskrivning
pinMode(a,pin,state)	a: arduino-objektet pin: pin-nummer (2-13) state:'OUTPUT', 'INPUT'	Initialiserar port med pin-nummer till in- eller utgång.
digitalWrite(a,pin,level)	a: arduino-objektet pin: pin-nummer (2-13) level: 0, 1	Sätter en pinne till 0 eller 1. Måste först deklareras som utgång
digitalRead(a,pin)	a: arduino-objektet pin: pin-nummer (2-13)	Läser av status (digitalt värdet) av en pin. Bör deklareras som ingång innan den läses.
analogWrite(a,duty)	a: arduino-objektet Duty: 8-bit PWM signal 0..255	PWM-utgång, låst till DAC1. Använd motorshielden för att styra motorer!!
analogRead(a,pin)	a: arduino-objektet pin: 'A0', 'A1', ('A3')	Läser av en analog pinne. Skall ej deklareras som ingång innan läsning. AD-upplösning 1024bit

```
>> a=arduino_com('COM4')
Attempting connection .............
Connection successful!

a =

Connected to COM4 port

fx >>
```

Exempel av nyckelfrågor i samband med reflektioner och utvärdering av det eget lärande

- **Är du nöjd med samarbetet i gruppen? Vad fungerade bra vad mindre bra?**
- **Hur fungerade beslutsprocessen i gruppen? Kunde det ha varit bättre?**
- **Vilken rolluppdelning valde ni? Hur kändes det?**
- **Hur effektivt och systematiskt var ni? På vilket sätt finns utrymme till förbättring?**

- **Hur uppfattade ni sambandet mellan teori och tillämpning?**
- **Hur uppfattade ni sambandet mellan föreläsning och laboration?**
- **Genomförde ni uppgifterna i ordningen som de står i rapporten eller i vilken ordning besvarade ni frågorna? Hur kändes det så som ni gjorde?**
- **Vad gjorde ni när det blev problem och/eller ni inte kom vidare? Använde ni både samma strategi? Hur kändes det?**
- **Hur organiserade ni er inför slutinlämningen? Gick ni igenom alla frågor en gång till gemensam eller lämnade ni bara in allt när ni kände att ni var färdiga?**
- **Planerade ni fasta tider när ni gemensam i gruppen genomförde laborationsuppgifterna eller hur organiserade ni er?**
- **??**