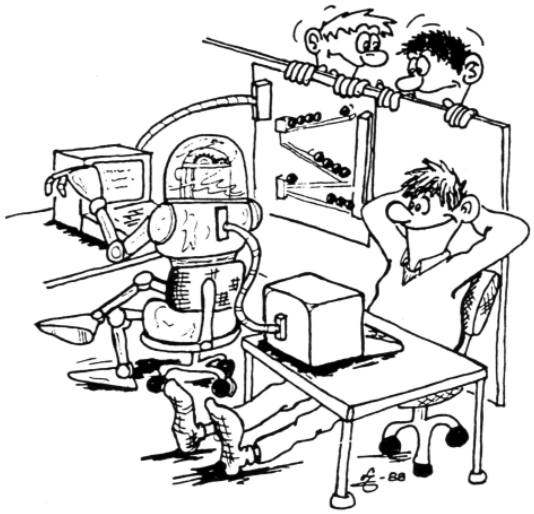
Industriautomation

Tentamen SSY 066, fredag 16/01-2015, fm, H

Lärare: Kristofer Bengtsson, 0768 979561



Fullständig lösning ska lämnas på samtliga uppgifter. I förekommande fall av tvetydigt formulerade tentamensuppgifter ska den föreslagna lösningen och eventuella antaganden motiveras. Examinator förbehåller sig rätten att godkänna rimligheten i antaganden och motiveringar.

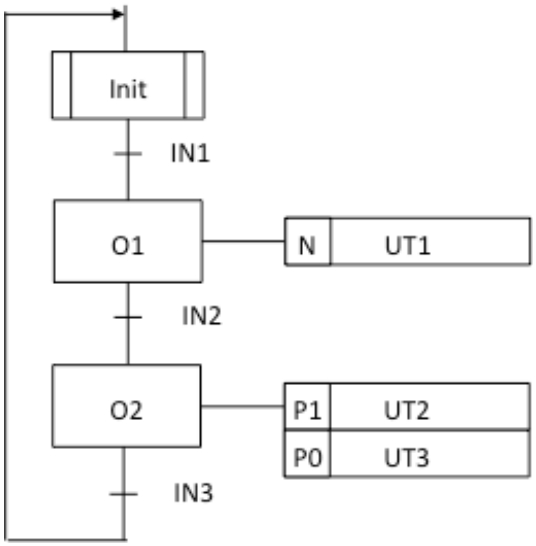
Totalt omfattar tentamen 15 poäng. För betygen tre, fyra och fem krävs 7, 10 resp. 13 poäng. Lösningar anslås direkt efter tentatillfället på kursens hemsida i Pingpong. Granskning av rättningen sker 28 januari kl. 12:30 – 13:30 på institutionen.

OBS. Inga hjälpmedel är tillåtna.

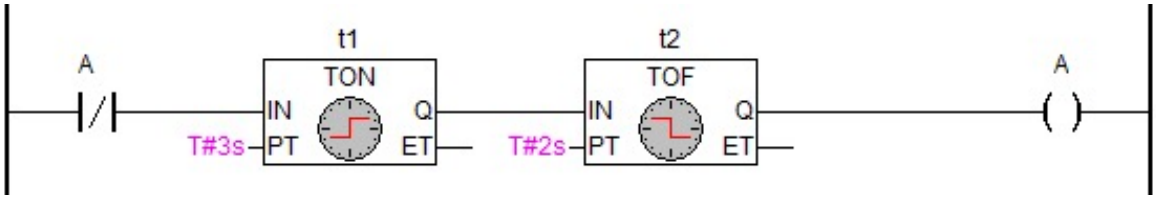
Uppgift 1 Ladder

a) Ett system styrs med följande SFC. Översätt dess exakta beteende till ladderlogik.

(3 poäng)



b) Ladder bygger på enkla principer, men tar lång tid att bemästra. Följande mycket trevliga rung klurade jag ihop under mellandagarna.



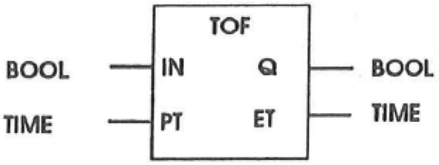
Beskriv variabel A:s förändring under 10 sekunder med hjälp av ett timingdiagram. Variabel A är false initialt. Viktigt att sätta ut tider i diagrammet för full poäng.

Olika timerfunktionsblock och exempel på timingdiagram är beskrivna på nästa sida.

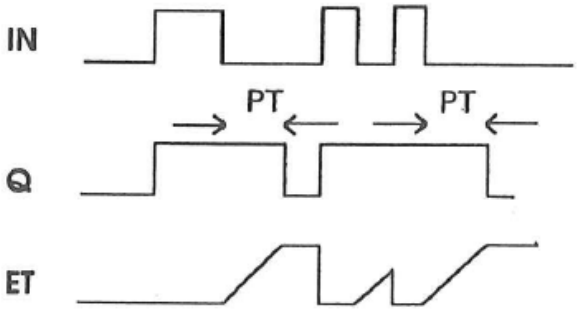
(2 poäng)

Timers

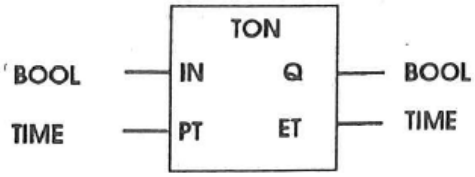
Off Timer function block



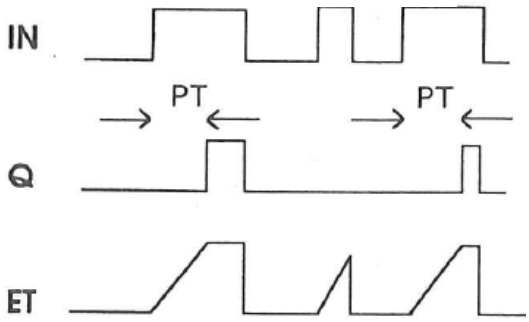
Off Timer timing diagram



On Timer function block

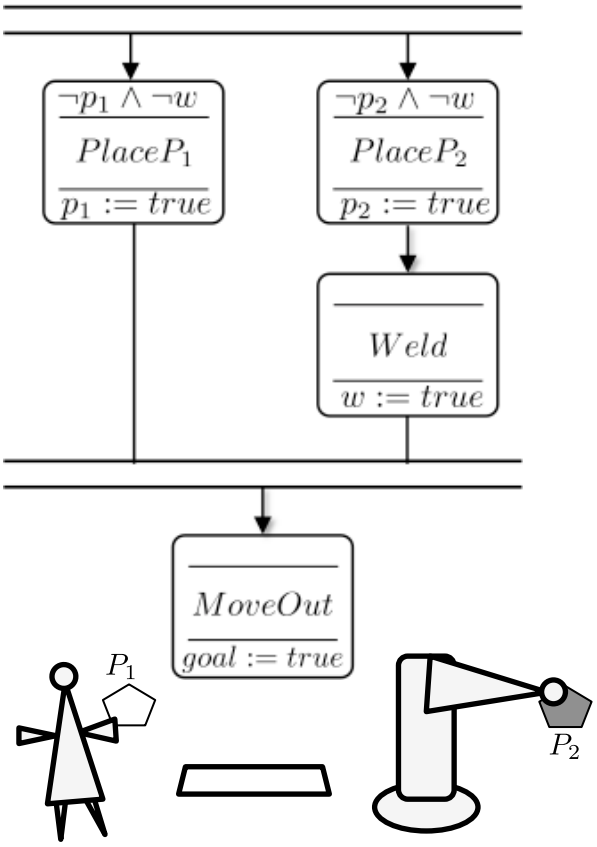


On Timer timing diagram



Uppgift 2 Graf och specifikation

I denna uppgift skall du undersöka specifikationen till höger. Systemet består av en operatör och en robot som skall svetsa ihop en liten produkt. Produkten består av plåtbitarna P_1 och P_2. Operatören lägger ner P_1 (operation *PlaceP₁*) på bordet och roboten lägger ner P_2 (*PlaceP₂*). Roboten svetsar ihop de båda delarna (operation *Weld*) och sedan plockar operatören ut produkten (*MoveOut*). I uppgiften tittar vi endast på en produktionscykel.



Till höger har jag gjort en specifikation för systemet med hjälp av en SOP. Operationernas pre och postcondition beskrivs både med hjälp av conditions skrivna som predikat och med conditions beskrivna med sekvensen (de grafiska elementen). Sekvensen säger t.ex. att *Weld* inte kan starta förrän *PlaceP₂* är färdig och att MoveOut inte kan starta förrän både *PlaceP₁* och *Weld* är färdiga. I denna uppgift kan ni förutsätta att en operation inte tar någon tid, den startar och slutar direkt. Specen använder de fyra boolska variablerna p_1, p_2, w och *goal* med initial false.

a) Rita upp en graf utifrån SOPen. Markera initialtillståndet med en liten pil och tillståndet som uppfyller målpredikatet: *goal* == *true* med en extra cirkel.

(2 poäng)

b) Tyvärr kan systemet hänga sig enligt denna specifikation, då den inte kan nå måltillståndet från vissa tillstånd. Markera deadlock-tillståndet/tillstånden i din graf. Föreslå också en ny guard på en av operationerna så vi inte hamnar i detta jobbiga tillstånd

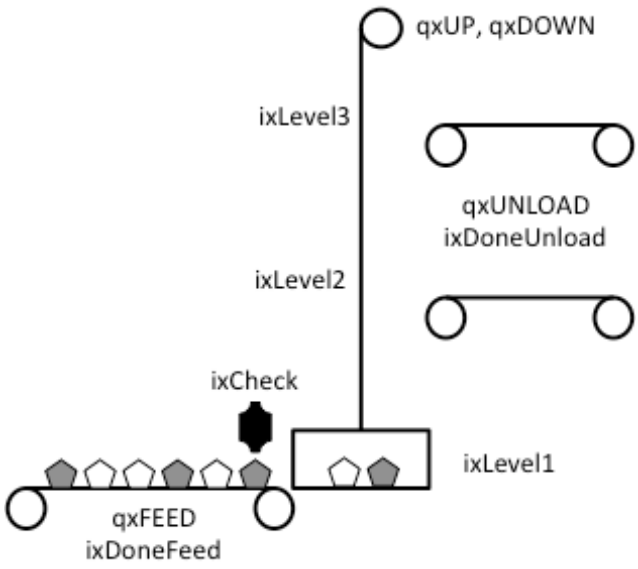
(2 poäng)

c) En SOP beskriver operationernas conditions både med sekvenser och predikat som jag skrev ovan. Oftast behöver man inte skriva ut predikatet i en operation utan kan använda bara sekvensen för att visa alla conditions. Rita om SOPen utan guard och action predikat (alltså bara med pilar och sträck) där du tar hänsyn till din nya guard.

(1 poäng)

Uppgift 3 SFC

Lagerhissen till höger transporterar pentagoner till olika våningar. Din uppgift är att implementera styrningen av systemet med hjälp av flera SFCer.



Hissen transporterar två pentagoner samtidigt. Det kan vara en grå och en vit (i olika ordning), två gråa eller två vita pentagoner i hissen.

Den grå typen skall köras till våning 2 och den vita skall till våning 3. Vilken typ det är upptäcks innan de matas in i hissen med *ixCheck*. Om *ixCheck* är lika med 2 så är pentagonen grå och om värdet är 3 är den vit. Pentagonen som lastas på först i hissen skall alltid lastas av först på rätt våning.

För att mata in en ny pentagon i hissen sätts utsignalen *qxFEED* true av din kod. När insignalen *ixDoneFeed* blir hög, då sätter din kod *qxFEED* till false. Detta behöver alltså göras två gånger för att två pentagoner skall finnas i hissen. Man mäter på den pentagon som skall matas in i hissen.

När hissen är fylld med två pentagoner, skall hissen åka och lämna av den första pentagonen på rätt våning (enligt dess färg) och sedan den andra. Hissen körs upp med *qxUP* och ner med *qxDOWN*, och när *ixLevelx*, där x = {1, 2, 3} är hög är hissen vid våning x. När hissen är vid en våning matas pentagonen ut med hjälp av *qxUNLOAD* tills *ixDoneUnload* blir hög. Alltså på samma sätt som inmatningen.

Observera att koden måste åka den bästa vägen. Det är inte tillåtet åka till samma våning flera gånger eller stanna på en våning där du inte lastar av. Om man skall lasta av båda pentagonerna på samma våning skall man alltså inte åka iväg från våningen mellan avlastningarna.

Dela upp implementeringen i flera SFCer (minst två). Inmatningen och hissen MÅSTE vara i olika SFCer. Dessa SFCer kan kommunicera antigen via variabler eller med hjälp av stegets namn som i kulbanelabben.

(5 poäng)