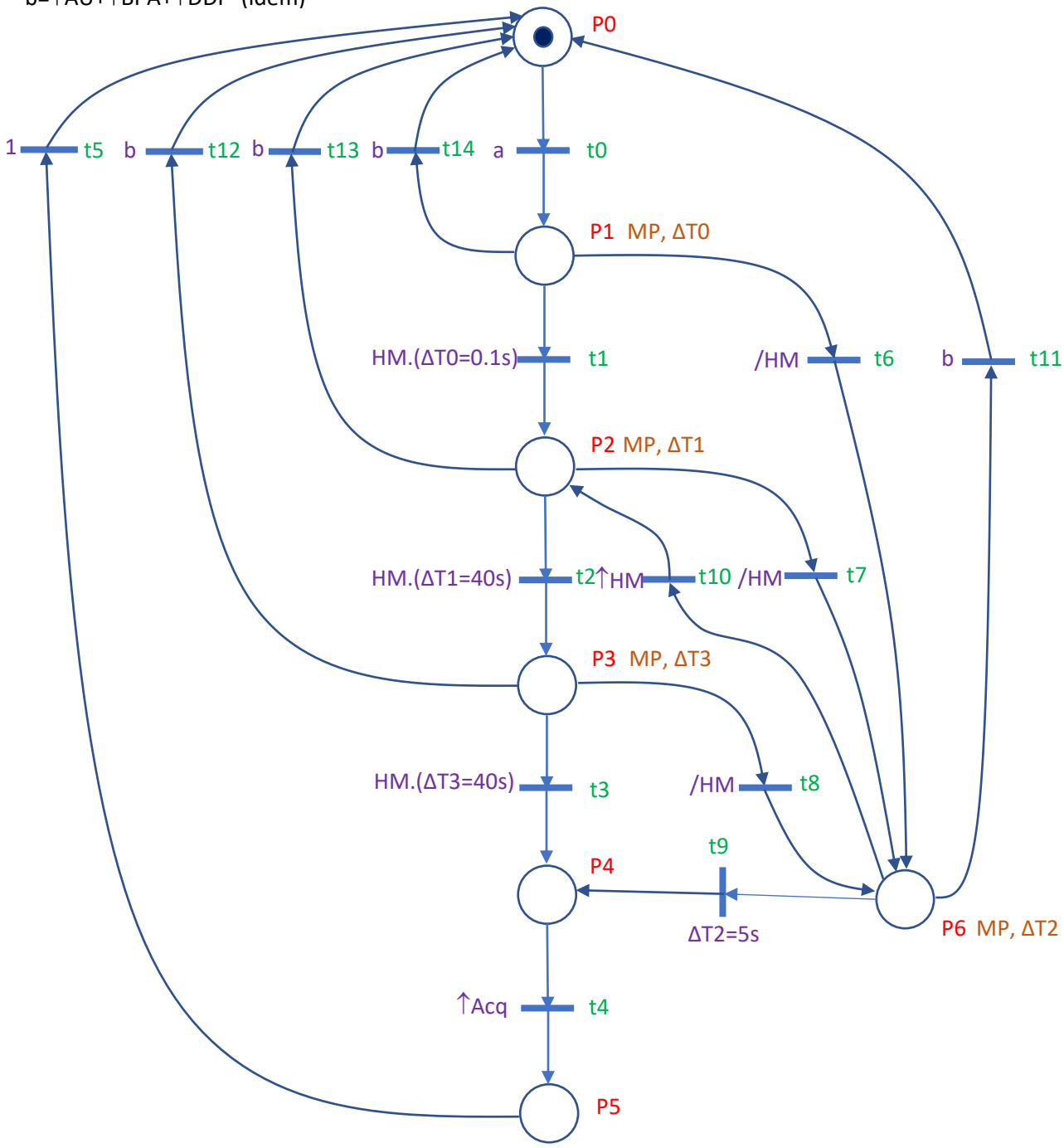
Sécurité 'Homme Mort'

Modèle RdP et codage Arduino sous Python, Laurent Tromp, Saint Louis, Paris

Sur une suggestion de Robert Papanicola, 16 janvier 2019

a=↑HM.BPM./AU./DHM./DPP (adaptation pour avoir un événement)

b=↑AU+↑BPA+↑DDP (idem)



Rappel :

- pour une place Pi de variable associée xi (état courant) et Xi (état suivant visé),
- ε_{xi} la condition de mise à 1 et δ_{xi} la condition de mise à 0 sur le principe de fonctionnement d'une bascule à marche prioritaire
- condition Ci associée à la transition ti, la fin d'une temporisation est un événement
- $Xi=\varepsilon_{xi} + xi./\delta_{xi}$

Par exemple $\varepsilon_{x4}=x3.C3+x6.C9$ et $\delta_{xi}=C4$

L'équation gouvernant la place Pi est donc X4 = x3. HM.(ΔT3=40s)+x6.(ΔT2=5s) + x4./↑Acq

```
# Système Homme Mort
# L. Tromp, SL, 16 janvier 2019

from py2duino import *             #importation du module de traduction
import time                         #module de gestion du temps
ar1=Arduino(4)                      #déclaration de la carte sur le port COM4
time.sleep(1);                      #tempo permettant la communication de s'établir

#Numéros des broches:
MP =  8;    #Pompe en marche
HM =  1     #Poignée HM
BPM = 2;    #Bouton poussoir marche
BPA = 3;    #Bouton poussoir marche
AU =  4;    #Arrêt urgence
DHM = 5;    #...
DPP = 6;    #...
Acq = 7;    #...

#Constantes
infini=pow(10,15);                  #grande durée pour les temporisations
F=False;                            #raccourcis
T=True;
tempo=[0.100,40.000,5.000,20.000];    #durées des temporisations
# Variables:
timer=[infini,infini,infini,infini];  #Initialisation des compteurs
x=[T,F,F,F,F,F,F];                    #Etat courant : seule la place P1 est marquée
(G7 : étape active initiale)
X=[T,F,F,F,F,F,F];                    #Etat suivant : l'état pris à la fin de
l'itération, après rétroaction de l'état

C=[F,F,F,F,F,F,F,F,F,F,F,F,F,F,F];    #événements graphe d'état : il y a 15
transitions et donc 15 événements

E=[F, F, F, F, F, F, F];        #valeurs initiales des 7 entrées
S=["LOW"];                      #Sorties graphe d'état : commande de marche de la
pompe

BPMprec=F;                      #initialisation des variables de mémorisation des
entrées, nécessaires pour construire les fronts si nécessaire
BPAprec=F;
HMprec=F;
AUprec=F;
DHMprec=F;
DPPprec=F;
Acqprec=F

ar1.pinMode(MP,     "OUTPUT")        #déclaration des broches
ar1.pinMode(BPM,    "INPUT")
ar1.pinMode(BPA,    "INPUT")
ar1.pinMode(HM,     "INPUT")
ar1.pinMode(AU,     "INPUT")
ar1.pinMode(DHM,    "INPUT")
ar1.pinMode(DPP,    "INPUT")
ar1.pinMode(Acq,    "INPUT")

while True:                             #Boucle sans fin

    E[0]=ar1.digitalRead(BPM);       #lecture du bouton poussoir
    E[1]=ar1.digitalRead(BPA);       #lecture du bouton poussoir
    E[2]=ar1.digitalRead(HM);       #lecture du bouton poussoir
    E[3]=ar1.digitalRead(AU);       #lecture du bouton poussoir
    E[4]=ar1.digitalRead(DHM);       #lecture du bouton poussoir
    E[5]=ar1.digitalRead(DPP);       #lecture du bouton poussoir
    E[5]=ar1.digitalRead(DPP);       #lecture du bouton poussoir
    E[6]=ar1.digitalRead(Acq);       #lecture du bouton poussoir

    FBPM=E[0] and not(BPMprec);     #construction du front montant BPM
```

```
    FBPA=E[1] and not(BPAprec);     #construction du front montant BPA
    FAU =E[3] and not(AUprec);      #construction du front montant AU
    FDPP=E[5] and not(DPPprec);     #construction du front montant DPP
    FAcq=E[6] and not(Acqprec);     #construction du front montant Acq

    BPMprec=E[0];                   #Mémorisation des entrées pour l'itération
suivante
    BPAprec=E[1];
    AUprec =E[3];
    DPPprec=E[5];
    Acqprec=E[6];

    ##### contruction des événements
    a= FBPM and HMprec and not AUprec and not DHMprec and not DPPprec   # Premier
évcénement
    b= FAU or FBPA or FDDP                                              # événement
d'arrêt

    if a : C[0] = T;              #pour la transition t0
    else: C[0] = F;

    if HMprec and (time.time()-timer[0]>tempo[0]) : C[1]= T;  timer[0]=infini;
#condition + fin de tempo t1
    else: C[1]= F;

    if HMprec and (time.time()-timer[1]>tempo[1]) : C[2]= T; timer[1]=infini;
#condition + fin de tempo t2
    else: C[2]= F;

    if HMprec and (time.time()-timer[3]>tempo[3]) : C[3]= T; timer[0]=infini;
#condition + fin de tempo t3
    else: C[3]= F;

    if FAcq  : C[4]= T;    #Front Acquittement
    else: C[4]= F;

    C[5] = T;     #franchement étrange, mais écrit comme ça, Il faudrait sans doute
une tempo minime pour assurer l'exécution des sorties...

    if not HM : C[6]=T ; C[7]=T ; C[8]=T;   #seulement une condition de garde
    else C[6]=F ; C[7]=F ; C[8]=F;

    if (time.time()-timer[2]>tempo[2])   : C[9]= T; timer[2]=infini;
#condition + fin de tempo t9
    else: C[9]= F;

    if HM : C[10]=T;
    else C[10]=F;

    if b : C[11]=T ; C[12]=T ; C[13]=T; C[14]=T;
    else C[11]=F ; C[12]=F ; C[13]=F; C[14]=F;


    ##### équations d'évolution de l'état de graphe d'état de commande des feux
Voiture
    X[0]= ((x[1] or x[2] or x[3] ) and b or x[5])) or x[0] and not C[0];
#Calcul place 0
    X[1]= (x[0] and C[0]) or x[1] and not C[1];
#Calcul place 1
    X[2]= (x[1] and C[1] or x[6] and C[10]) or x[2] and not (C[2] or C[7]);
#Calcul place 2
    X[3]= (x[2] and C[2]) or x[3] and not (C[3] or C[8]);
#Calcul place 3
    X[4]= (x[3] and C[3] or x[6] and C[9]) or x[4] and not C[4];
#Calcul place 4
    X[5]= (x[4] and C[4]) or x[5] and not C[5];
#Calcul place 5
```

```
    X[6]= ((x[1] or x[2] or x[3] ) and not HMprec or x[6] and not (C[9] or C[10] or
C[11]);     #Calcul place 6


    ##### activité interne: lancement des temporisations
    if (X[1] and not x[1]): timer[0]=time.time();  #tempo feu rouge sur front
    if (X[2] and not x[2]): timer[1]=time.time();  #tempo feu rouge sur front
    if (X[3] and not x[3]): timer[3]=time.time();  #tempo feu rouge sur front
    if (X[6] and not x[6]): timer[2]=time.time();  #tempo feu rouge sur front


##### activités externes: calcul des sorties : mise en marche de la pompe (faisable
par action mémorisée sur front)
    if x[1] or x[2] or x[3] or x[6]: S[0]="HIGH";
    else : S[0]="LOW";

    ##### émission des décisions : commande des ports
    ar1.digitalWrite(MP,S[0]);          #Affectation des sorties

    ##### Actualisation de l'état : l'état suivant devient le nouvel état
    for i in range(len(x)):
        x[i]=X[i];                      #Actualisation de l'état graphe d'état
Voiture
```