

PPE Convoyeur à bande
Annexes**prog_base_3.c**

```
/*
 * Auteur Romain BOCHET
 * P.P.E. Convoyeur a bande
 * Programme PIC C.N.A.
 * ****
 *prog_base_3.c *
 * ****
 */

#include std84.h          //declaration d'inclusion
#include bit84.h
#define sortie portb      //declaration d'équivalence
#define marche porta.1    //declaration bouton poussoir
marche

char cpt,                //compteur boucle accélération deceleration
      tempoAccelDecel, // tempo entre deux vitesses successives de
l'accélération ms
      tempoVitCons;      // tempo palier vitesse constante

void main()
{
    /*Initialisation des parametres de conception*/

    trisb=0x0;            //Configuration port b en sortie ==> vers le
C.N.A.
    trisa=0b10;           //Configuration RA1 en entrée
    cpt=0;                //compteur des rampes accélération et
deceleration
    sortie=0;             //initialisation de la sortie a 0 ==> aucune
activité

    /*Initialisation des parametres de reglage*/

    tempoAccelDecel=25;    // temporisation par palier
acceleration deceleration millisecondes
    tempoVitCons=5;        //declaration de la tempo 2 en secondes

    /*****Debut boucle infinie*****/

    for(;;)
    {
```

PPE Convoyeur à bande
Annexes

```

/*****Attente marche*****/
while(!marche)
{
    delay(0);
    sortie=0;
}
// Fin while(marche)
/*****Rampe acceleration*****/
for(cpt=0;cpt<255;cpt++)
{
    sortie=cpt+1;    //car cpt max = 254
    microdelay(tempoAccelDecel);
    microdelay(tempoAccelDecel);
    microdelay(tempoAccelDecel);
    microdelay(tempoAccelDecel);
}
/*****Palier de vitesse*****/
delays(tempoVitCons);
/*****Rampe de deceleration*****/
for(cpt=sortie;cpt>0;cpt--)
{
    sortie=cpt;
    microdelay(tempoAccelDecel);
    microdelay(tempoAccelDecel);
    microdelay(tempoAccelDecel);
    microdelay(tempoAccelDecel);
}

}

//fin for(;;)
}
//fin void main()
```

PPE Convoyeur à bande
Annexes**codeuse_b_20.c**

```
/*
 * Auteur Romain BOCHET
 * P.P.E. Convoyeur a bande
 * Programme PIC C.N.A.
 *
 *codeuse_b_20.c *
 */

#include std84.h          //declaration d'inclusion
#include bit84.h
#define sortie portb      //declaration d'équivalence
#define marche porta.0    //declaration bouton poussoir
marche
#define codeuse_v1 porta.1 //Declaration des quatres voies
de la roue codeuse
#define codeuse_v2 porta.2
#define codeuse_v3 porta.3
#define codeuse_v4 porta.4

char cpt,                //compteur boucle accélération deceleration
      tempoAccel,        // tempo entre deux vitesses successives de
l'accélération ms
      tempoDecel,        // tempo entre deux vitesses successives de la
decelération ms
      tempoVitCons,      // tempo palier vitesse constante
      sp,
      codeuse;           // stockage de la valeur de la roue codeuse

void main()
{
    /*Initialisation des parametres de conception*/

    trisb=0x0;           //Configuration port b en sortie ==> vers le
C.N.A.
    trisa=0x1F;          //Configuration porta en entrée ==> bp marche +
roue codeuse
    cpt=0;               //compteur des rampes accélération et
deceleration
    sortie=0;            //initialisation de la sortie a 0 ==> aucune
```

PPE Convoyeur à bande

Annexes

activité

```

/*Initialisation des parametres de reglage*/

tempoAccel=25;      // temporisation par palier acceleration
millisecondes
tempoDecel=25;      // temporisation par palier deceleration
millisecondes
tempoVitCons=5;     //declaration de la tempo 2 en secondes

/*****Debut boucle infinie*****/

for(;;)
{
    /*****Attente marche*****/
    while(!marche)
    {
        delay(0);
    }
    // Fin while(marche)
    /*****Calcul de la tempo*****/
    codeuse = lirecodeusedecal();
    tempoAccel = calculTempoAccel(codeuse);
    tempoDecel = calculTempoDecel(codeuse);
    /*****Rampe acceleration*****/
    for(cpt=0;cpt<255;cpt++)
    {
        sortie=cpt+1;      //car cpt max = 254
        delay(tempoAccel);
    }
    /*****Palier de vitesse*****/
    //sortie = tempoDecel;      //testrb      affichage tempo
    decel
    delays(tempoVitCons);
    /*****Rampe de deceleration*****/
    for(cpt=sortie;cpt>0;cpt--)
    {
        sortie=cpt;
        delay(tempoDecel);
    }

}

//fin for(;;)
}
//fin void main()

/*****
*****
***** fonction lirecodeusedecal()

```

PPE Convoyeur à bande
Annexes

```

*****
*****      lit l'etat de la roue codeuse
*****
*****      decalage de bit
*****
*****
*****/
char lirecodeusedecal()
{
    char cod;
    cod = 0;
    if(codeuse_v4)
    {   cod++;}
    cod = cod<<1;
    if(codeuse_v3)
    {   cod++;}
    cod = cod<<1;
    if(codeuse_v2)
    {   cod++;}
    cod = cod<<1;
        if(codeuse_v1)
    {   cod++;}

    return(cod);
}

/*****
*****
*****      fonction lirecodeuseaddi()
*****
*****      lit l'etat de la roue codeuse
*****
*****      utilisation d'adition de bit
*****
*****
*****/
char lirecodeuseaddi()
{
    char codif;
    codif=0;           // initialisation de codeuse
    if(codeuse_v1)
    {
        codif=codif+1; //Ajout du pids des bits respectifs
    }
    if(codeuse_v2)
    {
        codif=codif+2;
    }
}

```

PPE Convoyeur à bande
Annexes

```

    }
    if(codeuse_v3)
    {
        codif=codif+4;
    }
    if(codeuse_v4)
    {
        codif=codif+8;
    }
    return(codif);
}
/*****
*****
***** fonction calculTempoAccel(codeuse)
*****
***** calcule la tempo de la rampe d'acc.
*****
***** variable d'entrée :
*****
***** ->codeuse : etat de la roue codeuse
*****
***** variable de sortie :
*****
***** ->tempoAccel: tempo rampe accel.
*****
*****
*****/
char calculTempoAccel(char codeuse)
{
    char temp;                //Variable temporaire definie localement
    temp=0;
    if(codeuse==0)
    {temp=0 ;}
    if(codeuse==1)
    {temp=1 ;}
    if(codeuse==2)
    {temp=2 ;}
    if(codeuse==3)
    {temp=3 ;}
    if(codeuse==4)
    {temp=4 ;}
    if(codeuse==5)
    {temp=6 ;}
    if(codeuse==6)
    {temp=8 ;}
    if(codeuse==7)
    {temp=10 ;}

```

PPE Convoyeur à bande
Annexes

```

    if(codeuse==8)
    {temp=12 ;}
    if(codeuse>=9)
    {temp=14 ;}
    return(temp);
}
/*****
*****
*****      fonction calculTempoDecel(codeuse)
*****
*****      calcule la tempo de la rampe de dec
*****
*****      variable d'entrée :
*****
*****      ->codeuse : etat de la roue codeuse
*****
*****      variable de sortie :
*****
*****      ->tempoDecel:tempo de la rampe dec.
*****
*****/
char calculTempoDecel(char codeuse)
{
    char temp;                //Variable temporaire definie localement
    if(codeuse==0)
    {temp=0 ;}
    if(codeuse==1)
    {temp=1 ;}
    if(codeuse==2)
    {temp=2 ;}
    if(codeuse==3)
    {temp=3 ;}
    if(codeuse==4)
    {temp=4 ;}
    if(codeuse==5)
    {temp=5 ;}
    if(codeuse==6)
    {temp=7 ;}
    if(codeuse==7)
    {temp=9 ;}
    if(codeuse==8)
    {temp=11 ;}
    if(codeuse>=9)
    {temp=13 ;}
    return(temp);
}

```

PPE Convoyeur à bande
Annexes**phase_6_complet.c**

```
/*
 * Auteur Romain BOCHET
 * P.P.E. Convoyeur a bande
 * Programme PIC C.N.A.
 ****
 *phase_6_complet.c *
 ****

 */

#include std84.h          //declaration d'inclusion
#include bit84.h
#define sortie portb      //declaration d'équivalence

#define entree porta

char
    vitesse,           // vitesse intantanee
    deplacement,
    vitesse_max,        //vitesse maxi (0 --> 255
    vitesse_arret ,     // vitesse d'arret direct
    rampe_acc[10] = {1,5,10,15,20,40,50,60,70,80},
    rampe_dec[10] = {80,70,55,40,33,22,11,5,3,1},
    deplacement_max[10] = {10,10,10,10,100,100,100,200,200,200},
    tempo_palier,
    bp_marche,          // état du bouton poussoir marche/arret
    bp_marche_precedent, // état precedent du bouton marche
    phase ,             // 0 : arret, 1 :accel, 2 vit constante
, 3 : dec

    tempoAccel,         // tempo entre deux vitesses successives de
l'accélération ms
    tempoDecel,         // tempo entre deux vitesses successives de la
decelération ms
    tempoVitCons,       // tempo palier vitesse constante
    sp,
    codeuse;            // stockage de la valeur de la roue codeuse

void main()
{
    /*Initialisation des parametres de conception*/
```

PPE Convoyeur à bande

Annexes

```

    trisb=0x0;          //Configuration port b en sortie ==> vers le
C.N.A.
    trisa=0x1F;         //Configuration porta en entrée ==> bp marche +
roue codeuse
    sortie=0;           //initialisation de la sortie a 0 ==> aucune
activité

                        /*Initialisation des parametres de reglage*/

//rampe_a[]= {16, 240};

tempo_palier = 10;
vitesse_max = 63;      // vitesse maxi atteinte par la bande
vitesse_arret = 3;
vitesse = 0 ;

lecture_entree(porta); // suppression du font initial ;
/*****Debut boucle infinie*****/
sortie = 0; phase = 0;
    for(;;)
    {
        /*****Attente marche*****/
        lecture_entree(porta);
        evoluer_phase();
        vitesse = piloter_vitesse(vitesse);
        //sortie = phase ; //vitesse;
        sortie = vitesse;

        delay();

    }
                                //fin for(;;)
}                                //fin void main()

/*****
*****
*****      lecture des entrées et decodage
*****
*****
*****
*****      variable d'entree : port A
*****
*****      affecte les variables globales

```

PPE Convoyeur à bande

Annexes

```

*****
*****                               : bouton_marche
*****
*****                               : codeuse
*****
*****
*****/
void lecture_entree(char port)
{
    codeuse = port>>1;           // suppression du bit porta.0 (bouton
marche)

    bp_marche_precedent = bp_marche; // enregistrement de l'etat
precedent du bouton marche
    bp_marche = port & 1 ;        // extraction du bit porta.0
                                /****** calcul des fronts
                                * montant          :
((bp_marche==1)&&(bp_marche_precedent==0))
                                * descendant      :
((bp_marche==0)&&(bp_marche_precedent==1))
                                */
} // fin : lecture_entree(char port)

/*****
*****
*****          changer de phase de fonctionnement
*****
*****
*****
***** variable d'entree : aucune
*****
***** affecte la variable globale
*****
*****          : phase
*****
*****
*****/
void evoluer_phase()
{
    switch (phase)                //structure de test a multiples choix
    {
        case 0 :{
            if ((bp_marche==1)&&(bp_marche_precedent==0)) //si
front montant bp
                {phase = 1;           //vers la rampe d'accélération

```

PPE Convoyeur à bande
Annexes

```

    }
    break;}          //fin de la structure de test

    case 1 :{
        if (vitesse>= vitesse_max ) //(vitesse >= vitesse_max)
// fin accel --> vitesse constante
        { phase = 2;
        }
        if ((bp_marche==1)&&(bp_marche_precedent==0)) // demande
arret --> decel
        { phase = 3;          //deceleration
        }

        break;}

    case 2 :{
        // fin plateau
        if (deplacement>=deplacement_max[codeuse])          //si fin
vitesse plateau
        {
            phase = 3;
        }

        if ((bp_marche==1)&&(bp_marche_precedent==0)) // demande
arret --> decel
        { phase = 3;
        }

        break;}
    case 3 :{
        if (vitesse<=vitesse_arret) //(vitesse >= vitesse_arret)
// fin accel --> vitesse constante
        { phase = 0;
        }
        // introduire possibilité reaccélération passage en ph1
        break;}

    }

}

/*****
*****
*****      determiner nouvelle vitesse
*****
*****/

```

PPE Convoyeur à bande Annexes

```

*****
*****
***** variable d'entree : vitesse
instantanée*****
***** renvoie la nouvelle vitesse
*****
*****/

```

```

char piloter_vitesse(char vit )
{
    if (phase==0)          //attente de bouton Dcy
    {
        vit = 0;
        deplacement = 0;
    }
    if ((phase==1)&&(vit<vitesse_max))          //premiere phase
d'accélération
    {
        vit++;
        delay(rampe_acc[codeuse]);
    }
    if ((phase==2))          //phase de palier
    {
        vit = vitesse_max;
        deplacement++;
        delay(tempo_palier);
    }

    if (phase == 3)          //phase de décélération ou d'arret
apres bp Dcy
    {
        if (vit <= vitesse_arret)
        {
            vit = 0;
        }
        else
        {
            vit--;
        }

        delay( rampe_dec[codeuse]);
    }
    return  vit;
}

```

PPE Convoyeur à bande
Annexes

```
    /* switch (phase)
{
    case 0 :{
        vit = 0;
        break;}
} */
```