```
/* -------------------------------------------------------------------------
 * Carte LAUCHPAD Rev.1.5 - MSP430G2553 - Yann DUCHEMIN/ESIGELEC
 * Project demo for Servo Motor Control - Rev.27XII2018
 * Motor GWS S03N or similar (i.e. Futaba 3003)
 * Vcc (3.3V) & GND, Command
 * -------------------------------------------------------------------------
 *                   MSP430G2553 / Launchpad Rev.1.5
 *               +-------------------------+
 *         |  1 |Vcc        \/         Vss | 20 |
 *  P1.0   |  2 |LED-1                    | 19 |P2.6 TA0.1
 *  P1.1   |  3 |Rx - Tx-RN42             | 18 |P2.7
 *  P1.2   |  4 |Tx - Rx-RN42             | 17 |Tst
 *  P1.3   |  5 |                         | 16 |!RST
 *  P1.4   |  6 |                         | 15 |P1.7
 *  P1.5   |  7 |TA0.0                LED-2| 14 |P1.6 TA0.1
 *  P2.0   |  8 |TA1.0                    | 13 |P2.5 TA1.2
 *  P2.1   |  9 |TA1.1                    | 12 |P2.4 TA1.2
 *  P2.2   | 10 |TA1.1 [MOTEUR 1]         | 11 |P2.3 TA1.0
 *               +-------------------------+
 * -------------------------------------------------------------------------
 */
#define     MOTORS_FREQUENCY        33333    // motors period T=30 ms (0.0333 s) soit 30 Hz
#define     MOTORS_DUTYCYCLE        1250     // motors duty cycle 50% soit 0.00125 ms 1250 µs
#define     STEP_ANGLE              72       // step for 1 deg  PW[500-3000 µs]/180 deg
#define     PW_MIN                  504      // 500/72=7 7*72=504
#define     PW_MAX                  2448     // 3000/72=41 41*72=2952

/*
 * Headers
 */
#include <msp430.h>

/*
 * Prototypes
 */
void init_Board( void );
void init_Timer( void );

/*
 * Variables Globales pour interruptions
 */
unsigned int up = 0;    // sens de variation
unsigned int cmd = 0;   // periode du signal de commande moteur

/* -------------------------------------------------------------------------
 * Fonction d'initialisation de la carte TI LauchPAD
 * Entree : -
 * Sorties:  -
 */
void init_Board( void )
{
  // Arret du "watchdog" pour eviter les redemarrages sur boucles infinies
  WDTCTL = WDTPW | WDTHOLD;

  // Calibration usine de l'oscillateur numerique interne
  if(CALBC1_1MHZ==0xFF || CALDCO_1MHZ==0xFF)
    __bis_SR_register(LPM4_bits);
  else
  {
    DCOCTL = 0;
    BCSCTL1 = CALBC1_1MHZ;
    DCOCTL = CALDCO_1MHZ;
  }

  //--- Securisation des entrees/sorties
  P1SEL  = 0x00;        // GPIO
  P1SEL2 = 0x00;        // GPIO
  P2SEL  = 0x00;        // GPIO
  P2SEL2 = 0x00;        // GPIO
  P1DIR = 0x00;         // IN
```

```
71      P2DIR = 0x00;           // IN
72      //---
73
74      P1SEL &= ~(BIT0 | BIT6);  // Port 1, ligne 0 et 6 en fonction primaire
75      P1SEL2 &= ~(BIT0 | BIT6); // GPIO
76      P1DIR |= (BIT0 | BIT6 );  // P1.0 et P1.6 en sortie
77      P1OUT &= ~(BIT0 | BIT6);  // P1.0 et P1.6 à 0
78
79      P1DIR &=~BIT3;        // Port 1 ligne 3 en entrée
80      P1REN |= BIT3;        // Activation de la resistance de tirage
81      P1OUT |= BIT3;        // Resistance en Pull-Up
82      P1IES &=~BIT3;        // Detection de front montant
83      P1IE |= BIT3;         // Activation des interruptions sur P1.3
84
85      P2SEL |= BIT2;        // Port 2, ligne 2 en fonction secondaire
86      P2SEL2 &= ~BIT2;      // Timer
87      P2DIR |= BIT2;        // Port 2, ligne 2 en sortie
88    }
89
90    /* -------------------------------------------------------------------------
91     * FONCTION D'INITIALISATION DU TIMER
92     * Entree : -
93     * Sorties:  -
94     */
95    void init_Timer( void )
96    {
97       TA1CTL &= ~MC_0;                                    // arret du timer
98       TA1CCR0 = MOTORS_FREQUENCY;                         // periode du signal PWM 2KHz
99       TA1CTL = (TASSEL_2 | MC_1 | ID_0 | TACLR);          // select TimerA source
         SMCLK, set mode to up-counting
100      TA1CCTL1 = 0 | OUTMOD_7;                            // select timer compare mode
101   }
102
103   /* -------------------------------------------------------------------------
104    *  Fonction Principale
105    */
106   void main(void)
107   {
108       init_Board();
109       init_Timer();
110
111       cmd = MOTORS_DUTYCYCLE;
112       up = 1;
113
114       TA1CCR1 = cmd;
115
116       __bis_SR_register(LPM0_bits | GIE);      // general interrupts enable & Low
          Power Mode
117   }
118
119   /* ********************************************************************** */
120   /* VECTEUR INTERRUPTION PORT 1                                            */
121   /* ********************************************************************** */
122   #pragma vector = PORT1_VECTOR
123   __interrupt void PORT1_ISR(void)
124   {
125     if( P1IFG & BIT3)                     // interruption Entree/sortie Port 1 ligne 3
126     {                                     // si appui sur le bouton P1.3
127       if( !up )                           // Sens décroissant
128       {
129           P1OUT &=~BIT6;                  // Eteindre la Led du port 1 ligne 6
130           if (cmd > (PW_MIN+STEP_ANGLE) ) // Si Période mini non encore atteinte
131           {
132               cmd -= STEP_ANGLE;          // Décrémenter la période
133           }
134           else                            // Sinon
135           {
136               cmd = PW_MIN;               // Ajuster la période
137               up = 1;                     // Changer le sens de boucle
138           }
139           P1OUT ^= BIT0;                  // Faire clignoter la Led a chaque itération
140       }
141       else                                // Sinon
```

```
142       {
143           P1OUT &=~BIT0;                    // Eteindre la Led de P1.0
144           if(cmd < (PW_MAX-STEP_ANGLE) )    // Si Période inférieure au max
145           {
146              cmd += STEP_ANGLE;             // Augmenter la période
147           }
148          else                               // Sinon
149           {
150              cmd = PW_MAX;                  // Ajuster la période
151              up = 0;                        // Inverser le sens de boucle
152           }
153           P1OUT ^= BIT6;                    // Faire clignoter la Led
154       }
155       TA1CCR1 = cmd;                        // Modifier la valeur de commptage Timer
          (Rapport Cyclique)
156     }
157     P1IFG &= ~BIT3;                         // Acquiter l'interruption
158   }
```