Annexes

I – Programme du système émetteur

II – Programme du système récepteur

III – Logiciel de traitement des fichiers

I – Programme du système émetteur

```
//Variables relatives à la carte SD :
const int chipSelect = 8;
#include <SdFat.h>
SdFat sd;
SdFile myFile;
int data;
String ligne="";

// Constantes :
const int laserzero = 3;
const int laserun = 2;
const int ledtemoin = 5;
const int int_emission = 7; //Lance l'émission
const int int_etalonnage = 6; //Lance l'étalonnage
const int freq=50; //temps d'allumage des lasers

//Variables
int taille;
String caractere;
String caracterebin;
int i;
int j;
String motbin;
int taillebin;
int etatint_emission = 0;
int etatint_etalonnage = 0;
int phaseemission=0;
int chrono;
int delai_int=500;

//Fonction setup
void setup() {
  Serial.begin(9600); //Début de la communication serial
  Serial.println("Systeme de tranmission laser.");
  Serial.println("");
  if (!sd.begin(chipSelect, SPI_FULL_SPEED)) sd.initErrorHalt(); //Initialisation de la communicat
  // On intialise les pins
  pinMode(laserzero, OUTPUT);
  pinMode(laserun, OUTPUT);
  pinMode(ledtemoin, OUTPUT);
  pinMode(int_emission, INPUT);
  pinMode(int_etalonnage, INPUT);
  digitalWrite(laserzero, HIGH); //Eteint les lasers
  digitalWrite(laserun, HIGH);
}

//Boucle principale
void loop() {
  if ((digitalRead(int_etalonnage) == HIGH) && (phaseemission == 0) && (millis()-chrono>delai_int)
    digitalWrite(laserzero, !digitalRead(laserzero)); //Toggle sur les lasers
    digitalWrite(laserun, !digitalRead(laserun));
    chrono=millis();
       if ((digitalRead(laserzero)==HIGH) && (digitalRead(laserun)==HIGH)) { //Si les deux sont ét
       digitalWrite(ledtemoin,LOW);
        }
  else {
    digitalWrite(ledtemoin,HIGH); //Sinon on l'allume
  }
  Serial.println("Action volontaire sur les lasers.");
  delay(100);
}
```

I – Programme du système émetteur

```
  if ((digitalRead(int_emission) == HIGH) && (phaseemission == 0)) { //si le bouton est enfonc
    digitalWrite(ledtemoin,HIGH);  //On allume la led témoin d'émission
    phaseemission=1;
      if (!myFile.open("emission.txt", O_READ)) { //Ouverture du fichier en lecture seule
        sd.errorHalt("Erreur d'ouverture du fichier en lecture seule.");
      }
      while ((data = myFile.read()) >= 0) { //tant qu'on ne lit pas un truc vide
        ligne = ligne + String(char(data)); //On convertit un int en char puis en String et on
          if (data==10) { //si retour à la ligne
          transmission(ligne,1); //On transmet la ligne en mode 1 (supression du retour à la l
          ligne="";
          }
      }
    transmission(ligne,0); //On transmet la dernière ligne en mode 0
    ligne="";
    digitalWrite(ledtemoin,LOW); //On éteint la led témoin
    Serial.println("Fin de tranmission.");
    myFile.close(); //Permet de pouvoir recommencer l'envoi
  }
}

//Fonction transmission
void transmission(String chaine, int type_ligne) {
  taille=chaine.length(); //Taile de la chaine à transmettre
  Serial.println("Debut de transmission pour "+chaine+" :");
  for(j=1;j<=(taille-type_ligne);j++) { //pour chaque caractere ascii (type_ligne=0 pour une l
    caractere = chaine.substring(j-1,j); //on deplace le curseur sur les différents caracteres
    if ((j==taille-type_ligne) && (type_ligne==1)) {caractere="+";} //Gestion de la transmissi
    motbin="111111"; //Au cas où le caratère ne serait pas reconnu, on décode alors (caractere
//Tests sur les valeurs ascii - Caractères notables non gérés : à,é,è,ê,U,W,X,Y,Z,!,: dû à la
if (caractere == "A") {
    motbin="000000";
  }

if (caractere == "a") {
    motbin="000001";
  }
if (caractere == "à") { //transmis comme un a
    motbin="000001";
  }
if (caractere == "B") {
    motbin="000010";
  }
if (caractere == "b") {
    motbin="000011";
  }
if (caractere == "C") {
    motbin="000100";
  }
if (caractere == "c") {
    motbin="000101";
  }
if (caractere == "D") {
    motbin="000110";
  }
if (caractere == "d") {
    motbin="000111";
  }
if (caractere == "E") {
    motbin="001000";
  }
if (caractere == "e") {
    motbin="001001";
```

I – Programme du système émetteur

```
if (caractere == "u") {
    motbin="101000";
  }
if (caractere == "V") {
    motbin="101001";
  }
if (caractere == "v") {
    motbin="101010";
  }
if (caractere == "W") { //transmis comme un w
    motbin="101011";
  }
if (caractere == "w") {
    motbin="101011";
  }
if (caractere == "X") { //transmis comme un x
    motbin="101100";
  }

if (caractere == "x") {
    motbin="101100";
  }
if (caractere == "Y") { //transmis comme un y
    motbin="101101";
  }
if (caractere == "y") {
    motbin="101101";
  }
if (caractere == "Z") { //transmis comme un z
    motbin="101110";
  }
if (caractere == "z") {
    motbin="101110";
  }

if (caractere == "+") { //Correspond au retour à la ligne
    motbin="111110";
  }
//Fin du test sur les valeurs ascii

    taillebin=motbin.length(); //Taille du caractere en binaire
    Serial.println("Transmission de "+caractere+" "+"("+motbin+").");
      for(i=1;i<=taillebin;i++) { //Boucle pour un curseur se déplaçant le long du caractere bir
        caracterebin = motbin.substring(i-1,i); //On isole un bit
         if (caracterebin == "0") { //Si c'est un 0
            digitalWrite(laserzero,LOW);
            delay(freq); //Temps d'allumage de la LED
            digitalWrite(laserzero,HIGH);
          }
          else {
            digitalWrite(laserun,LOW);
            delay(freq);
            digitalWrite(laserun,HIGH);
          }
      }
  }
  phaseemission=0;
  chrono=0;
  Serial.println("");
}
```

I – Programme du système récepteur

```
//Variables relatives à la carte SD :
const int chipSelect = 8;
#include <SdFat.h>
SdFat sd;
SdFile myFile;

// Constantes :
const int photo0 = A3; //Pin analogique de la photorésistance codant pour les 0
const int photo1 = A2;
const int intreglage=11;
const int boutonlum=12;
const int ledreg=8; //rouge
const int ledtemoin=9; //verte
const int freq=20; //temps d'allumage de la led


//variables
String motbin;
int val1 = 0;
int val0 = 0;
int temps0=0;
int temps1=0;
int j;
int taille;
int nbcar;
String caracterebin="";
String caract;
String mot;
int phaserecept=0;
int phasereglage=0;
float chrono=0;
float timer;
int rien0;
int rien1;
int lumiere0=0;
int lumiere1=0;
int seuil0 = 1000;
int seuil1 = 1000;
int timerbouton;
int delai;


void setup() {
Serial.begin(9600);
if (!sd.begin(chipSelect, SPI_FULL_SPEED)) sd.initErrorHalt(); //Initialisation de la communicati
if (!myFile.open("recept.txt", O_RDWR | O_CREAT | O_AT_END)) {
    sd.errorHalt("Echec de l'ouverture de recept.txt en écriture.");
  }
  pinMode(photo0, INPUT);
  pinMode(photo1, INPUT);
  pinMode(intreglage, INPUT);
  pinMode(boutonlum, INPUT);
  pinMode(ledreg, OUTPUT);
  pinMode(ledtemoin, OUTPUT);
  digitalWrite(ledreg, HIGH);
  delai=5*freq; //Délai avant de déclarer la fin de la réception
}


void loop() {

  if ((digitalRead(intreglage) == HIGH) && (phaserecept == 0) && (millis()-timerbouton>500)) {
    timerbouton=millis();
    rien0=analogRead(photo0);
    rien1=analogRead(photo1);
    Serial.println ("Debut des reglages..");
    Serial.println ("Valeurs des absences de lumieres enregistrees.");
    delay(300);
    phasereglage=1;

  }
```

I – Programme du système récepteur

```
  if ((digitalRead(boutonlum) == HIGH) && (phaserecept == 0) && (phasereglage==1) && (millis()-timerb
    timerbouton=millis();
    lumiere0=analogRead(photo0);
    lumiere1=analogRead(photo1);
    Serial.println ("Valeur des presences de lumiere enregistrees.");
  }

  if ((lumiere0>0) && (lumiere1>0) && (phasereglage==1) && (phaserecept==0)) {
    phasereglage=0;
    digitalWrite(ledreg, LOW);
    seuil0=int(3*((lumiere0-rien0)/4))+rien0;
    seuil1=int(3*((lumiere1-rien1)/4))+rien1;
    Serial.println ("Seuils de detection operationnels.");
    Serial.println ("Eteignez les lasers.");
      while ((analogRead(photo0)>seuil0)||(analogRead(photo1)>seuil1)) {
        //Attente
      }
    Serial.println ("Pret pour la reception.");
    Serial.println ("");
  }

  if ((phaserecept == 1) && (digitalRead(ledtemoin)==LOW)) {
    digitalWrite(ledtemoin, HIGH);
    Serial.println ("Reception en cours...");
    Serial.println ("");
  }

  val1 = analogRead(photo1);
  val0 = analogRead(photo0);

  if((val1>=seuil1) && (millis()-temps1>freq) && (phasereglage==0)){ //Si le seuil de détection est
    if (chrono==0) {chrono=temps1;}
    phaserecept=1;
    Serial.println ("1 recu");
    temps1=millis();
    motbin=motbin + "1";
  }

  if((val0>=seuil0) && (millis()-temps0>freq) && (phasereglage==0)){ //Si le seuil de détection est
    if (chrono==0) {chrono=temps0;}
    phaserecept=1;
    Serial.println ("0 recu");
    temps0=millis();
    motbin=motbin + "0";
  }

  if(phaserecept==1) {
    if(millis()-temps1>delai) {
        if(millis()-temps0>delai) {
          Serial.println("Resultat de la reception : "+motbin);
          phaserecept=0;
          digitalWrite(ledtemoin, LOW);
          taille=motbin.length();
          nbcar=taille/6;

//DECODAGE
for(j=1;j<=nbcar;j++) { //pour chaque caractere ascii
  decodage(motbin.substring((j-1)*6,6*j));
  if (caract=="+") {
    //Détection d'un retour à la ligne
    caract="";
    myFile.println(mot); //Ecriture sur la carte SD
    mot="";
  }
  else {
    mot=mot+caract;
  }
}
```

I – Programme du système récepteur

```
chrono=0;
motbin="";
mot="";
temps0=millis();
temps1=millis();
}
}
}
}


//Fonction décodage :
void decodage(String caracterebin) {

if (caracterebin == "000000") {
    caract="A";
  }
if (caracterebin == "000001") {
    caract="a";
  }
if (caracterebin == "000010") {
    caract="B";
  }
if (caracterebin == "000011") {
    caract="b";
  }
if (caracterebin == "000100") {
    caract="C";
  }
if (caracterebin == "000101") {
    caract="c";
  }
```

```
if (caracterebin == "111101") {
    caract="'";
  }
if (caracterebin == "111110") {
    caract="+";
  }
if (caracterebin == "111111") {
    caract="(caractere non reconnu)";
  }
}
```

III – Logiciel de traitement des fichiers

```
#include <ButtonConstants.au3>
#include <ComboConstants.au3>
#include <EditConstants.au3>
#include <GUIConstantsEx.au3>
#include <ProgressConstants.au3>
#include <StaticConstants.au3>
#include <WindowsConstants.au3>
#include <Array.au3>
#include <File.au3>
#include <Crypt.au3>
#include <Bitmap_String.au3>
#include <ScreenCapture.au3>

Global $input_fichier, $fichier, $encod_crypt, $decod_crypt, $fichier_sortie_enc = @WorkingDir &
Global $contenu_fichier_clair, $contenu_fichier_encode, $contenu_fichier_sortie_enc, $ligne_fichi

#region ### START Koda GUI section ### Form=
Global $Form1 = GUICreate("Transfert de données par laser.", 525, 421)
Global $Tab1 = GUICtrlCreateTab(0, 0, 527, 421)
GUICtrlCreateTabItem("Encoder un fichier")
GUICtrlSetFont(-1, 14, 400, 0, "Calibri (Corps)")
Global $Label1 = GUICtrlCreateLabel("Sélection du fichier :", 32, 40, 177, 27)
GUICtrlSetFont(-1, 14, 400, 0, "Calibri (Corps)")
Global $Input1 = GUICtrlCreateInput("", 216, 40, 201, 32)
GUICtrlSetFont(-1, 14, 400, 0, "Calibri (Corps)")
Global $Button1 = GUICtrlCreateButton("...", 432, 40, 51, 33)
GUICtrlSetFont(-1, 16, 400, 0, "Calibri (Corps)")
Global $Checkbox1 = GUICtrlCreateCheckbox("Crypter le fichier", 32, 96, 193, 33)
GUICtrlSetFont(-1, 14, 400, 0, "Calibri (Corps)")
Global $Label2 = GUICtrlCreateLabel("Algorithme de cryptage :", 96, 136, 213, 27)
GUICtrlSetFont(-1, 14, 400, 0, "Calibri (Corps)")
Global $Combo1 = GUICtrlCreateCombo("", 328, 136, 145, 25, BitOR($CBS_DROPDOWN, $CBS_AUTOHSCROLL)
GUICtrlSetData(-1, "AES (128bit)|AES (192bit)|AES (256bit)|RC4", "RC4")
GUICtrlSetFont(-1, 14, 400, 0, "Calibri (Corps)")
Global $Label3 = GUICtrlCreateLabel("Clé de cryptage :", 96, 192, 148, 27)
GUICtrlSetFont(-1, 14, 400, 0, "Calibri (Corps)")
Global $Input2 = GUICtrlCreateInput("", 256, 192, 121, 31, BitOR($GUI_SS_DEFAULT_INPUT, $ES_PASSW
GUICtrlSetFont(-1, 14, 400, 0, "Calibri (Corps)")
Global $Progress1 = GUICtrlCreateProgress(40, 248, 446, 65)
Global $Button2 = GUICtrlCreateButton("Encoder", 168, 336, 193, 65)
GUICtrlSetFont(-1, 28, 400, 0, "Calibri (Corps)")
Global $Tab2 = GUICtrlCreateTabItem("Décoder un fichier")
GUICtrlSetFont(-1, 14, 400, 0, "Calibri (Corps)")
Global $Label22 = GUICtrlCreateLabel("Algorithme de cryptage :", 96, 136, 213, 27)
GUICtrlSetFont(-1, 14, 400, 0, "Calibri (Corps)")
Global $Combo12 = GUICtrlCreateCombo("", 328, 136, 145, 25, BitOR($CBS_DROPDOWN, $CBS_AUTOHSCROLL)
GUICtrlSetData(-1, "AES (128bit)|AES (192bit)|AES (256bit)|RC4", "RC4")
GUICtrlSetFont(-1, 14, 400, 0, "Calibri (Corps)")
Global $Label32 = GUICtrlCreateLabel("Clé de cryptage :", 96, 192, 148, 27)
GUICtrlSetFont(-1, 14, 400, 0, "Calibri (Corps)")
Global $Input22 = GUICtrlCreateInput("", 256, 192, 121, 31, BitOR($GUI_SS_DEFAULT_INPUT, $ES_PASSW
GUICtrlSetFont(-1, 14, 400, 0, "Calibri (Corps)")
Global $Progress12 = GUICtrlCreateProgress(40, 248, 446, 65)
Global $Button22 = GUICtrlCreateButton("Décoder", 168, 336, 193, 65)
GUICtrlSetFont(-1, 28, 400, 0, "Calibri (Corps)")
GUICtrlSetState($Label2, $GUI_DISABLE)
GUICtrlSetState($Combo1, $GUI_DISABLE)
GUICtrlSetState($Label3, $GUI_DISABLE)
GUICtrlSetState($Input2, $GUI_DISABLE)
GUICtrlSetState($Label22, $GUI_DISABLE)
GUICtrlSetState($Combo12, $GUI_DISABLE)
GUICtrlSetState($Label32, $GUI_DISABLE)
GUICtrlSetState($Input22, $GUI_DISABLE)
GUISetState(@SW_SHOW)
#endregion ### END Koda GUI section ###

_Crypt_Startup()
_GDIPlus_Startup()
```

III – Logiciel de traitement des fichiers

```
While 1
    $nMsg = GUIGetMsg()
    Switch $nMsg
        Case $GUI_EVENT_CLOSE
            _Crypt_Shutdown()
            _GDIPlus_Shutdown()
            Exit
        Case $Button1
            selection_fichier_clair($Input1)
        Case $Button12
            selection_fichier_enc($Input12)
        Case $Button2
            encoder()
        Case $Button22
            decoder()
        Case $Checkbox1
            If GUICtrlRead($Checkbox1) = 1 Then
                $encod_crypt = 1
                GUICtrlSetState($Label2, $GUI_ENABLE)
                GUICtrlSetState($Combo1, $GUI_ENABLE)
                GUICtrlSetState($Label3, $GUI_ENABLE)
                GUICtrlSetState($Input2, $GUI_ENABLE)
            Else
                $encod_crypt = 0
                GUICtrlSetState($Label2, $GUI_DISABLE)
                GUICtrlSetState($Combo1, $GUI_DISABLE)
                GUICtrlSetState($Label3, $GUI_DISABLE)
                GUICtrlSetState($Input2, $GUI_DISABLE)
            EndIf
        Case $Checkbox12
            If GUICtrlRead($Checkbox12) = 1 Then
                $decod_crypt = 1
                GUICtrlSetState($Label22, $GUI_ENABLE)
                GUICtrlSetState($Combo12, $GUI_ENABLE)
                GUICtrlSetState($Label32, $GUI_ENABLE)
                GUICtrlSetState($Input22, $GUI_ENABLE)
            Else
                $decod_crypt = 0
                GUICtrlSetState($Label22, $GUI_DISABLE)
                GUICtrlSetState($Combo12, $GUI_DISABLE)
                GUICtrlSetState($Label32, $GUI_DISABLE)
                GUICtrlSetState($Input22, $GUI_DISABLE)
            EndIf

    EndSwitch
WEnd
```

```
Func selection_fichier_clair($input_fichier)
    GUICtrlSetData($Progress1, 0)
    GUICtrlSetData($Progress12, 0)
    $fichier = FileOpenDialog("Sélectionner un fichier...", @WorkingDir & "\", "Fichiers tex
    GUICtrlSetData($input_fichier, $fichier)
    Local $array = StringSplit($fichier, "\")
    $nom_fichier = _ArrayPop($array)
EndFunc   ;==>selection_fichier_clair

Func selection_fichier_enc($input_fichier)
    GUICtrlSetData($Progress1, 0)
    GUICtrlSetData($Progress12, 0)
    $fichier = FileOpenDialog("Sélectionner un fichier...", @WorkingDir & "\", "Fichiers tex
    GUICtrlSetData($input_fichier, $fichier)
    Local $array = StringSplit($fichier, "\")
    $nom_fichier = _ArrayPop($array)
EndFunc   ;==>selection_fichier_enc
```

III – Logiciel de traitement des fichiers

```
Func encoder()
    GUICtrlSetState($Label2, $GUI_DISABLE)
    GUICtrlSetState($Combo1, $GUI_DISABLE)
    GUICtrlSetState($Label3, $GUI_DISABLE)
    GUICtrlSetState($Input2, $GUI_DISABLE)
    If $encod_crypt = 1 Then
        Switch GUICtrlRead($Combo1)
            Case "AES (128bit)"
                $bAlgorithm = $CALG_AES_128
            Case "AES (192bit)"
                $bAlgorithm = $CALG_AES_192
            Case "AES (256bit)"
                $bAlgorithm = $CALG_AES_256
            Case "RC4"
                $bAlgorithm = $CALG_RC4
        EndSwitch
    EndIf
    $contenu_fichier_sortie_enc = FileOpen($fichier_sortie_enc, 2 + 8)
    Ascii2Bin($nom_fichier)
    FileWrite($contenu_fichier_sortie_enc, $TabR2 & @CRLF)
    If StringRight($nom_fichier, 3) = "bmp" Then
        $bitmap = _GDIPlus_ImageLoadFromFile($nom_fichier)
        $str = _Bitmap2BinaryString($bitmap, 100)
;~      Progression à ajouter + cryptage
        Ascii2Bin($str)
        FileWrite($contenu_fichier_sortie_enc, $TabR2)
        _GDIPlus_BitmapDispose($bitmap)
    Else
        $contenu_fichier_clair = FileOpen($fichier, 0)
        $nb_lignes = _FileCountLines($fichier)
        For $j = 1 To $nb_lignes
            GUICtrlSetData($Progress1, ($j / $nb_lignes) * 100)
            $ligne_fichier_clair = FileReadLine($contenu_fichier_clair, $j)
            If $encod_crypt = 1 Then
                Ascii2Bin(_Crypt_EncryptData($ligne_fichier_clair, GUICtrlRead($Input2), $bAlgori
            Else
                Ascii2Bin($ligne_fichier_clair)
            EndIf
            FileWrite($contenu_fichier_sortie_enc, $TabR2 & @CRLF)
        Next
        If $encod_crypt = 1 Then
            GUICtrlSetState($Label2, $GUI_ENABLE)
            GUICtrlSetState($Combo1, $GUI_ENABLE)
            GUICtrlSetState($Label3, $GUI_ENABLE)
            GUICtrlSetState($Input2, $GUI_ENABLE)
        EndIf
    EndIf
    GUICtrlSetData($Progress1,25)
    Sleep(2000)
    MsgBox(0, "", "fini !")
EndFunc   ;==>encoder
```

```
Func decoder()
    GUICtrlSetState($Label22, $GUI_DISABLE)
    GUICtrlSetState($Combo12, $GUI_DISABLE)
    GUICtrlSetState($Label32, $GUI_DISABLE)
    GUICtrlSetState($Input22, $GUI_DISABLE)
    If $decod_crypt = 1 Then
        Switch GUICtrlRead($Combo12)
            Case "AES (128bit)"
                $bAlgorithm = $CALG_AES_128
            Case "AES (192bit)"
                $bAlgorithm = $CALG_AES_192
            Case "AES (256bit)"
                $bAlgorithm = $CALG_AES_256
            Case "RC4"
                $bAlgorithm = $CALG_RC4
        EndSwitch
    EndIf
```

III – Logiciel de traitement des fichiers

```
    $contenu_fichier_encode = FileOpen($fichier, 0)
    Bin2Ascii(FileReadLine($contenu_fichier_encode, 1))
    $nom_fichier_sortie = $ResultatFinal
    $fichier_sortie_clair = @WorkingDir & "\" & $nom_fichier_sortie
    If StringRight($nom_fichier_sortie, 3) = "bmp" Then
        $nb_lignes = _FileCountLines($fichier)
;~      Progression + cryptage à ajouter
        For $j = 2 To $nb_lignes
            $ligne_fichier_encode = FileReadLine($contenu_fichier_encode, $j)
            Bin2Ascii($ligne_fichier_encode)
            $bitmap_texte = $bitmap_texte & $ResultatFinal
        Next
        $bitmap = _BinaryString2Bitmap($bitmap_texte)
        _GDIPlus_ImageSaveToFile($bitmap, @WorkingDir & "\sortie.bmp")
        FileMove(@WorkingDir & "\sortie.bmp", $fichier_sortie_clair)
    Else
        $contenu_fichier_sortie_clair = FileOpen($fichier_sortie_clair, 2 + 8)
        $nb_lignes = _FileCountLines($fichier)
        For $j = 2 To $nb_lignes
            GUICtrlSetData($Progress12, ($j / $nb_lignes) * 100)
            $ligne_fichier_encode = FileReadLine($contenu_fichier_encode, $j)
            Bin2Ascii($ligne_fichier_encode)
            If $decod_crypt = 1 Then
                FileWrite($contenu_fichier_sortie_clair, BinaryToString(_Crypt_DecryptData($Resul
            Else
                FileWrite($contenu_fichier_sortie_clair, $ResultatFinal & @CRLF)
            EndIf
        Next
        _FileWriteToLine($fichier_sortie_clair, $nb_lignes, "", 1)
        If $decod_crypt = 1 Then
            GUICtrlSetState($Label22, $GUI_ENABLE)
            GUICtrlSetState($Combo12, $GUI_ENABLE)
            GUICtrlSetState($Label32, $GUI_ENABLE)
            GUICtrlSetState($Input22, $GUI_ENABLE)
        EndIf
    EndIf
    GUICtrlSetData($Progress12,50)
    Sleep(1000)
    MsgBox(0, "", "fini !")
EndFunc   ;==>decoder
```

```
Func Ascii2Bin($chaine_ascii2binaire)
    $Nbr = StringLen($chaine_ascii2binaire)
    If ($Nbr <= 0) Then
        MsgBox(48, "Erreur", "La ligne est vide")
    Else
        $Flag1 = StringSplit($chaine_ascii2binaire, "")
        $Flag2 = _ArrayToString($Flag1, " ")
        $Flag3 = StringSplit($Flag2, " ")
        Dim $Tab1[$Flag3[1]]
        Dim $Tab2[$Flag3[1]]
        For $i = 0 To $Flag3[1] - 1 Step 1
            $Tab1[$i] = $Flag1[$i + 1]
            If ($Flag1[$i + 1] = "") Then
                $Tab1[$i] = " "
            EndIf
            $Tab2[$i] = Dec2Bin(AscW($Tab1[$i]))
        Next
        $TabR1 = _ArrayToString($Tab1, "")
        $TabR2 = _ArrayToString($Tab2, " ")
    EndIf
EndFunc   ;==>Ascii2Bin
```

III – Logiciel de traitement des fichiers

```
Func Bin2Ascii($chaine_bin2ascii)
    $Nbr = StringLen($chaine_bin2ascii)
    If ($Nbr <= 0) Then
        MsgBox(48, "Erreur", "Le champs est vide")
    Else
        Global $Bool = True
        $Flag1 = StringSplit($chaine_bin2ascii, " ")
        $Flag2 = _ArrayToString($Flag1, " ")
        $Flag3 = StringSplit($Flag2, " ")
        Dim $Tab1[$Flag3[1]]
        For $i = 0 To $Flag3[1] - 1 Step 1
            $Tab1[$i] = $Flag1[$i + 1]
            If ($Flag1[$i + 1] = "") Then
                $Tab1[$i] = "100000"
            EndIf
        Next
        Dim $TabF[$Flag3[1]]
        For $a = 0 To $Flag3[1] - 1 Step 1
            If Not StringRegExp($Tab1[$a], '[^01]') Then
                $Flag4 = StringSplit($Tab1[$a], "")
                $Nbr2 = StringLen($Tab1[$a])
                Global $V1 = 1
                Dim $Tab2[$Nbr2]
                $Tab2[0] = "1"
                For $i = 1 To $Nbr2 - 1 Step 1
                    $V1 *= 2
                    $Tab2[$i] = $V1
                Next
                $Rtab2 = _ArrayToString($Tab2)
                $V1 = 1
                Local $V2
                For $i = $Flag4[0] To 1 Step -1
                    $V2 &= $Flag4[$i]
                Next
                $Flag5 = StringSplit($V2, "")
                $V2 = ""
                Global $V3
                Dim $Tab3[$Nbr2]
                For $i = 0 To $Nbr2 - 1 Step 1
                    $Tab3[$i] = $Flag5[$i + 1]
                    If ($Tab3[$i] = "1") Then
                        $V3 += $Tab2[$i]
                    EndIf
                Next
                $ChrW = ChrW($V3)
                $TabF[$a] = $ChrW
                $V3 = ""
            Else
                $Bool = False
            EndIf
        Next
        If ($Bool = True) Then
            $ResultatFinal = _ArrayToString($TabF, "")
        Else
            MsgBox(48, "Erreur Binaire", "Vous devez respecter" & @LF & "les caractères suivant
        EndIf
    EndIf
EndFunc   ;==>Bin2Ascii


Func Dec2Bin($iNumber)
    Local $sRet = ""
    Do
        $sRet = BitAND($iNumber, 1) & $sRet
        $iNumber = BitShift($iNumber, 1)
    Until $iNumber = 0
    Return $sRet
EndFunc   ;==>Dec2Bin
```