# EID003 Network board on PC104 bus

# Practical Works Manual

didalab
GÉNIE ÉLECTRIQUE

DIDALAB
5 Rue du Groupe Manoukian
78990 Elancourt
Tel: 01.30.66.08.88
Fax: 01.30.66.72.20
E-mail : ge@didalab.fr

Edition :31/08/2015

# SUMMARY

# EID003_EX 1. **TELNET MANAGEMENT**

## 1.1 Topic

| | |
|---|---|
| ***Purpose:*** | Be capable to use the given library, and to manage the Telnet reception. |
| ***Specifications :*** | **Subject :**<br>Write a C program which will manage the Telnet port opening, and characters reception with EID003 board. |

### Necessary Equipment :

PC Micro-Computer using Windows® 95 or latter,
68332 Micro-Controller 16/32 bits Mother Board, Ref. : EID 210 001
Web Board: EID00300
Network connection cable and RS232 cable, Ref. : EGD 000 003
AC/AC 8V Power Supply, 1 A, Ref. : EGD000001,

### Necessary Document :

Document : DMS Web: EID00300

### Time : 3 hours

## 1.2 Analysis and solution

The TCP/IP port number for the Telnet application is 23.
We must open the port with order SetPort(23), to carry out the Telnet applications with the EID003 board.

The EID003 board control:

- The opening and closing of the communication.
- Receiving data from the FIFO buffer (reception).

The user should control the inputs data.

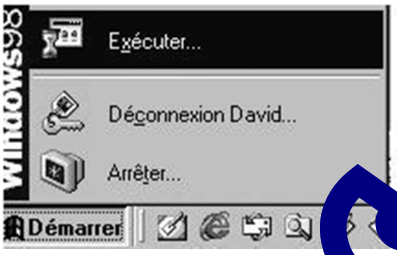
Each received character is displayed as well as the register status.
The letter 'q' is set at the end to close the telnet application.

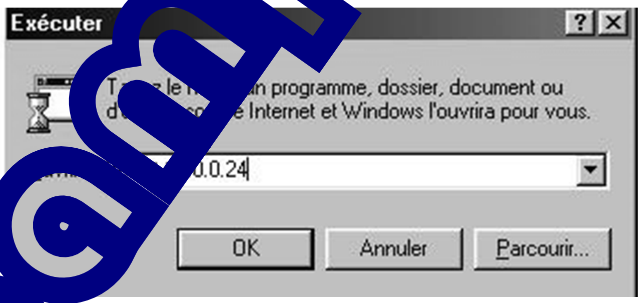
The `EID003_RX_COUNT` register returns the number of characters which are in the receive buffer `EID003_BUFFER`.

The EID003.h file includes the following sub-functions :

`Init_pc104_eid003();` can initialize the access time of the web board
`LitPort();` enable to know the opening [illegible]

Once the program started, we must run the Telnet application window :



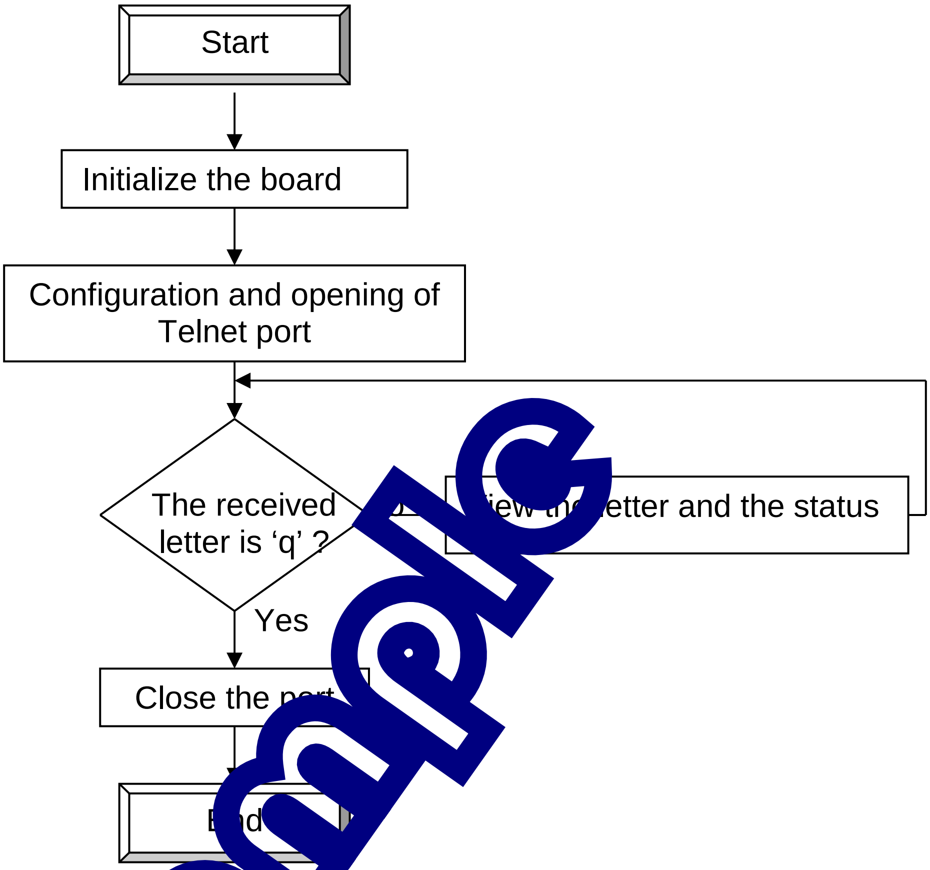

Exécuter

Tapez le nom d'un programme, dossier, document ou d'une ressource Internet et Windows l'ouvrira pour vous.

Ouvrir : 0.0.24

The address is noted 10.0.0.24 because it is the board default address.

## 1.3 Main flowchart


Start
Initialize the board
Configuration and opening of Telnet port
The received letter is 'q' ?
No
View the letter and the status
Yes
Close the port
End

## 1.4 C program

```
/*******************************************************
*     PRACTICAL WORKS BASED ON EID003 BOARD     *
*******************************************************
*     Write a C program              *
*     which realizes the Telnet management     *
*******************************************************
*     File Name: EID003_TP1.c                   *
*     **************                                  *
*******************************************************/

#include "eid210.h"
#include "eid003.h"


main()
{
int i;
short port;
char status;
char travail;
clsscr();
Init_pc104_eid003();
SetPort(23);            // TCP/IP Telnet port
port=LitPort();
printf("Port=%d\n",port);
EID003_CONFIG=0x01;     // open the port
//EID003_CONFIG=0x02;   // empty the buffer
travail=0;
printf("STATUS=%2.2x\n",EID003_STATUS);
while(travail!='q')
      {
            while (EID003_RX_COUNT>0)
                  {
                  travail=EID003_BUFFER;
                  EID003_BUFFER=travail;
            printf("character=%c status=%2.2X\n",travail,EID003_STATUS);
                  }
      }
SetPort(0);       // close the port
EID003_CONFIG=0x01;
port=LitPort();
printf("Port=%d\n",port);
}
```

# EID003_EX 2. CREATION OF A CAT WITH THE TELNET PORT

## 2.1 Topic

| | |
|---|---|
| ***Purpose:*** | Be capable to use the given library, and to completely manage the Telnet port. |
| ***Specifications :*** | **Subject**<br>Write a C program which will realize the Telnet port opening, and characters transmission and reception with EID003 board. |

Necessary Equipment :

PC Micro-Computer using Windows 98 or latter,
68332 Micro-Controller 16/32 bits Mother Board, Ref. : EID 210 001
Web Board: EID00300
Network connection cable and RS232 cable, Ref. : EGD 000 003
AC/AC 8V Power Supply, 1 A, Ref. : EGD000001,

Necessary Document :

Document: DMS Web: EID00300
EID210 board : EID210041

Time : 3 hours

## 2.2 Analysis and solution

### 2.2.1 Practical works context

The TCP/IP port number for the application Telnet is 23.
We must open the port with order SetPort(23), to carry out the applications Telnet with the EID003 board.

The EID003 board control:

- The opening and closing of the communication.
- Receiving and transmitting data from the FIFO buffer (reception and transmission).

The user should control the inputs and outputs data.
An interface can see the received text and the typed text.
The letter '$' terminates the dialogue.

### 2.2.2 Useful registers

`EID003_BUFFER` its function contains receiving a character when reading, and sending it when writing.

`EID003_RX_COUNT` It returns the character number in the reception buffer `EID003_BUFFER`.

`EID003_TX_COUNT` It returns the character number located in the transmission buffer `EID003_BUFFER`.

`EID003_CONFIG` the bit 2 allows sending the buffer.

`EID003_STATUS` the bit 4 is active when sending.

### 2.2.3 The available libraries

To manage the serial link we must configurate it with the help of registers `SCCR0, SCCR1` ; and to detect the press on the keyboard key, the register `SCSR, SCDR` and the mask `RDRF` are useful. Their instruction is explained in the documentation EID210 (ref : EID210041, EX3), and are defined in the file qsm.h .

It's possible to change the address of TCP, and the board network maske :
void `config_tcp(int tcp_adr,int tcp_mask)` the parameters must be given in hexadecimal. For example, in order to set the address 10.0.0.25 and the mask 255.255.255.0, we must give the value 0a000019 and ffffff00.

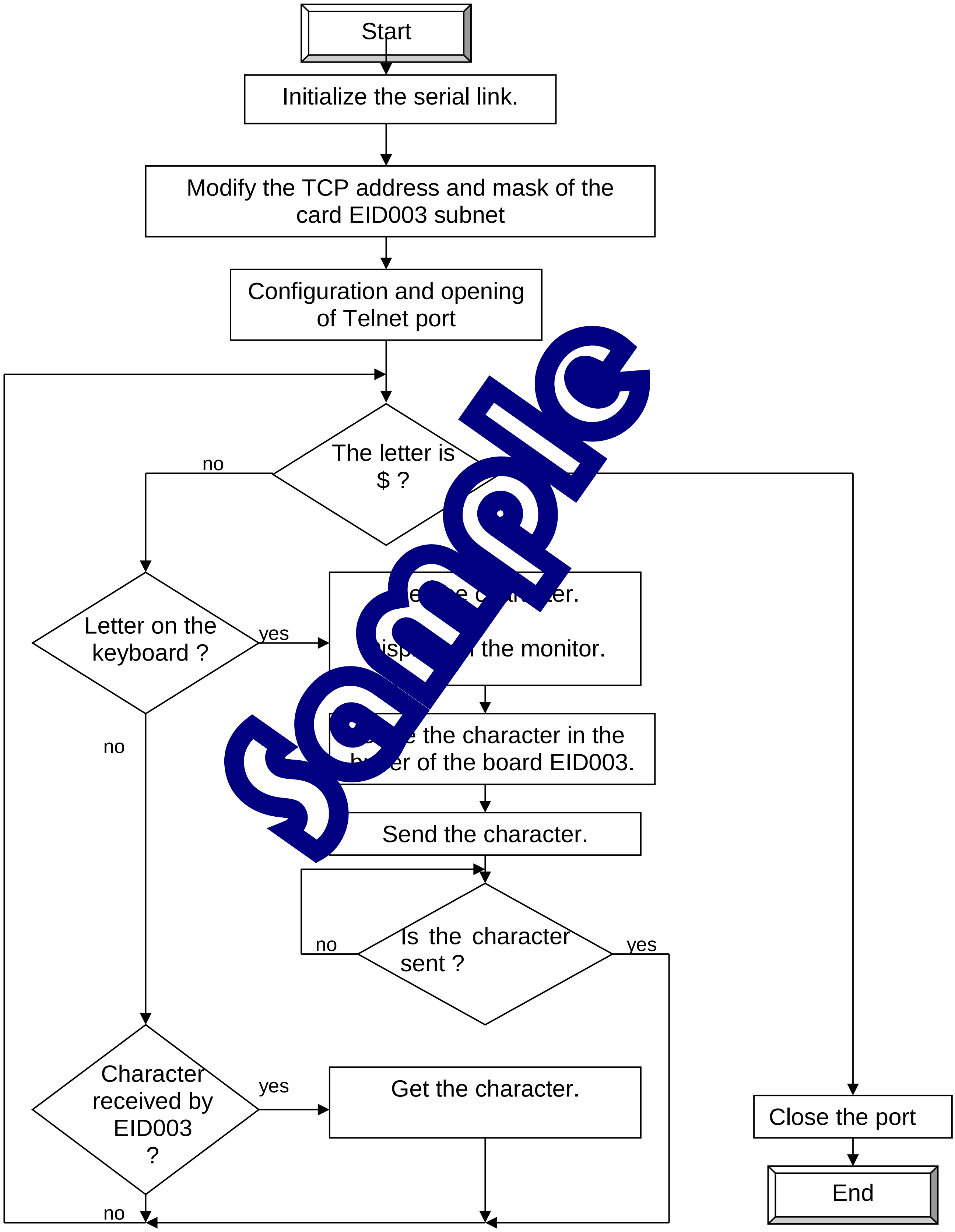
As the same, it is possible to read them by using :

```
int Lit_tcp_adr()
int Lit_tcp_mask()
```

These libraries are stored in the file EID003_TCPIP.c

Remark : If the board address is modified in the Experiment, it will then run the Telnet application with the corresponding address.

## 2.3 Flowchart


Start
Initialize the serial link.
Modify the TCP address and mask of the card EID003 subnet
Configuration and opening of Telnet port
The letter is $ ?
no
Letter on the keyboard ?
yes
Receive the character.
Display it on the monitor.
Store the character in the buffer of the board EID003.
Send the character.
no
Is the character sent ?
no
yes
Character received by EID003 ?
yes
Get the character.
no
Close the port
End

## 2.4 C Program

```
/**************************************************************************/
/*    Program aim:                                                        */
/* Cat creation on the Telnet port                                        */
/**************************************************************************/
#include "eid210.h"
#include "qsm.h"

#include "eid003.h"
#include "eid003_TCPIP.c"

//------------------------------------- Main function
main()
{
      int carac,i;
      char ligrec = 18,colrec = 1,ligemi = 3,colemi = 1;
      clsscr();

      /*Configuration of serial link 57600 Baud for transmission and the
reception*/
      SCCR0 = 9;        // For the transmission [illegible] for 57600 bauds
      SCCR1 = 0x000C;   // To validate the transmission and the reception

      //Initialization of EID210 board access [illegible]
      Init_pc104_eid003();

      //Modification of board address
      printf("The board base address is, by default, 10.0.0.24\n");
      printf("to mofify it on 10.0.0.25 press on Y\n");
      carac=InRs232();
      if(carac=='Y' | carac=='y') config_tcp(0x0a000019,0xFFFFFF00);
            else config_tcp(0x0A000018,0xFFFFFF00);
      for(i=0;i<100000;i++);
      clsscr();
      //Show board address and mask
      carac=Lit_tcp_adr();
      printf("Board address is: %x \n",carac);
      carac=Lit_tcp_mask();
      printf("Board network mask is: %x \n",carac);
      for(i=0;i<100000;i++);

      //Configuration and opening of  TPC/IP TELNET port
      SetPort(23);
      EID003_CONFIG=0x01;

      //Operation Information
      puts("Press a key to continue ...");
      puts("Press $ to exit the program...");
      InRs232();

      //Display interface
      clsscr();
      for(i=0;i<50;i++)
      {
            gotoxy(i,15);
            puts("-");
      }
      gotoxy(0,0);
      puts("Transimission window:");
      puts("--------------------");
      gotoxy(0,16);
      puts("Reception window:");
      puts("----------------------");

//Main loop
while(carac!='$')
      {
      if(SCSR&RDRF)     //if press the key
            {
            carac=SCDR; //store the character
            EID003_BUFFER=carac;    //put it into the buffer
            EID003_CONFIG = 0x03;   //send it
            while((EID003_STATUS&0x04)==0x04); //wait for the transmission
end

            //Display on the monitor
            gotoxy(colemi,ligemi);
            printf("%c\n",carac);
            puts("                                    ");
            colemi++;
            if(colemi==35)    //numbers of characters by line
                  {
                  ligemi++;
                  colemi=1;
                  }
            if(ligemi==14)    //line number
                  {
                  colemi=1;
                  ligemi=3;
                  }
            }

      if(EID003_RX_COUNT!=0)  //if we receive a character
            {
            carac=EID003_BUFFER; //store it

            //Display on the monitor
            gotoxy(colrec,ligrec);
            printf("%c\n",carac);
            puts("                                    ");
            colrec++;
            if(colrec==35)    //numbers of characters by line
                  {
                  ligrec++;
                  colrec=1;
                  }
            if(ligrec==32)    //line number
                  {
                  colrec=1;
                  ligrec=18;
                  }
            }
      }
SetPort(0);       // close the port
EID003_CONFIG=0x01;
}
```

# EID003_EX 3. INTERNET EXPLORER FRAMES RECEPTION

## 3.1 Topic

| | |
|---|---|
| ***Purpose:*** | Be capable to use the given library, and to receive a frame from the Internet Explorer. |
| ***Specifications :*** | **Subject**<br>Write a C program which realize an Internet Explorer frame receipt, and display by identifying the different parts. |

Necessary Equipment :

PC Micro-Computer using Windows ® 95 or latter,
68332 Micro-Controller 16/32 bit Mother Board, Ref. : EID 210 001
Web Board: EID00300
Network connection cable and RS232 cable, Ref. : EGD 000 003
AC/AC 8V Power Supply, 1 A, Ref. : EGD000001,

Necessary Document :

Document: DMS Web: EID00300

Time : 3 hours

## 3.2 Analysis and solution

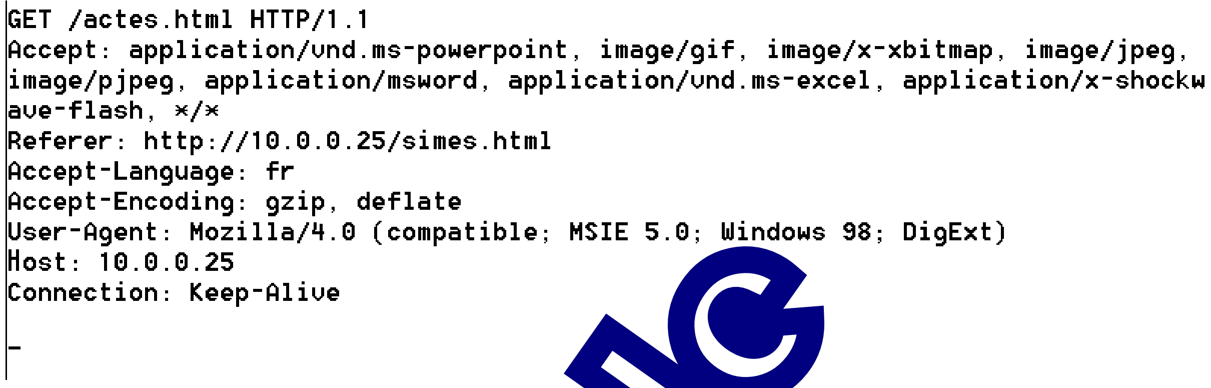
### *3.2.1 Composition of an Internet Explorer frame.*

A request is constituted of the following factors :

method URI version HTTP ↵
information↵↵
body(facultatif voir 3.2.1.1)

(the symbol ↵ represents a return to the line and its ASCii code is /r /n)

Exemple :



```
GET /actes.html HTTP/1.1
Accept: application/vnd.ms-powerpoint, image/gif, image/x-xbitmap, image/jpeg,
image/pjpeg, application/msword, application/vnd.ms-excel, application/x-shockw
ave-flash, */*
Referer: http://10.0.0.25/simes.html
Accept-Language: fr
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 5.0; Windows 98; DigExt)
Host: 10.0.0.25
Connection: Keep-Alive

_
```

#### 3.2.1.1 The method

We are interested in two types of method : the GET and the POST

The GET method is the most common, we can ask for a page (with the modification if necessary). The body is not present in this case.

The POST method is dedicated to changes within a page (ex Know the position of a button, the validation of a checkbox,…). In this case in the information we find Content-Length: XX

The XX represents the number of characters in the body

In the example the method above is GET, the Content-Length does not appear so the body does not exist.

#### 3.2.1.2 The URI

It is simply the requested page.

It appears as form / PageName.html

We will see later that it can also be a picture (bmp,  jpg, …) or a music (mp3, wav,…).

#### 3.2.1.3 The http version

The Web board is designed for the HTTP 1.1 version.

It appears as form HTTPX.X

It’s also possible to treat the 1.0 and 0.9 version.

Attention : in these cases the response protocol is not the same.

Remark : each of these three elements is separated by a space and the line ends with one return “enter”

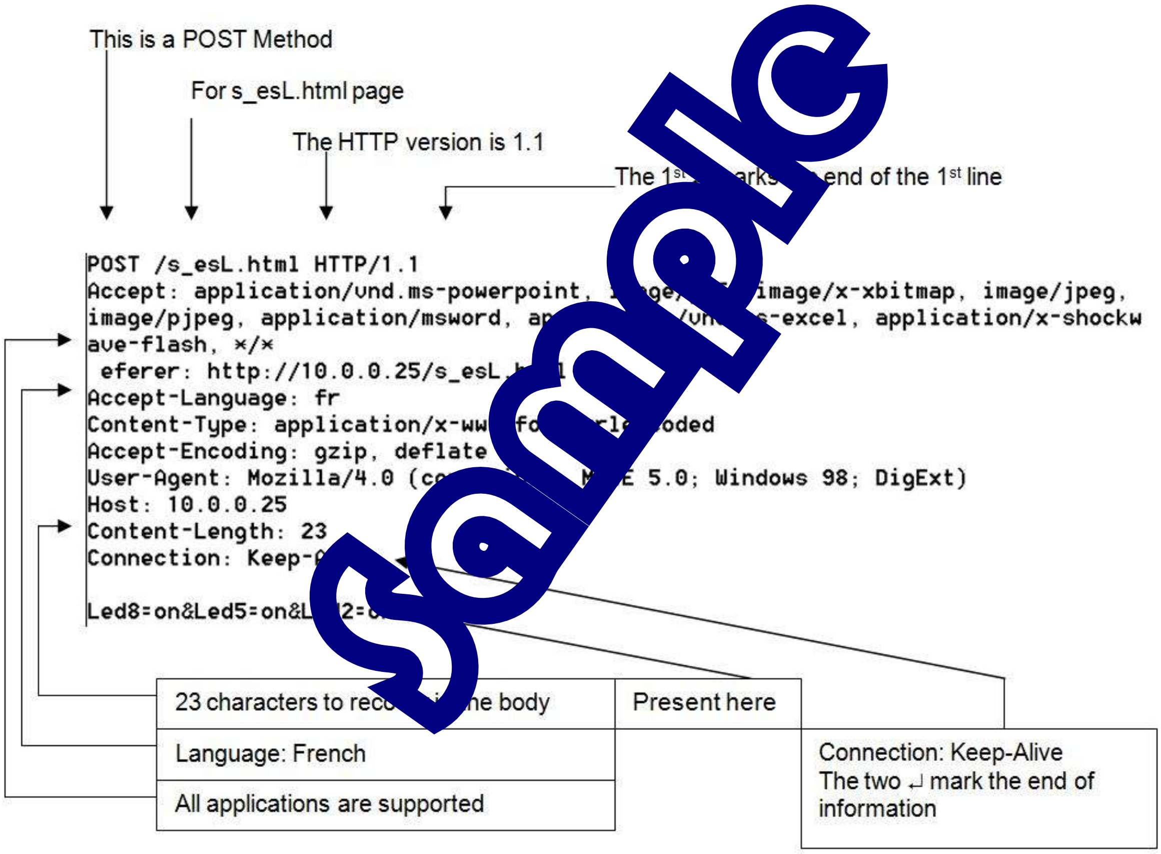
#### 3.2.1.4 The information

In this section, we can determine which applications the client can handle.
In our examples, the « Accept » line application, we can say that the client can handle gif images, bitmap, jpeg and also the type of word, excel, and flash.
The ultimate symbol * / * means that the client can handle all applications.
The « Referer » line gives us the previous address typed by the client.
« Accept-language »: it is the default language.
« Connection »: Keep-Alive and the two ↵ indicate the end of information.

#### 3.2.1.5 The Body

It appears if the Content-Length value is different to 0 in the information.
In general it is present during the POST method.

#### 3.2.1.6 Commented Example



This is a POST Method
For s_esL.html page
The HTTP version is 1.1
The 1st arks end of the 1st line
23 characters to rec he body
Present here
Language: French
All applications are supported
Connection: Keep-Alive
The two ↵ mark the end of information

```
POST /s_esL.html HTTP/1.1
Accept: application/vnd.ms-powerpoint, i  ge/      image/x-xbitmap, image/jpeg,
image/pjpeg, application/msword, ap      /vn   s-excel, application/x-shockw
ave-flash, */*
 eferer: http://10.0.0.25/s_esL.h  l
Accept-Language: fr
Content-Type: application/x-ww  fo   rl  oded
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (co        M  E 5.0; Windows 98; DigExt)
Host: 10.0.0.25
Content-Length: 23
Connection: Keep-A

Led8=on&Led5=on&L  2=o
```

### *3.2.2 Practical works context*

The number of the TCP / IP port for Internet Explorer application is 80.
You must open the port with SetPort (80) to realize the applications with Internet Explorer board EID003.

The program must be able to execute GET and POST methods, as well as HTTP1.1 version.

Language and supported application will be displayed.
The first line will be identified argument by argument and displayed.
The requests must be fully displayed.
For this, a structure is defined in the file eid003_HTML.c (we must be included at the beginning of the program).

```
struct
{
char complet[1000];
char methode[10];
char uri[150];
char version[10];
     struct
     {
     char ToutType;
     char gif;
     char bitmap;
     char jpeg;
     char pjpeg;
     char word;
     char excel;
     }accept;
char body[150];
char tmp[100];
int cpt;
}Buffer_reception;
```

Buffer_reception is composed of all members above.

Par example, Buffer_reception.complet is a buffer who occupies the 1000 characters Buffer_reception, while Buffer_reception.methode occupies the following 10 characters …
Thus, when a request is received, each received character will be put in Buffer_reception.complet [X]. Then it muist extract all elements of Buffer_reception.complet and classify them into Buffer_reception.methode, Buffer_reception.uri, Buffer_reception.version, Buffer_reception.body (in the case of POST).

Buffer_reception.tmp is a buffer of 100 characters left free to the user.

Buffer_reception.cpt is a variable that allows the user to classify each character in the right place of different buffers (see the X above).

Buffer_reception.accept is constituted by a structure.
It can know the accepted applications by client. To access to one of them, we must handle Buffer_reception.accept.gif.

The necessary libraries are :

`Init();` Clear the variables in Buffer_reception and Buffer_emission.

`VideBufferRX();`Empty the buffer reception as long as `EID003_RX_COUNT` >0 and put them in Buffer_reception.complet[ Buffer_reception.cpt].
Attention : we must be sure that Buffer_reception.cpt is correctly initialized! ! !

`IndentifieLigneRequete();` Organize Buffer_reception.methode, Buffer_reception.uri, Buffer_reception.version for Buffer_reception.complet

`IndentifieLigneAccept();`identify the supported applications by the client, and displays on the monitor.

`IndentifieLigneLanguage();`identify the supported language by the client, and displays on the monitor.

`IndentifieLigneBody();`identify the Content-Length xx and put the body context in Buffer_reception.body .

`AfficheBuffer(char *buff,int y);`display the buffer(1st parameter) on the line (2nd parameter).

`int Search_String(char *ptr1,char *ptr2);` Find in the 1st buffer the 2nd buffer. The function return 0 if the 2nd buffer isn't in the 1st buffer. If the 2nd buffer is in the 1st buffer, the function returns the end position; ex :

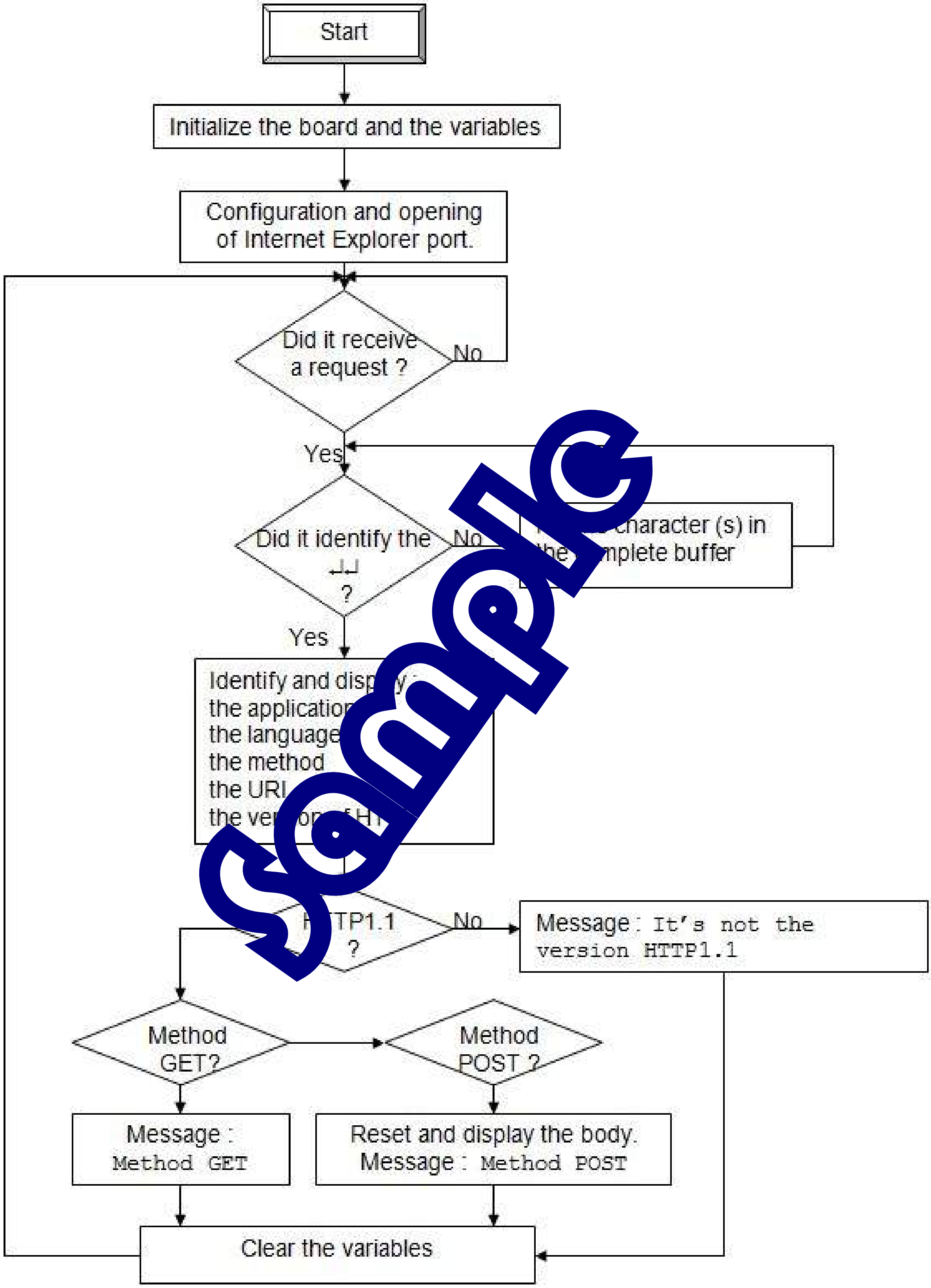
```
    result=Search_String("azerty","er");
```

In this case the result=5.
This function helps us to recover the value of Content-Length and change the buffer.

We must integrate the stdio.h and string.h libraries for applying the string handling function (ex :strstr, strcmp,…).

Once the program executed, we must start the Windows Internet Explorer application, with the address http://10.0.0.24/index.html . ( 10.0.0.24 is the default address)

## 3.3 Flowchart



## 3.4 C Program

```
/***************************************************************************
/
/*    Creation of a HTLM server                                            */
/*    Program aim:                                                         */
/*          Realization of a server HTTP1.1                                */
/*          Analysis of the frames from Internet Explorer (IE)             */
/*                                                                         */
/***************************************************************************
/

#include <string.h>
#include <stdio.h>

#include "eid210.h"
#include "eid003.h"

#include "eid003_HTML.c"

/*************************************/
/*    Program          */
/*************************************/
main()
{
//int code_erreur;
clsscr();

//Initialization of EID210 board address
Init_pc104_eid003();

//Configuration and opening of TCP/IP TELNET port
SetPort(80);
EID003_CONFIG=0x01;

//Initialization of the variables
Init();

//main function
clsscr();
while(CRTL==1)
      {
      while(EID003_RX_COUNT==0);    //If the buffer is empty, wait
      clsscr();

      while(0==strstr(Buffer_reception.complet,standar[1])) //Receive all the
buffer
      VideBufferRX();                                //until \r\n\r\n

      code_erreur=0;
      //Check the applications and the language used by the client
      IndentifieLigneAccept();          //Supported formats
      IndentifieLigneLanguage();        //Used language
      InRs232();                        //Wait for a key-press
      clsscr();
      //Identify all the elements from IE
      IndentifieLigneRequete();         //Identify the line of Method request,
URI_PAGE, VERSION_HTTP
      //Display them
      AfficheBuffer(Buffer_reception.methode,2);
      AfficheBuffer(Buffer_reception.uri,4);
      AfficheBuffer(Buffer_reception.version,6);
      AfficheBuffer(Buffer_reception.complet,10);

      if(0!=strstr(Buffer_reception.version,version[0]))   //Identify the
HTTP1.1 version
            {
            code_erreur=code_erreur|0x0002; //code HTTP1.1
            printf("Version HTTP 1.1\n"); //HTTP1.1
            if(0!=strstr(Buffer_reception.methode,methode[0])) // GET Method
                  {
                  code_erreur=code_erreur|0x0020; //code GET
                  printf("Methode GET\n"); //code GET
                  }
            if(0!=strstr(Buffer_reception.methode,methode[1])) // POST Method
                  {
                  code_erreur=code_erreur|0x0040; //code POST
                  printf("Methode POST\n");
                  if(0!=strstr(Buffer_reception.complet,"Content-Length:
"))//If the Content-Length appears
                        {
                        while(EID003_RX_COUNT!=0) [illegible]BufferRX();      // we
must recover the body from IE
                        IndentifieLigneBody(Buffer_reception.body);
      //Identify the body and put it in its buffer
                        AfficheBuffer(Buffer_reception.body,8);
                        }
                  }
            }
            else  //It's not the HTTP1.1 version. Inform the user
            {     //and do code_erreur=code_erreur|0x0004ou8.
                  //If not, we can handle other version here
            printf("It's not the version HTTP1.1\n");
            }

      //Analyse_code_erreur(); //Analyze the errors
      //Clears variables of reception, identification and response
      Init();
      }
}
```

# EID003_EX 4. RECEPTION AND RESPONSE ON INTERNET EXPLORER

## 4.1 Topic

| | |
|---|---|
| ***Purpose:*** | Be capable to use the given library, and to interpret and respond an Internet request. |
| ***Specifications :*** | **Subject**<br>Write a C program which will analyze the request and respond accordingly.<br>We will also handle the errors. |

Necessary Equipment :

PC Micro-Computer using Windows ® 95 or latter,
68332 Micro-Controller 16/32 bits Mother Board, Ref. : EID 210 001
Web Board: EID00300
Network connection cable and RS232 cable, Ref. : EGD 000 003
AC/AC 8V Power Supply, 1 A, Ref. : EGD000001,

Necessary Document :

Document: DMS Web: EID00300

Time : 3 hours

# 4.2 Analysis and solution

## *4.2.1 Standard response of HTTP1.1 protocol*

The frame is presented as the following :
HTTP/1.1
Space
Status-Code space Reason-Phrase
↵↵
Body
↵↵

The goal will be to create a buffer including all the elements.
Remark : For the HTTP 0.9, it can afford to send : Body↵↵

### 4.2.1.1 Constitution of buffer

To do this, there is a structure created in the eid210_HTML.c file:

```
Struct
{
char version[8];
char space;
char reason [50];
char crlf[2];
char doublecrlf[4];
char body[5000];
}Buffer_emission;
```

Its function is similar to Buffer_Reception.
During the initialization by the program Init();), Buffer_emission.version, Buffer_emission.reason, Buffer_emission.body are filled by the character NULL.
Buffer_emission.space is the space ; Buffer_emission.crlf is a return « enter », "↵"; Buffer_emission.doublecrlf is a double return « enter », like "↵↵"
Before sending this buffer, you must complete the version, reason and body.

Another buffer `char Buff_tmp[15000]` is available for the User.

### 4.2.1.2 Buffer_emission.version

We have to put the version in this buffer.
We can apply for the Buffer_reception.version, and make it a copy and paste in Buffer_emission.version
Remark : the protocol HTTP1.1 and HTTP1.0 are identical.

### 4.2.1.3 Buffer_emission.reason

The reason is constituted by the Status-Code, a space, and the Reason-Phrase.

These two elements are defined by the protocol.
What we found most frequently are defined in the eid003_pageHTML.c file.
This time still copy and paste will put it in the Buffer.

| | |
|---|---|
| `// reason mentioned in a problem`<br>`char Reason_200[]="200 OK";`<br>`char Reason_400[]="400 Bad Request";`<br>`char Reason_404[]="404 Not Found";`<br>`char Reason_414[]="414 Request-URI Too Long";`<br>`char Reason_501[]="501 Not Implemented";`<br>`char Reason_505[]="505 HTTP Version Not Supported";` | Request understood.<br>Error at the request.<br>URI not existent.<br>URI too long.<br>Function not processed by the server.<br>Version HTTP different. |

#### 4.2.1.4 Buffer_emission.body

It includes page HTML.
Attention : this time it's obligatory.
In this case where the reason is `"200 OK"`, we put it the HTML.
In the case of error, we will put it in the `ErreurHTML[]` which is constituted of `"<html></html>"`

#### 4.2.1.5 Comparative table buffer

To list the different methods, URI, HTTP versions, and the table buffer are created in this form:

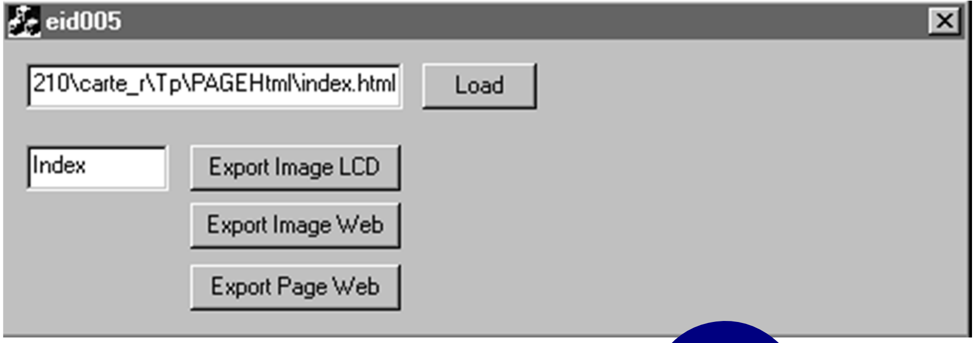
| | | |
|---|---|---|
| `char *methode[]={`<br>`"GET",`<br>`"POST",`<br>`"HEAD",`<br>`"DELAY"`<br>`};` | `char *page[]={`<br>`"/index.html",`<br>`"/Sim[illegible]c.html",`<br>`"/favicon.ico"`<br>`};` | `char *version[]={`<br>`"HTTP/1.1",`<br>`"HTTP/1.0",`<br>`"HTTP/0.9"`<br>`};` |

Remark : defined in this manner, the "strstr" function may be used directly.

### 4.2.2 Page HTML constitution

We must create a HTML page with its editor.
Then we must transform it into a buffer with the provided software.

Press on load to select the HTML page to converter.
In the window below, it show the name of the buffer.
Press on Export Page Web



eid005

210\carte_r\Tp\PAGEHtml\index.html

Index

The page index.html will be transcribed in “.c” code and the buffer will be named Index_html. The “.c” file will be created in the file directory to rewrite.
In the practical works, all the HTML pages are put together in only one file : eid003_pageHTML.c .

### 4.2.3 EID003 circular buffer management

Function:
EID003_BUFFER is a circular buffer.
Each written character in this buffer is stored.
When the eid003 buffer is full ([illegible] characters), it will be sent in the IP2222 buffer.
When the IP2222 buffer is full ([illegible] characters), it is automatically sent through the network.
If the IP2222 buffer is not full, and if there are still characters to send, it will force to send it.

### 4.2.4 Error Handling

A variable error code can check the function condition.
It is defined in the eid003_HTML file, and is composed of the following mode:

```
int code_erreur;  // lets you know where is a potential error
                  //the  1st  byte :version HTTP:     0=> not identified
                  //                                  2=> version 1.1
                  //                                  4=> version 1.0
                  //                                  8=> version 0.9
                  //the 2nd byte : method :     0=> not handled
                  //                                  2=> GET or HEAD
                  //                                  4=> POST
                  //                                  8=> free
                  //the 3ed byte : page         :     0=> page not listed
                  //                                  2=> page listed & identified
                  //                                  [illegible]ge listed & identified
```

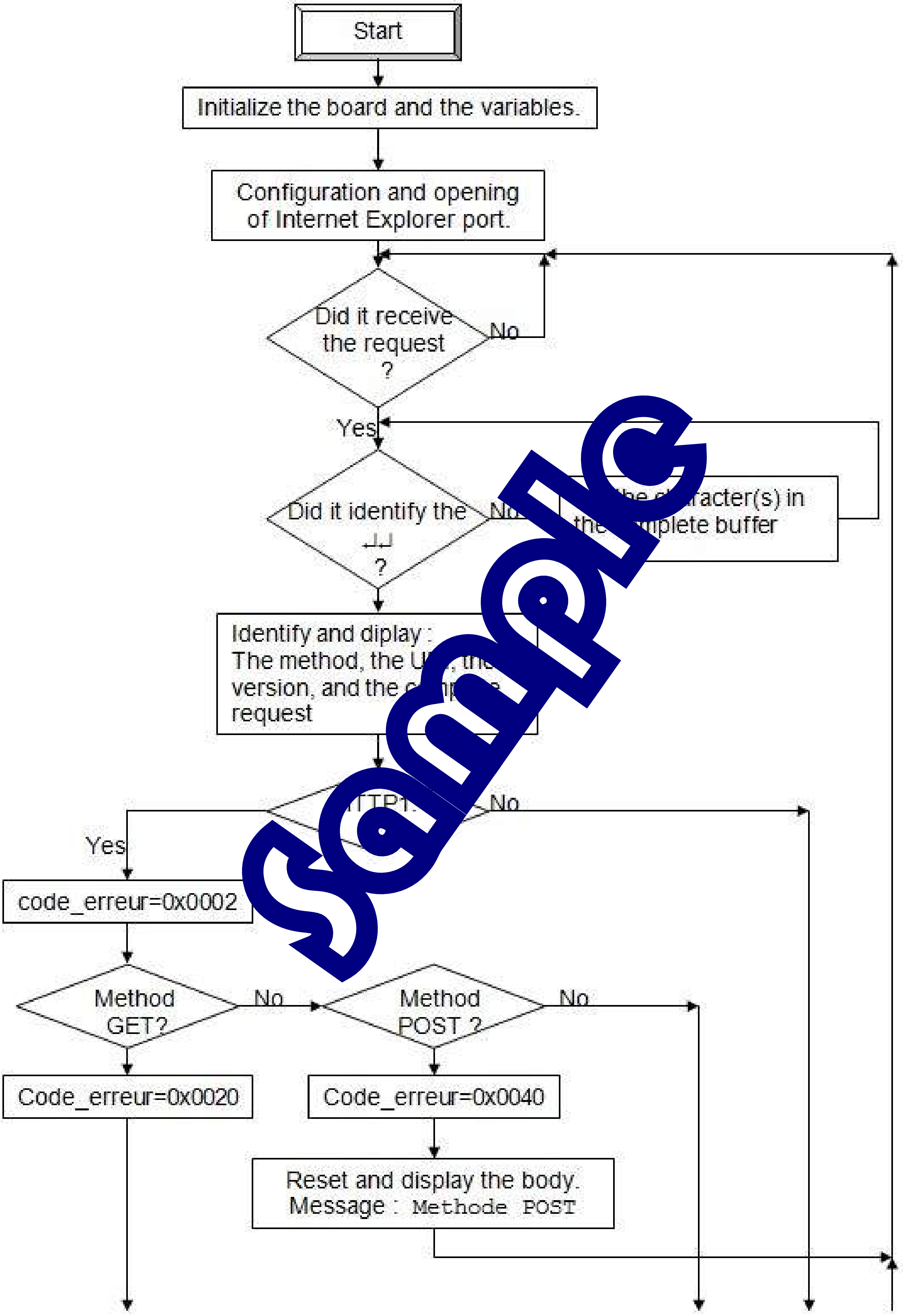
### 4.2.5 Useful libraries

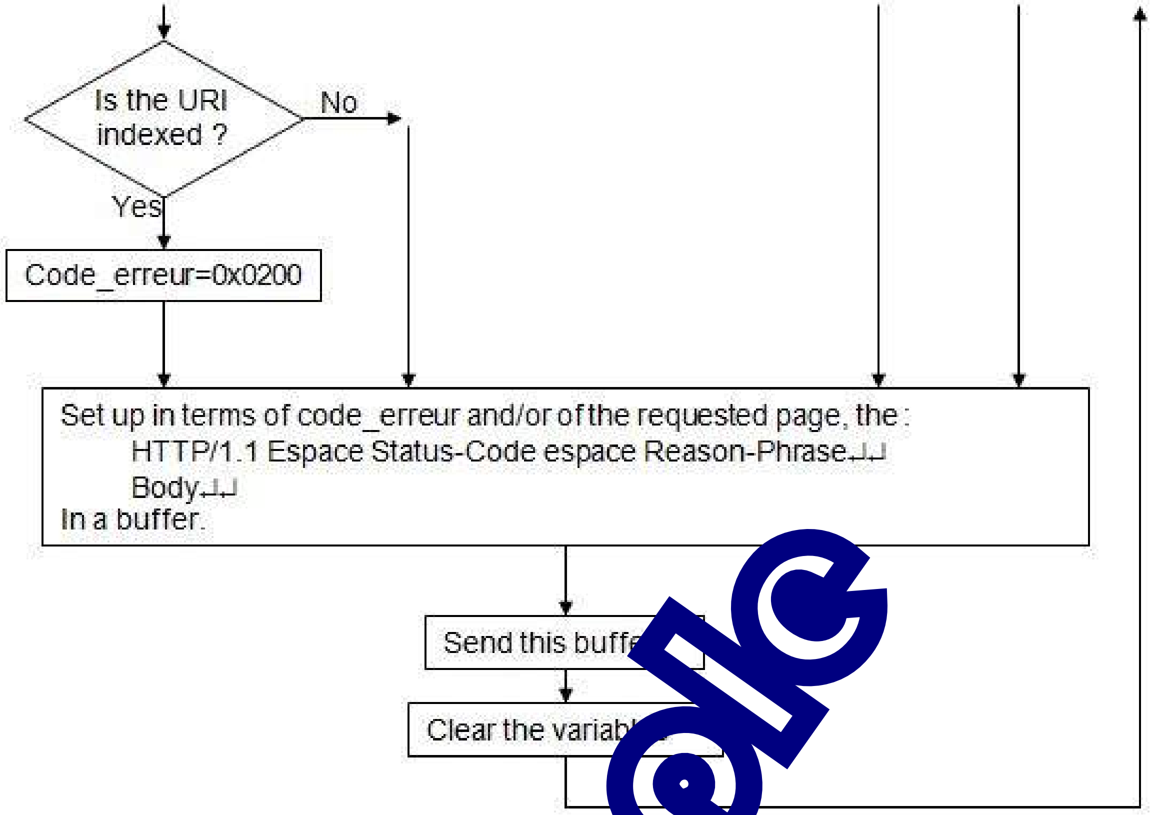
`Reponse_Requete(char Page_HTML[], char Reason[]);` Enable to create the complete buffer with the reason for the right page.

`EnvoiePageHTML(char HTML[]);` Enable to send a buffer.

`Analyse_code_erreur()`: Enable to control the errors : Unrecognized method, different HTTP version, unknown page,….

## 4.3 Flowchart


Start
Initialize the board and the variables.
Configuration and opening of Internet Explorer port.
Did it receive the request ?
No
Yes
Did it identify the ┘┘ ?
No
the character(s) in the complete buffer
Identify and diplay : The method, the U..., the version, and the c... request
HTTP1.
No
Yes
code_erreur=0x0002
Method GET?
No
Method POST ?
No
Code_erreur=0x0020
Code_erreur=0x0040
Reset and display the body. Message : Methode POST


Is the URI indexed ?
No
Yes
Code_erreur=0x0200
Set up in terms of code_erreur and/or of the requested page, the :
HTTP/1.1 Espace Status-Code espace Reason-Phrase↵↵
Body↵↵
In a buffer.
Send this buffe
Clear the variab

## 4.4 C Program

```
/*************************************************************************/
/*    Creation of a HTLM server                                          */
/*    Program aim:                                                       */
/*          Realization of a server HTTP1.1                              */
/*          Analysis of the frames from Internet Explorer (IE)           */
/*          Response according to the standard HTTP1.1                   */
/*************************************************************************/


#include <string.h>
#include <stdio.h>

#include "eid210.h"
#include "eid003.h"

#include "eid003_pageHTML.c"
#include "eid003_HTML.c"



/*************************************/
/*    Main program            */
/*************************************/
main()
{
clsscr();
//Initialization of EID210 board access ti[illegible]
Init_pc104_eid003();

//Configuration and opening of TPC/IP Internet Explorer port
SetPort(80);
EID003_CONFIG=0x01;

//Initialization of the variables
Init();

//main function
clsscr();
while(CRTL==1)
      {
      while(EID003_RX_COUNT==0);    //If the buffer is empty, wait
      clsscr();

      while(0==strstr(Buffer_reception.complet,standar[1])) //Receive all
the buffer
      VideBufferRX();                               //until \r\n\r\n

      code_erreur=0;
      //Identify all the elements from IE
      IndentifieLigneRequete();          //Identify the line of Method
request, URI_PAGE, VERSION_HTTP
      //Display them
      AfficheBuffer(Buffer_reception.methode,2);
      AfficheBuffer(Buffer_reception.uri,4);
      AfficheBuffer(Buffer_reception.version,6);
      AfficheBuffer(Buffer_reception.complet,10);

      if(0!=strstr(Buffer_reception.version,version[0]))   //Identify the
HTTP1.1 version
```

```
            {
            code_erreur=code_erreur|0x0002; //HTTP1.1 code
            if(0!=strstr(Buffer_reception.methode,methode[0])) // GET Method

                  {
                  code_erreur=code_erreur|0x0020; //GET code
                  //Search the page that was requested
                  if(0!=strstr(Buffer_reception.uri,pages[0]))   // Index
Page
                        Reponse_Requete(Index_HTML,Reason_200);
                  if(0!=strstr(Buffer_reception.uri,pages[1]))   // Sim I/O
Page
                        Reponse_Requete(Simes_HTML,Reason_200);
                  //request to add to favorites
                  if(0!=strstr(Buffer_reception.uri,pages[2]))
                        Reponse_Requete(Erreur_HTML,Reason_200);
                  }
            if(0!=strstr(Buffer_reception.methode,methode[1])) //POST Method
                  {
                  code_erreur=code_erreur|0x0040; //POST code
                  printf("POST Method \n");
                  if(0!=strstr(Buffer_reception.contenu,"Content-Length:
"))//If the Content-Length appears
                        {
                        while(EID003_RX_COUNT!=0) VideBufferRX();      //we
must recover the body from IE
                        IndentifieLigneBody(Buffer_reception.body);
      //Identify the body and put them in a buffer
                        AfficheBuffer(Buffer_reception.body,8);
                        }
                  }
            }
            else  //It's not the HTTP 1.1 version . Inform the user
            {     //and do code_erreur=code_erreur|0x0004ou8.
                  //If not, [illegible] other version here
            printf("It's not the HTTP 1.1 version\n");
            }

      Analyse_code_erreur(code_erreur); //Analyze the errors
      //Clears variables of reception, identification and response
      Init();
      }
}
```

# EID003_EX 5. **CGI INTEGRATION**

## 5.1 Topic

| | |
|---|---|
| ***Purpose:*** | Be capable to use the given library, and to integrate images and sounds.<br>Return an interactive page. |
| ***Specifications :*** | **Subject**<br>Be capable to integrate sounds and images in an HTML page.<br>Operate and Monitor the input / output simulator via an HTML page. |

### Necessary Equipment :

PC Micro-Computer using Windows [illegible] or latter,
68332 Micro-Controller 16/32 bits Master Board, Ref. : EID 210 001
Web Board: EID00300
Network connection cable and RS232 cable, Ref. : EGD 000 003
AC/AC 8V Power Supply, 1 A, Ref. : EGD000001,

### Necessary Document :

Document: DMS Web: EID00300
Document Input / output board : EID001001

Time : 4 hours

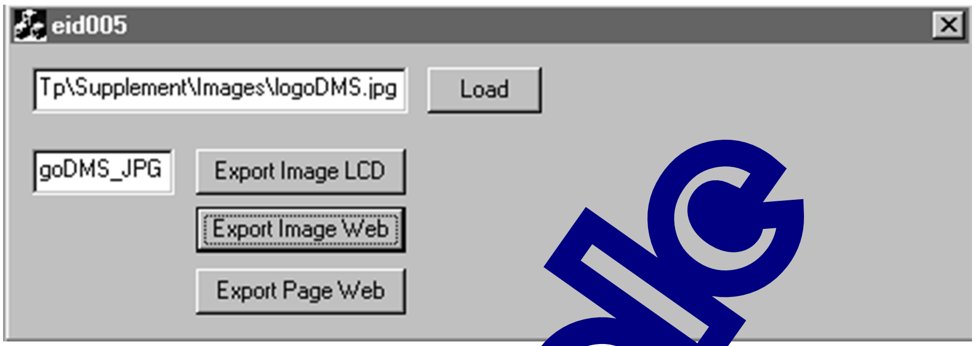
## 5.2 Analysis and solution

### *5.2.1 Integration of image and sound*

#### 5.2.1.1 Images and sounds conversion

The goal is to rewrite the .jpg or .mp3 file into a value table.
The provided software can do this.

Press on load to select the HTML page to converter.
In the window below, it show the name of the buffer.
Press on Export Page Web.



The logosDMS.jpg file is rewritten in logoDMS.c and the table is named as logoDMS_JPG.
Furthermore, in the logoDMS.c file, a variable named logoDMS_JPG_taille gives the number of elements in the table.
The file .C will be created in the directory of the file to re-transcribe.
In the practical works, all the pages HTML are put together in only one file: eid003_pageHTML.c.
It will include image.c and son.c files in which will be found respectively images and sounds.

This is valid for all types of images and sounds.
! ! ! Attention : Because the eid210 memory is not infinite, it is necessary to compress up the files you want to include.
Some privilege infraction may appear. ! ! !

#### 5.2.1.2 Send the images or sounds

The principle is the same as a HTML page.
However, another function has been created in order to pass the value table size as a parameter:

```
Reponse_Image(char Image[],char Reason[],int Taille)
EnvoieImage(char Img[],int Taille)
```

### 5.2.2 *Driving of Inputs / Outputs simulator*

Driving of Inputs / Outputs simulator
For the operation of Inputs / Outputs simulator, we must refer to the Inputs / Outputs board document: EID001001.
For the driving of Inputs / Outputs simulator, a library and some functions were written in eid001.h and eid003.c files.
We must include the object file eid001.o in the linker, and initialize the ports A, B, and C.

### 5.2.3 *Return an interactive page*

When building an HTML page, it is possible to include the buttons, the Checkbox, the Select ... and to ask for a page update.
In this case, when Checkbox is activated, we find in the request nomdelacheckbox = on. If it is not, nothing appears in the request.

The aim will be to find in the request the LEDs and the value that the client wants to display on the 7-segment.
It will also read the value of the switches and potentiometer on eid001 board, update the HTML page code.
Finally we will return the modified page to the client.

For the used method, the handling, even the same operation, will not be in the same place

#### 5.2.3.1 Update a page with the GET method

In this case the request appears like this:

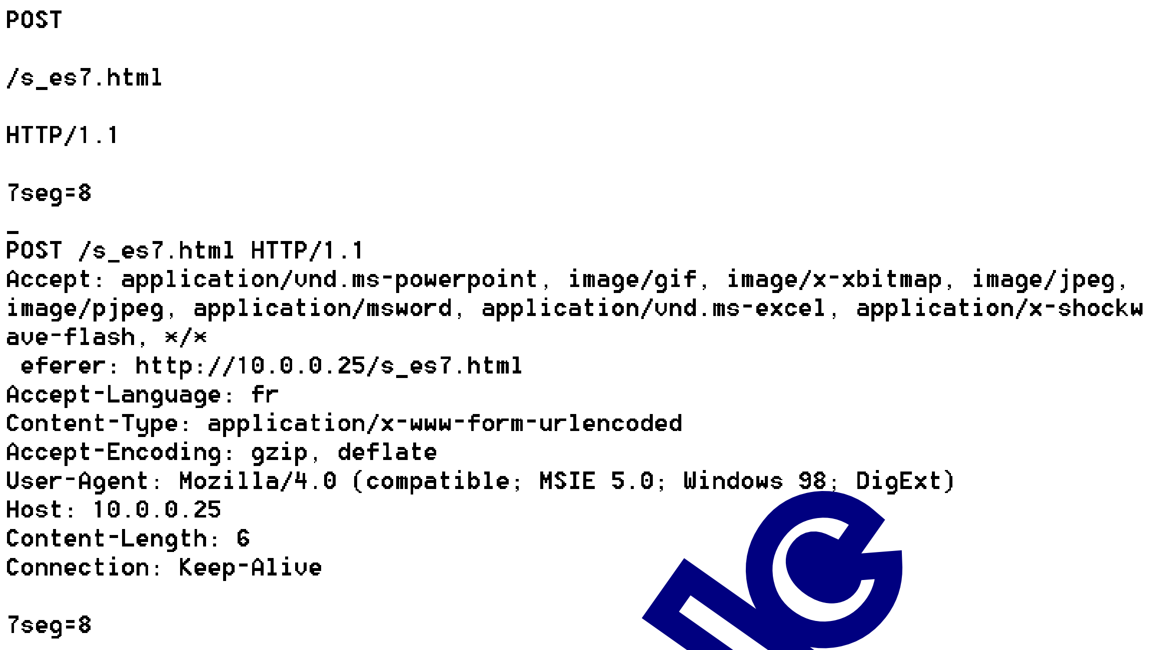
```
GET

/actes.html?Potar=f7e&Switchs=00&Leds=bb&7Seg=7

HTTP/1.1



GET /actes.html?Potar=f7e&Switchs=00&Leds=bb&7Seg=7 HTTP/1.1
Accept: application/vnd.ms-powerpoint, image/gif, image/x-xbitmap, image/jpeg,
image/pjpeg, application/msword, application/vnd.ms-excel, application/x-shockw
 ve-flash, */*
Referer: http://10.0.0.25/actes.html?Potar=f8a&Switchs=00&Leds=aa&7Seg=a
Accept-Language: fr
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 5.0; Windows 98; DigExt)
Host: 10.0.0.25
Connection: Keep-Alive
```

We see that the URI contains the new values of the potentiometer and switches, and the new request for LEDs and the 7 segments.
In the information, the “Referer” line gives us the previous page, or we can find the previous values of the potentiometer and switches, and previous request for LEDs and the 7 segments

During a GET method, the modifications appear in the URI, as the form / nomdelapage.html?modifications.
We must therefore make a distinction between / nomdelapage.html and / nomdelapage.html? to recognize if the code change should be made.

#### 5.2.3.2 Update a page with the POST method

In this case the request appears like this:



```
POST

/s_es7.html

HTTP/1.1

7seg=8

-
POST /s_es7.html HTTP/1.1
Accept: application/vnd.ms-powerpoint, image/gif, image/x-xbitmap, image/jpeg,
image/pjpeg, application/msword, application/vnd.ms-excel, application/x-shockw
ave-flash, */*
 eferer: http://10.0.0.25/s_es7.html
Accept-Language: fr
Content-Type: application/x-www-form-urlencoded
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 5.0; Windows 98; DigExt)
Host: 10.0.0.25
Content-Length: 6
Connection: Keep-Alive

7seg=8
```

This time, the values to modify are presented in the body.
The information (the “Referer” line : ) don't contain the modification.

### 5.3.4 CGI context

When a change is asked, it should recover the information, and handle them. In order to do this, the `Search_String` library is available in the eid003_HTML.c file:

```
int Search_String(char *ptr1,char *ptr2)
{
char *ptr;
int ret=0;
     (int) ptr=strstr(ptr1,ptr2);
          ret =(ptr-ptr1)+strlen(ptr2);
          if(ret<0) ret=0;
     return ret;
}
```

This function finds the second situation in the first situation.
If it is found, the function returns a variable that represents the position of the chain end.

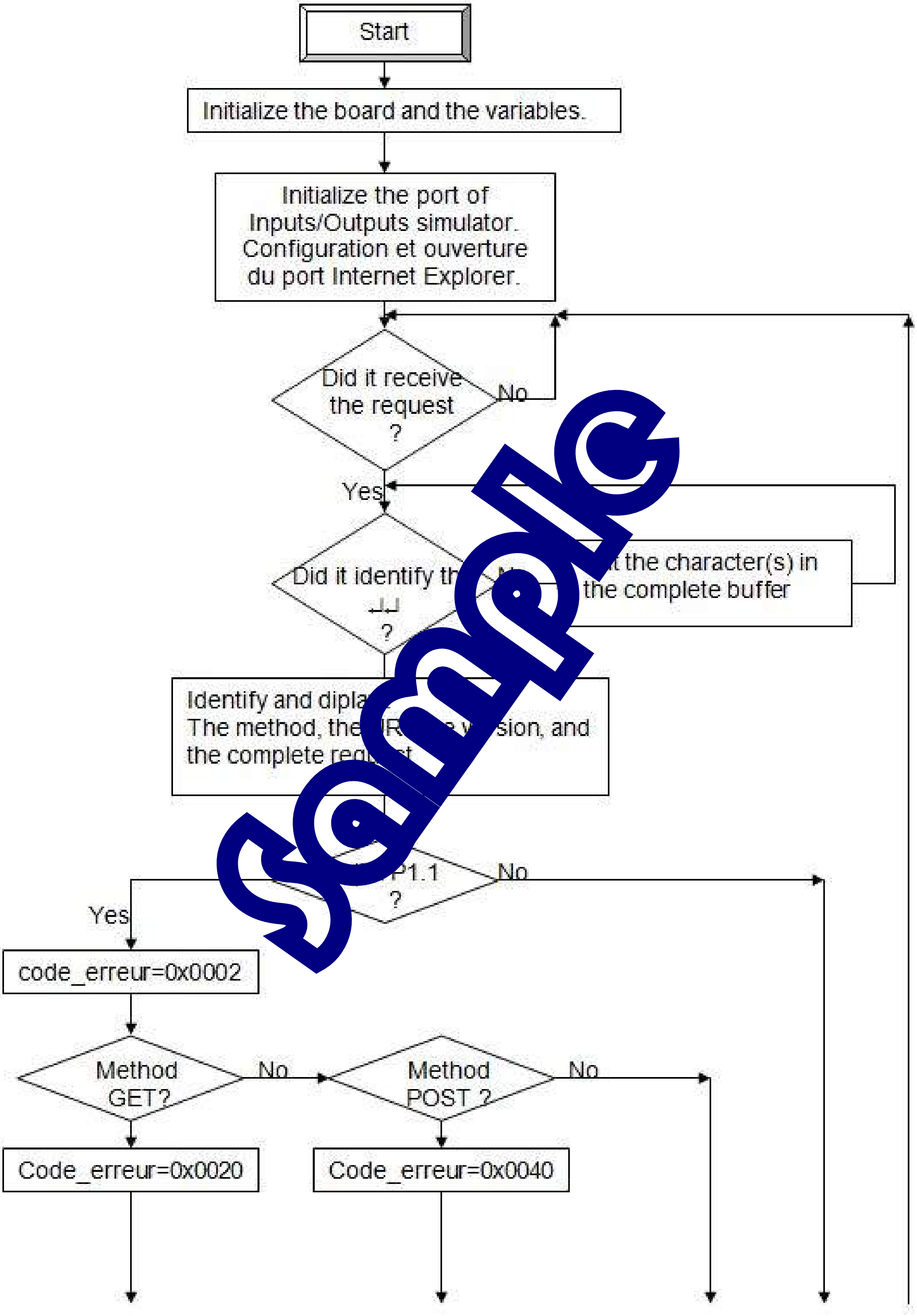
Example about the actes.html page:
If the URI buffer:
**`/actes.html?Potar=67e&Switchs=FF&Leds=55&7Seg=9`**
`place=Search_String(Buffer_reception.complet,"&Leds=");`
The function is going to find &Leds= in the URI buffer.
`&Leds=` is found, then the function returns 39. So place=39.
Then we can do `result=Buffer_reception.complet[place];`
We then store the value 9 in result, and we can displayit by using 7 segments.
If `&Leds=` was not found, then place = NULL, and the handling stops.

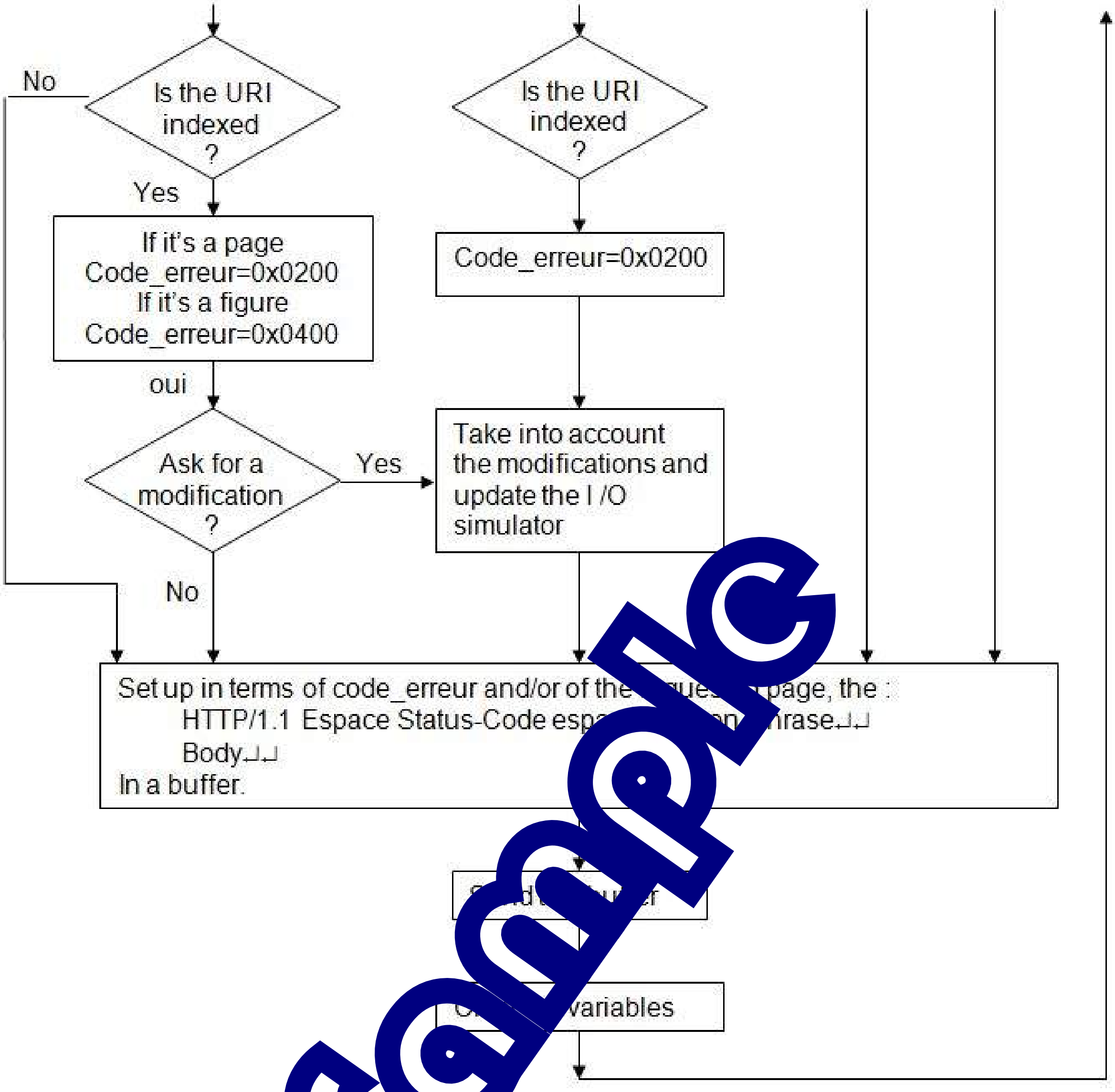
To update the value of switches, We can do this :

```
// Catch the switches value and handle results
result=CaptureSwitch();
j=result&0x0F;
result=(result>>4)&0x0F;
//Modify the page Actes_HTML code
place=Search_String(Actes_HTML,"name=\"Switchs\" value=\"");
//Put the most significant switches value in the code of HTML page
Actes_HTML[place]=tab_ascii[result];
// Put the least significant switches value in the code of HTML page
Actes_HTML[place+1]=tab_ascii[j];
```

All these operation are performed in the eid003_CGI.c file

## 5.3 Flowchart


Start
Initialize the board and the variables.
Initialize the port of Inputs/Outputs simulator. Configuration et ouverture du port Internet Explorer.
Did it receive the request ?
No
Yes
Did it identify th
?
No
the character(s) in the complete buffer
Identify and dipla
The method, the
sion, and the complete req
P1.1 ?
No
Yes
code_erreur=0x0002
Method GET?
No
Method POST ?
No
Code_erreur=0x0020
Code_erreur=0x0040


No
Is the URI indexed ?
Yes
If it's a page Code_erreur=0x0200 If it's a figure Code_erreur=0x0400
oui
Ask for a modification ?
Yes
No
Is the URI indexed ?
Code_erreur=0x0200
Take into account the modifications and update the I /O simulator
Set up in terms of code_erreur and/or of the reques page, the :
HTTP/1.1 Espace Status-Code espa n hrase↵↵
Body↵↵
In a buffer.
S d u r
C variables

# 5.4 C Program

```
/***************************************************************************/
/*    Creation of an HTLM server                                           */
/*    Program aim:                                                         */
/*          Realization of a HTTP1.1 server                                */
/*          Frames analysis comes from Internet Explorer (IE)              */
/*          Response on the HTTP1.1 standard                               */
/*          Return the interactive page                                    */
/***************************************************************************/

#include <string.h>
#include <stdio.h>

#include "eid210.h"
#include "eid003.h"

#include "eid003_pageHTML.c"

#include "eid003_HTML.c"
#include "eid003_CGI.c"
/*************************************/
/*    Main program           */
/*************************************/
main()
{
int tmp,i;
clsscr();

//Initialization of Input/output simulator
InitLED();
InitSwitch();
Init7Seg();

//Initialization of EID210 board [illegible] time
Init_pc104_eid003();

//Configuration and opening of TPC/IP Internet Explorer port
SetPort(80);
EID003_CONFIG=0x01;

//Initialization of the variables
Init();

//main function
clsscr();
while(CRTL==1)
      {
      while(EID003_RX_COUNT==0);   //If the buffer is empty, wait
      clsscr();

      while(0==strstr(Buffer_reception.complet,standar[1])) //Receive all the
buffer
      VideBufferRX();                                 //until \r\n\r\n

      code_erreur=0;
      //Identify all the elements from IE
      IndentifieLigneRequete();          //Identify the line of Method request,
URI_PAGE, VERSION_HTTP
      //Display them
      AfficheBuffer(Buffer_reception.methode,2);
      AfficheBuffer(Buffer_reception.uri,4);
      AfficheBuffer(Buffer_reception.version,6);
      AfficheBuffer(Buffer_reception.complet,10);

      if(0!=strstr(Buffer_reception.version,version[0]))   //Identify the HTTP1.1
version
            {
            code_erreur=code_erreur|0x0002; //HTTP1.1 code
            if(0!=strstr(Buffer_reception.methode,methode[0])) // GET Method
                  {
                        code_erreur=code_erreur|0x0020; // GET code

                        //Search the page that was requested
                        if(0!=strstr(Buffer_reception.uri,pages[0]))   //Index
Page
                              Reponse_Requete(Index_HTML,Reason_200);

                        if(0!=strstr(Buffer_reception.uri,pages[1]))   // Sim I/O
Page
                              Reponse_Requete(Simes_HTML,Reason_200);

                        //During the modification request with the GET method,
                        //the page appears in the form page.html?
                        //then we have to check at first whether we have asked for
the page change
                        if(0!=strstr(Buffer_reception.uri,pages[9]))   //Actes
Page for the modification
                              {
                              CGI_PageActes();                          //Modify
the page code
                              Reponse_Requete(Actes_HTML,Reason_200);   //Reply
                              tmp=strlen(Buffer_reception.uri);         //Clear
the URI variables
                              for(i=0;i<tmp;i++) Buffer_reception.uri[i]='\0';
                              }
                        if(0!=strstr(Buffer_reception.uri,pages[2]))   //Actes
Page
                              Reponse_Requete(Actes_HTML,Reason_200);

                        if(0!=strstr(Buffer_reception.uri,pages[10]))  //S_I/O2
Page for a modification
                              {
                              CGI_PageS_es2();                          //Modify
the page code
                              Reponse_Requete(S_es2_HTML,Reason_200);   //Reply
                              tmp=strlen(Buffer_reception.uri);         //Clear
the URI variables
                              for(i=0;i<tmp;i++) Buffer_reception.uri[i]='\0';
                              }
                        if(0!=strstr(Buffer_reception.uri,pages[3]))   //S_I/O2
Page
                              Reponse_Requete(S_es2_HTML,Reason_200);

                        //Frames, main page
                        if(0!=strstr(Buffer_reception.uri,pages[4]))   //S_I/O3
Page
                              Reponse_Requete(S_es3_HTML,Reason_200);
                        //Frames, main page components
                        if(0!=strstr(Buffer_reception.uri,pages[5]))   //S_I/OS
```

```
Page
                              Reponse_Requete(S_esS_HTML,Reason_200);
                        if(0!=strstr(Buffer_reception.uri,pages[6]))   //S_I/OP
Page
                              Reponse_Requete(S_esP_HTML,Reason_200);
                        if(0!=strstr(Buffer_reception.uri,pages[7]))   //S_I/OL
Page
                              Reponse_Requete(S_esL_HTML,Reason_200);
                        if(0!=strstr(Buffer_reception.uri,pages[8]))   //S_I/O7
Page
                              Reponse_Requete(S_es7_HTML,Reason_200);

                        //The page refreshes every 2 seconds
                        if(0!=strstr(Buffer_reception.uri,pages[11]))  //Auto
actu Page
                              {
                              CGI_PageAutoActu();
                              Reponse_Requete(Auto_Actu_HTML,Reason_200);
                              }
                        //Figures
                        if(0!=strstr(Buffer_reception.uri,images[0]))  //logoDMS
picture

      Reponse_Image(logoDMS_JPG,Reason_200,logoDMS_JPG_taille);
                        if(0!=strstr(Buffer_reception.uri,images[1]))  //Building
figure

      Reponse_Image(dmsbat_JPG,Reason_200,dmsbat_JPG_taille);

                        //Sound
                        if(0!=strstr(Buffer_reception.uri,son[0]))     //sound
when arrived on the page
                              Reponse_Image(demo_MP3,Reason_200,demo_MP3_taille);
//

                        if(0!=strstr(Buffer_reception.uri,pages[12]))  //ask to
add to favorites
                              Reponse_Requete(Erreur_HTML,Reason_200);

                  }
            if(0!=strstr(Buffer_reception.methode,methode[1])) // POST Method
                  {
                  code_erreur=code_erreur|0x0040; // POST code
                  if(0!=strstr(Buffer_reception.complet,"Content-Length: "))//If
the Content-Length appears
                        {
                        while(EID003_RX_COUNT!=0) VideBufferRX();      we must
recover the body from IE

                        IndentifieLigneBody(Buffer_reception.body);    //Identify
the body and put them in its buffer
                        AfficheBuffer(Buffer_reception.body,8);
                        }
                        if(0!=strstr(Buffer_reception.uri,pages[5]))   //S_I/OS
Page
                              {
                              CGI_PageS_esS();
                              Reponse_Requete(S_esS_HTML,Reason_200);
                              }
                        if(0!=strstr(Buffer_reception.uri,pages[6]))   //S_I/OP
Page
```

```
                              {
                              CGI_PageS_esP();
                              Reponse_Requete(S_esP_HTML,Reason_200);
                              }
                        if(0!=strstr(Buffer_reception.uri,pages[7]))   //S_I/OL
Page
                              {
                              CGI_PageS_esL();
                              Reponse_Requete(S_esL_HTML,Reason_200);
                              }
                        if(0!=strstr(Buffer_reception.uri,pages[8]))   //S_I/O7
Page
                              {
                              CGI_PageS_es7();
                              Reponse_Requete(S_es7_HTML,Reason_200);
                              }
                  }
            }
            else  //It's not the HTTP1.1 version . Inform the user
            {     //and do code_erreur=code_erreur|0x00408.
                  //If not, we can handle other version here
            printf("It's not the  HTTP1.1 version\r\n");
            }

      Analyse_code_erreur(code_erreur); //Analyze the errors
      //Clears variables of reception, identification and response
      Init();
      }
}
```