Titre (nom) du projet



(Une image originale représentant le projet)

Pittier Stéphane – Cin4B

ETML

Environ 110H

Chef de projet : Gilbert Gruaz

Expert 1 : Xavier Carrel

Expert 2 : Jean Zahn

Table des matières

[1 Spécifications 4](#_Toc482185325)

[1.1 Titre 4](#_Toc482185326)

[1.2 Description 4](#_Toc482185327)

[1.3 Matériel et logiciels à disposition 4](#_Toc482185328)

[1.4 Prérequis 4](#_Toc482185329)

[1.5 Cahier des charges 4](#_Toc482185330)

[2 Planification Initiale 5](#_Toc482185331)

[3 Analyse 6](#_Toc482185332)

[3.1 Opportunités 6](#_Toc482185333)

[3.2 Document d’analyse et conception 6](#_Toc482185334)

[3.2.1 Analyse concurrentielle 6](#_Toc482185335)

[3.2.2 Structure graphique 7](#_Toc482185336)

[3.2.3 Maquette 9](#_Toc482185337)

[3.2.4 Structogrammes 10](#_Toc482185338)

[3.3 Conception des tests 12](#_Toc482185339)

[3.4 Planification détaillée 12](#_Toc482185340)

[4 Réalisation 12](#_Toc482185341)

[4.1 Dossier de Réalisation 12](#_Toc482185342)

[4.2 Modifications 13](#_Toc482185343)

[5 Tests 13](#_Toc482185344)

[5.1 Dossier des tests 13](#_Toc482185345)

[6 Conclusion 13](#_Toc482185346)

[6.1 Bilan des fonctionnalités demandées 13](#_Toc482185347)

[6.2 Bilan de la planification 13](#_Toc482185348)

[6.3 Bilan personnel 13](#_Toc482185349)

[7 Divers 14](#_Toc482185350)

[7.1 Journal de travail 14](#_Toc482185351)

[7.2 Bibliographie 14](#_Toc482185352)

[7.3 Webographie 14](#_Toc482185353)

[8 Annexes 14](#_Toc482185354)

# Spécifications

## Titre

Réalisation d’une application de contrôle d’exercices en électronique numérique

## Description

Implémenter une application qui va permettre aux utilisateurs (élèves ou enseignants), de contrôler les résultats des exercices de conversions de données numérique en binaire, octal, décimal, hexadécimal, BCD nombres réels à virgule flottante, ainsi que des opérations élémentaires comme des compléments à deux, addition et soustraction.

## Matériel et logiciels à disposition

* 1 ordinateur standard ETML
* Visual Studio 2015
* Suite Office 2016

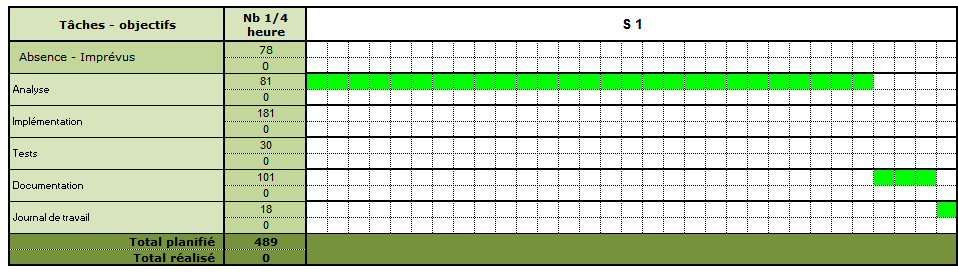
## Prérequis

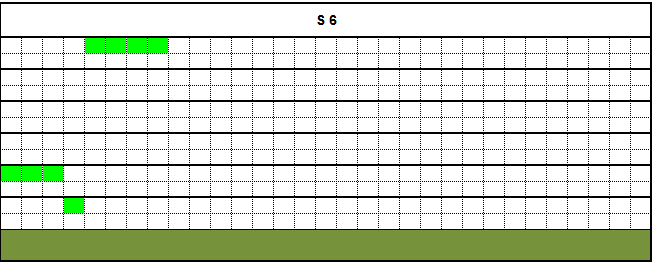
Avoir suivi les modules ICH à l’ETML, les projets et effectué des stages.

## Cahier des charges

Le document fourni par le chef de projet fait foi. Il doit être mis en annexe 🡺 Lien sur CDC

# Planification Initiale





# Analyse

## Opportunités

**A compléter**

## Document d’analyse et conception

Comme écrit dans les opportunités, le programme sera développé en C#. Il utilisera le type Windows Form Project, plus simple d’utilisation pour l’utilisateur que la console. Il sera aussi séparé en 2 fenêtre. Le convertisseur et le calculateur.

### Analyse concurrentielle

Pour cette partie, j’ai trouvé deux sites permettant de faire presque pareil que mon application. Le premier contient plus de types que je veux, mais la conversion en ternaire et en quintal n’est pas une priorité. Pour le deuxième, il ne peut convertir que le décimal, hexadécimal et binaire.

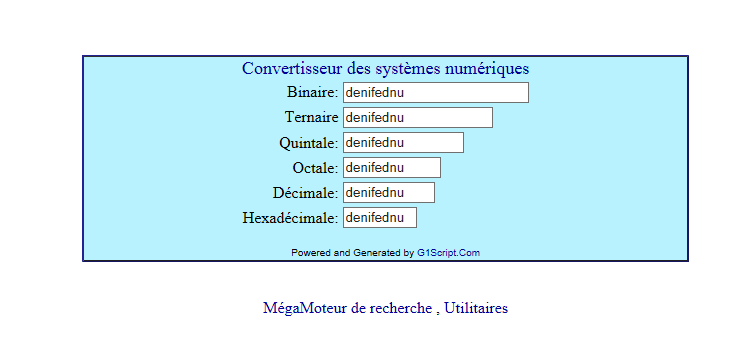
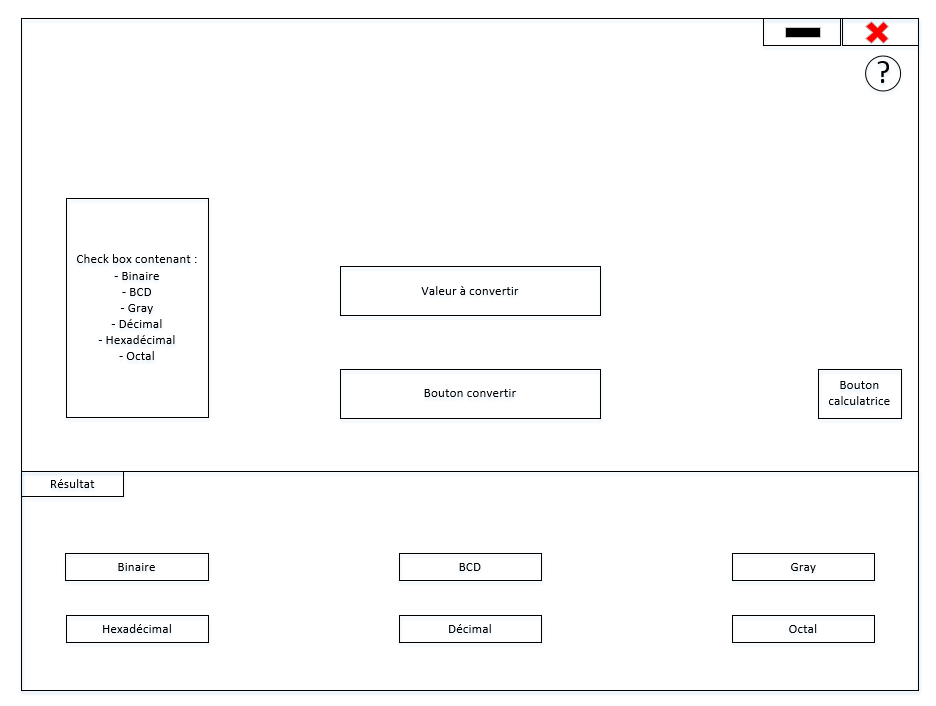
La différence se trouve sur le fait qu’aucun des deux sites ne puissent convertir des valeurs à virgule, et que le premier ne puisse pas convertir une valeur négative.

Figure Page d'accueil du site 2

Figure Exemple de conversion d'une valeur négative sur le site 1

### Structure graphique



6

5

4

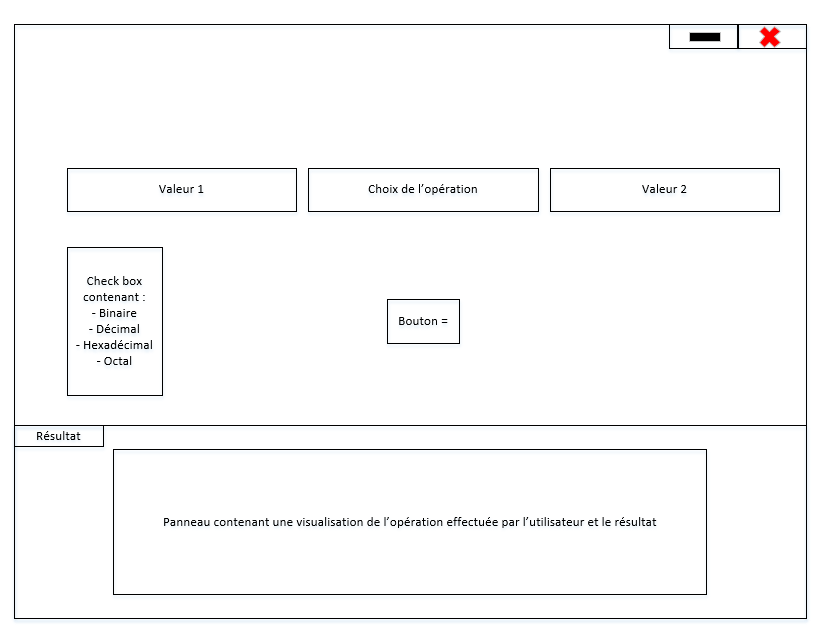
1

2

3

La page principale du programme contient la partie convertisseur. Elle contient un nombre de petites informations diverses :

1. L’utilisateur doit choisir son type de base. Une série de check box seront à sa disposition pour informer le programme de quelle base de chiffre il doit partir pour effectuer la conversion.
2. Après avoir choisi son type, l’utilisateur doit entrer la valeur à convertir. Une fois le bouton convertit cliquer, le programme va vérifier que toutes les informations sont correctes et lancer la méthode « Conversion ». Je décide de mettre ces options car dans le cas d’une recherche automatique du type, si l’utilisateur converti le nombre « 3654 » comment savoir s’il est question d’un nombre décimal, octal ou hexadécimal.
3. Une fois la conversion effectuée, le résultat va s’afficher. Pour gagner du temps, le résultat comprendra toutes les possibilités du programme (Binaire, BCD, Gray, Décimal, Octal et Hexadécimal).
4. Le bouton permettant d’afficher la fenêtre pour les opérations
5. Ce bouton permet d’afficher les informations relatives aux informations et au fonctionnement du logiciel
6. Bouton pour réduire et fermer le programme

La fenêtre pour les opérations ressemble beaucoup à celle de la conversion. Cette page s’ouvre à côté de l’autre et est indépendant de la conversion.

5

1

3

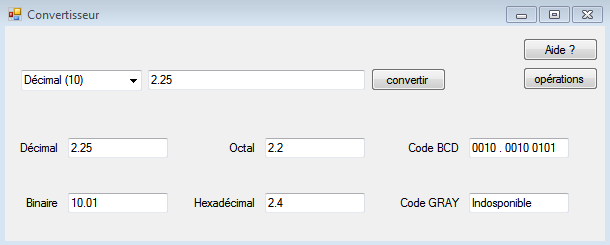
2

4

1. L’utilisateur entre les valeurs et choisi l’opération à effectuer (addition, soustraction et multiplication).
2. L’utilisateur choisi le type des valeurs qu’il a entré au point 1. Comme pour la fenêtre de conversion, le choix est indispensable pour éviter des problèmes dans le futur développement.
3. Une fois les valeurs entrées et le type choisi, l’utilisateur va cliquer sur le bouton « = ». Le programme va alors vérifier les valeurs et les envoyer dans la méthode « Calcul ».

1. Le panneau de résultat va afficher le résultat mais aussi un développement de l’opération pour que l’utilisateur puisse comprendre, de manière graphique, comment l’opération s’effectue.
2. Bouton pour réduire ou fermer la fenêtre de calcul. Entant donné qu’elle est indépendante par rapport à la conversion, si elle est fermée elle ne quitte pas le logiciel.

### Maquette

La maquette du site ressemble beaucoup à la version abordée au point 3.2.2. Cependant plusieurs modifications, discutée avec le chef de projet, viennent améliorer le programme.

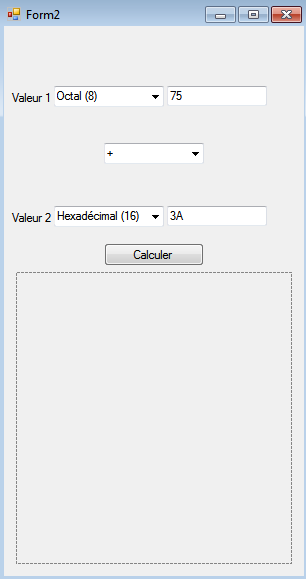
2

1

Un remaniement de l’espace a été envisagé pour permettre une meilleure lisibilité.

1. Les checkboxs de la structure graphique, ont été remplacées par une liste déroulante plus pratique et plus petites à placer sur la fenêtre
2. Petite précision pour le code BCD et GRAY. Tous les formats peuvent être converti en code BCD et inversement.

Le code GRAY peut être converti uniquement en code BCD, et s’obtient uniquement à partir du BCD.

Pour la maquette de la calculatrice, certaines différences font aussi leurs apparitions.

1. Les 2 valeurs possèdent maintenant aussi une liste avec les types pour permettre les opérations avec des valeurs de types différentes.

2

Les valeurs disponibles dans ces opérations sont le binaire, le décimal, l’octal et l’hexadécimal.

1

1. Les opérations sont aussi sous format de liste déroulante pour la même raison que les types des valeurs.

3

3A (16) = 72 (8)

1

75

+ 72

167 (8)

75 (8) = 3D (16)

1

3D

+ 3A

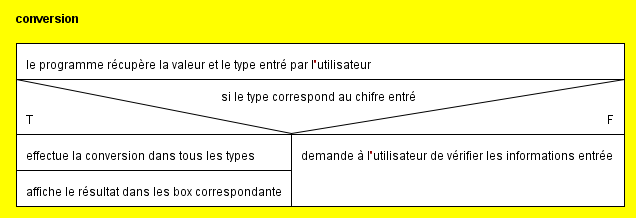
77 (16)

1. Pour les résultats étant donné qu’il faut une base commune pour effectuer les opérations, j’ai choisi de faire les 2 bases. La première version de l’opération sera toujours dans la base de la valeur 1. Et la deuxième dans la base de la valeur 2. Ensuite le programme prendra les deux valeurs pour effectuer l’opération demandée et afficher le détail sous forme d’opération en colonne avec les retenues.

### Structogrammes

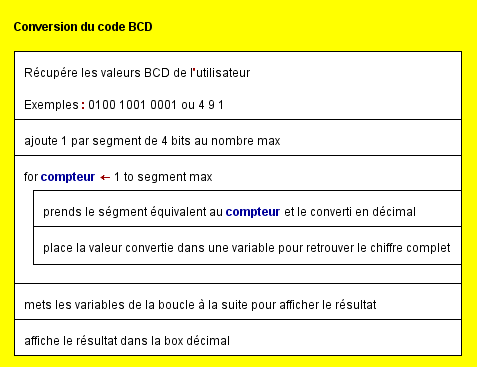
Le programme sera décomposé en 3 parties majeurs. La conversion, le calcul et l’affichage.

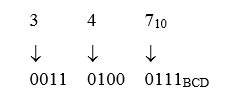
On peut aussi séparer les grandes méthodes en plus petites mais faire un structogramme pour ces étapes intermédiaires ne servirait pas à grand-chose.

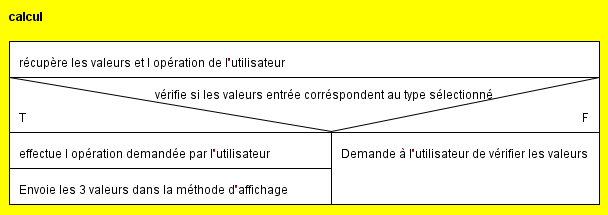
1. La méthode Conversion

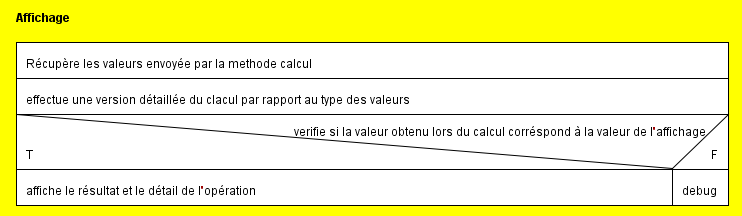
Ce structogramme va me servir pour la conversion binaire, décimal, octal et hexadécimal. Les fonctions fournies par c# suffiront amplement à remplir ce travail.

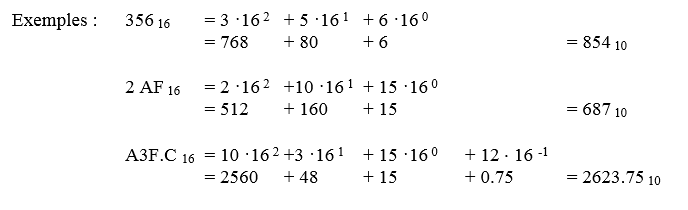
1. BCD et GRAY

Pour le BCD et le code gray les choses se compliques un peu. Pour le code BCD il faut récupérer les valeurs de chaque « segments » de 4 bits afin de les convertir dans les bons formats.

Ce qui suis la méthode présentée dans le document d’ELEOC donné aux élèves.

1. La méthode Calcul
2. La méthode Affichage



Pour la méthode d’affichage, je vais me baser sur le document utilisé en ELEOC. Il arrive à montrer de manière précise comment sont effectuées les opérations.

### Développement

La plus grosse partie de ce TPI réside dans la partie développement du programme. Pour ce faire, je vais utiliser l’ide Visual Studio 2015. Il sera développé en langage c# et suivra les normes de l’ETML.

## Conception des tests

Les tests se dérouleront en 2 parties :

* La grille de tests
* Les tests unitaires

Pour la grille elle comportera une série de manipulations, bonnes ou mauvaises, pour vérifier le bon fonctionnement

## Planification détaillée

A ce stade, après l’analyse complète du projet, un planning détaillé et complet (avec tâches, sous-tâches, dépendances, durée, …) peut être finalisé.

Le planning détaillé doit s’inscrire dans le planning initial. Il faut que l’on puisse situer cette planification détaillée par rapport à la planification initiale.

# Réalisation

## Dossier de Réalisation

Cette partie permet de reproduire ou reprendre le projet par un tiers.

Pour chaque étape, il faut décrire sa mise en œuvre. Typiquement :

Versions des outils logiciels utilisés (OS, applications, pilotes, librairies, etc.)

Configurations spéciales des outils (Equipements, PC, machines, outillage, etc.)

Code source commenté des éléments logiciels développés.

Modèle physique d’une base de données.

Arborescences des documents produits.

Schémas, plans d’adressages, plan de nommage, etc.

Il faut décrire le parcours de réalisation et justifier les choix.

## Modifications

Historique des modifications demandées (ou nécessaires) aux spécifications détaillées.

Date, raison, description, etc.

# Tests

## Dossier des tests

On dresse le bilan des tests effectués (qui, quand, avec quelles données…) sous forme de procédure. Lorsque cela est possible, fournir un tableau des tests effectués avec les résultats attendus et obtenus, ainsi que les actions à entreprendre en conséquence (et une estimation de leur durée).

Si des tests prévus dans la stratégie n'ont pas pu être effectués :

raison, décisions, etc.

Liste des bugs répertoriés avec la date de découverte et leur état:

Corrigé, date de correction, corrigé par, etc.

# Conclusion

## Bilan des fonctionnalités demandées

Il s’agit de reprendre point par point les fonctionnalités décrites dans les spécifications de départ et de définir si elles sont atteintes ou pas, et pourquoi.

Si ce n’est pas le cas, estimer en « % » ou en « temps supplémentaire » le travail qu’il reste à accomplir pour terminer le tout.

## Bilan de la planification

Distinguer et expliquer les tâches qui ont généré des retards ou de l'avance dans la gestion du projet. Indiquer les différences entre les planifications initiales et détaillées avec le journal de travail.

## Bilan personnel

Si c’était à refaire:

Qu’est-ce qu’il faudrait garder ? Les plus et les moins ?

Qu’est-ce qu’il faudrait gérer, réaliser ou traiter différemment ?

Qu’est que ce projet m’a appris ?

Suite à donner, améliorations souhaitables, …

Remerciements, signature, etc.

# Divers

## Journal de travail

Date, activité (description qui permet de reproduire le cheminement du projet), durée, liens et références sur des documents externes. Lorsqu’une activité de recherches a été entreprise, il convient d’énumérer ce qui a été trouvé, avec les références.

## Bibliographie

Références des livres, revues et publications utilisés durant le projet.

## Webographie

* Analyse concurrentielle
  + [www.aly-abbara.com/utilitaires/convertisseur/convertisseur\_chiffres.html](http://www.aly-abbara.com/utilitaires/convertisseur/convertisseur_chiffres.html)
  + [www.sebastienguillon.com/test/javascript/convertisseur.html](http://www.sebastienguillon.com/test/javascript/convertisseur.html)

# Annexes

Cahier des charges

Listing du code source (partiel ou, plus rarement complet)

Guide(s) d’utilisation et/ou guide de l’administrateur

Etat ou « dump » de la configuration des équipements (routeur, switch, robot, etc.).

Extraits de catalogue, documentation de fabricant, etc.