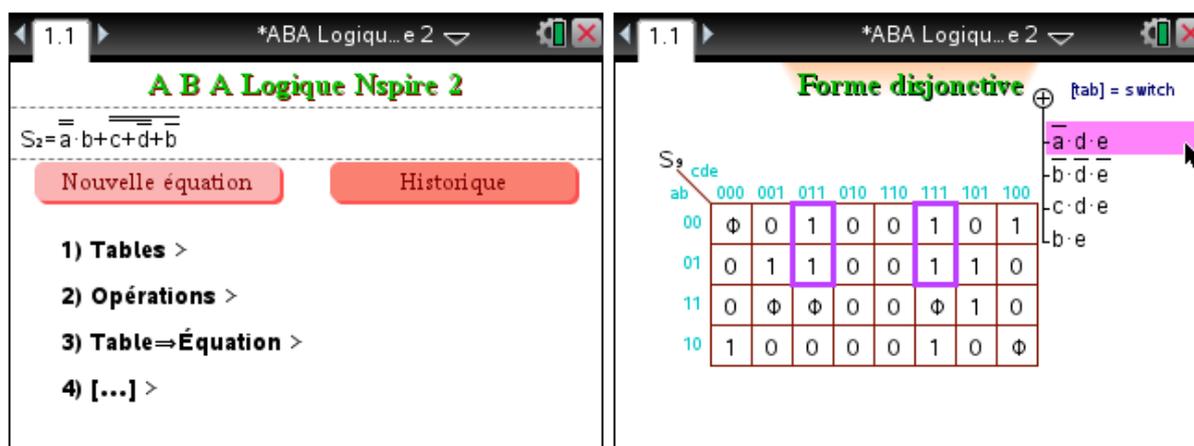


A B A Logique Nspire 2



Retrouvez mes autres programmes sur :

⇒ [Mes programmes - TI Planet.org](http://Mesprogrammes-TIPlanet.org)

Sommaire :

- 1) Description générale
- 2) Changements de la nouvelle version (changelog)
- 3) Installation
- 4) Description des fonctions et remarques

1) Description générale

A B A Logique Nspire 2 est un programme assez complet sur les équations logiques et l'algèbre de Boole. Il est l'aboutissement de plusieurs années de développement (par intermittence, je vous rassure ☺) et il s'agit ici de la deuxième version sur TI Nspire. (cf le changelog en 2) pour les nouveautés)

Notons que ABA Logique a remporté le Concours Lua TI-Planet/Inspired Lua 2011 dans la catégorie mathématiques !

Il existe une version pour TI 89, correspondant plutôt au niveau de la première version d'ABA Logique Nspire : ABA Logique v2.0

Voici les fonctionnalités principales de ABA Logique Nspire : (voir la partie 3) pour l'explication détaillée de chaque fonctionnalités)

Si l'on saisit une certaine expression, du style $//a*b+/(c+d)+b$ (il peut y avoir jusqu'à 6 variables avec les lettres de a à f) :

- Vous pouvez entrer votre équation plus efficacement en profitant de l'**auto-complétion** des « * » implicites.
- L'équation saisie est affichée en "**pretty print**".
- On peut visualiser le **tableau de Karnaugh**, la **table de vérité naturelle** et la **table de vérité en code Gray** de l'équation entrée.
- On peut **vérifier l'équivalence** de notre équation avec une autre forme, si on n'est pas sûr d'une simplification, ou d'une forme sensée être équivalente.
- On peut **simplifier** l'équation par tableau de Karnaugh, sous forme normale disjonctive ou conjonctive, avec l'affichage interactif des « blocs » de simplification.
- On peut transformer l'équation pour utilisation "**ladder**" (en circuit électrique) c'est à dire avec des variables uniques complémentées, par le théorème de De Morgan.
- On peut transformer l'équation pour une utilisation avec des **cellules universelles NAND**. (grâce au théorème de De Morgan)
- On peut aussi obtenir l'équation réduite **d'un tableau de Karnaugh ou d'une table de vérité** donnés, prenant même en charge des états indéterminés (les « don't care », notés « ϕ »)
- Toutes les équations entrées et les résultats sont stockés dans un **historique** que l'on peut rappeler !
- Cette version d'ABA Logique offre une ergonomie et une présentation travaillée (en particulier si vous avez la couleur avec une Nspire CX). Vous pouvez parcourir les menus à travers trois moyens différents : flèches, curseur, touches alphanumériques correspondant aux items.
- Compatibilité avec le mode ordinateur du Student Software. (vous pouvez utiliser le clic droit de la souris, sur l'ordinateur, à la place de [esc])

Remarque : le programme est sensé fonctionner sur toute calculatrice TI Nspire munie d'un OS ≥ 3.0 . Je n'ai cependant pas pu tester le rendu sur Nspire monochrome, le programme a été développé sur Nspire CX CAS, en couleur.

2) Changements de la nouvelle version (changelog)

Il s'agit de la deuxième version d'ABA Logique Nspire. La première datant de septembre 2011. Voici les nouveautés de cette version 2, release en septembre 2013 :

- Compatibilité OS ≥ 3.2 assurée : ces trois problèmes principaux ont été réglés :
 - Les nouveautés au niveau des opérateurs logiques dans l'OS 3.2 ont rendu nombre des fonctionnalités d'ABA Logique défailtantes.
 - Le comportement de la fonction Lua « math.eval » a aussi changé en défaveur du fonctionnement d'ABA Logique.
 - La suppression de la fonction de transparence « gc:setAlpha » utilisée par ABA Logique qui provoquait alors un bug sous l'OS 3.2.

- Nouveau moteur de simplification : simplification interactive par tableau de Karnaugh. L'ancien moteur de simplification algébrique a été remplacé par un algorithme de simplification par tableaux de Karnaugh. Il a plusieurs avantages :
 - simplification plus fiable, sous forme normale disjonctive ou conjonctive au choix, avec une factorisation proposée.
 - simplification plus rapide
 - présentation interactive des « blocs » de simplification correspondant à chaque terme de l'équation réduite, pour la forme normale disjonctive ou conjonctive au choix
 - compatibilité avec les Nspire **non CAS** !
 - prise en charge des états indéterminés (les « don't care », notés « ϕ ») de manière optimale
- Compatibilité complète avec les modèles non CAS
- Prise en charge des nouveaux opérateurs logiques de l'OS 3.2 : les opérateurs \Rightarrow et \Leftrightarrow sont supportés. (uniquement sur OS ≥ 3.2)
- La gestion de l'historique a été revue : avec notamment une numérotation en indice des équations entrées, permettant de retrouver plus facilement les formes sensées être équivalentes. (ex : $S_4 = \dots$)

3) Installation

Il vous faut donc une TI Nspire quelconque munie d'un **OS 3.0 ou supérieur** qui supporte le Lua, le logiciel **TI Nspire Student Software** (ou teacher) ou le **TI Nspire Computer Link** disponible gratuitement, et le câble ordi-calculatrice bien sûr. (USB \leftrightarrow mini USB B)

Dézippez au préalable le classeur ABA Logique Nspire 2.tns contenu dans l'archive que vous avez téléchargée.

> Pour le Student Software, ouvrez le classeur ABA Logique Nspire 2.tns. Vous pouvez d'ores et déjà tester le programme sur l'ordinateur ! Pour l'envoyer sur la calculatrice, sélectionnez Fichier>Enregistrer sur l'unité.

> Pour le computer link, transférez le classeur en faisant un cliquer-glisser de l'ordinateur vers le dossier voulu sur votre unité.

4) Description des fonctions et remarques

- Pour entrer une équation :

Les variables à utiliser vont de a à f . Pour écrire une équation, utilisez les opérateurs suivants :

* : ET	
+ : OU	
/ : NON	/\ La barre se met avant la variable à compléter : $/a = \bar{a}$; $/(a*b) = \overline{a * b}$
() : parenthèses	
Et uniquement pour OS ≥ 3.2 :	
=> : implique	$a \Rightarrow b = /a + b$
<=> : équivaut	$a \Leftrightarrow b = /a * /b + a * b$
N.B. : cela permet aussi le « ou exclusif » : $/(a \Leftrightarrow b) = /a * b + a * /b$	

Vous pouvez profiter de l'auto-complétion des « * » implicites pour faciliter la saisie. Par exemple :

$$ab + b/c + a/(cb + d/e)(e + /c)d \Rightarrow a*b + b*/c + a*/(c*b + d*/e)*(e + /c)*d$$

⇒ Vous pouvez également rappeler une équation précédente ou un résultat dans l'historique !

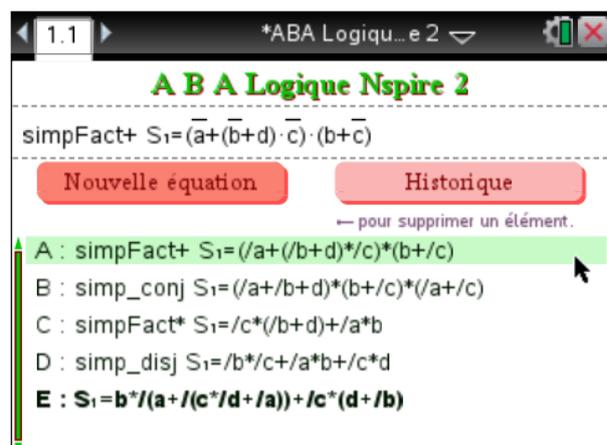
- L'historique

Chaque équation entrée ou résultat est stocké à la tête de l'historique, sans redondance sur les 4 dernières entrées. C'est-à-dire qu'avant qu'une nouvelle équation soit entrée dans l'historique, le programme vérifie si elle est identique à l'une des 4 dernières expressions. Si c'est le cas, pour éviter un encombrement inutile de l'historique, l'équation n'est alors pas ajoutée. Chaque élément de l'historique comporte un en-tête pour identifier la nature du résultat. (ex : simp_disj $S_1 = a \cdot b + c$ dénote la simplification en forme normale disjonctive de S_1) Notons aussi que l'équation courante est affichée en gras.

La numérotation (en indice de S) des équations fonctionne ainsi :

- L'indice est incrémenté lorsque l'on entre une nouvelle équation, soit par la saisie de son expression, soit par le remplissage d'une table.
- L'indice reste le même lorsque l'on effectue une opération sur l'équation. Ainsi, les équations de même indice sont normalement équivalentes.

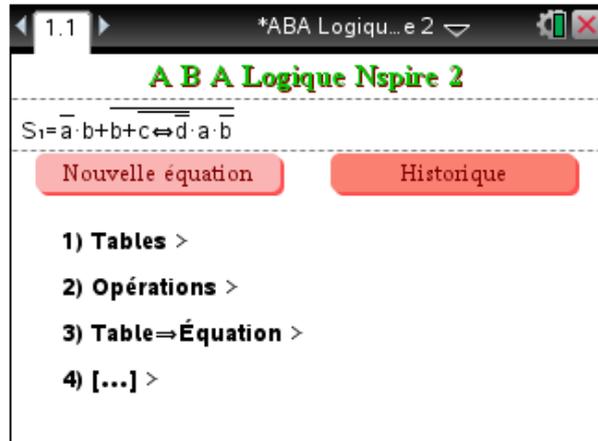
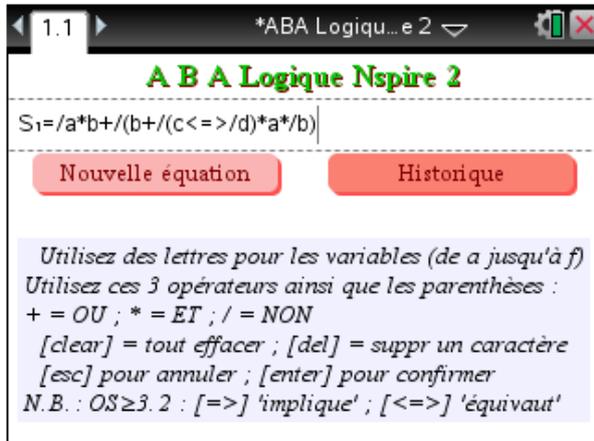
Attention cependant lors de la saisie d'une table avec des « ϕ », les formes disjonctive ou conjonctive résultantes peuvent ne pas regrouper les « ϕ » de la même manière et donc ne pas être équivalentes ! (cf : [Équation à partir d'une table](#) pour un exemple)



- L'affichage en « pretty print »

Les équations entrées et les résultats sont affichés en écriture naturelle ! La lecture devient bien plus aisée et évite les erreurs.

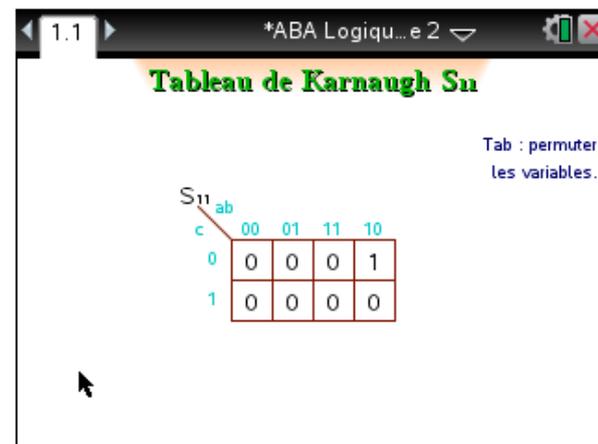
Ex :
$$S_1 = \overline{a} \cdot b + \overline{c} + \overline{d} + \overline{b}$$



- Tableau de Karnaugh :

L'affectation des variables pour chaque ligne/colonne est bien indiquée en bleu ciel. Il y a généralement deux ordres utilisés pour répartir les variables du tableau de Karnaugh... ABA Logique vous permet de passer de l'un à l'autre, selon vos habitudes de présentation, en appuyant sur [tab] lorsque vous visualisez le tableau.

Ex :
$$S_{11} = a \cdot \overline{b} + c$$



- Tables de vérité :

On peut l'obtenir en code **binaire naturel** ou en code **Gray (binaire réfléchi)** On peut la faire défiler avec les flèches, avec le curseur sur les extrémités de la barre de défilement, ou avec la molette de la souris (sur ordi).

a	b	c	d	S ₂
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0

a	b	c	d	S ₂
0	0	0	0	0
0	0	0	1	1
0	0	1	1	0
0	0	1	0	0
0	1	1	0	0
0	1	1	1	0
0	1	0	1	1
0	1	0	0	0

- Vérifier équivalence :

Cela permet de vérifier l'équivalence de deux équations sous une forme différente. Il faut entrer les deux équations ou les rappeler dans l'historique. Appuyez sur [tab] pour valider la saisie de chaque équation.

Leurs tableaux de Karnaugh seront comparés et affichés.

EX : $(a*b)$ et $(a+b)$ \Rightarrow équivalentes !

- Simplification :

Permet d'obtenir la forme réduite d'une expression logique, par la méthode des tableaux de Karnaugh. La solution est présentée de façon interactive !

Résultat sous deux formes réduites possibles :

- Forme normale disjonctive : somme de produits ; + une factorisation selon la loi « * »
- Forme normale conjonctive : produit de sommes ; + une factorisation selon la loi « + »

La touche [tab] permet de passer d'une forme à l'autre.

Simplification

◀ *Forme normale disjonctive* ▶

$S_4 = \bar{a} \cdot \bar{b} + a \cdot b + a \cdot c$

◀ *factorisation selon * :* ▶

$S_4 = a \cdot (b+c) + \bar{a} \cdot \bar{b}$

Appuyez sur [tab] pour choisir la forme de la simplification.

Appuyez sur [enter] pour visualiser la méthode de résolution !

Simplification

◀ *Forme normale conjonctive* ▶

$S_4 = (\bar{a} + b + c) \cdot (a + \bar{b})$

◀ *factorisation selon + :* ▶

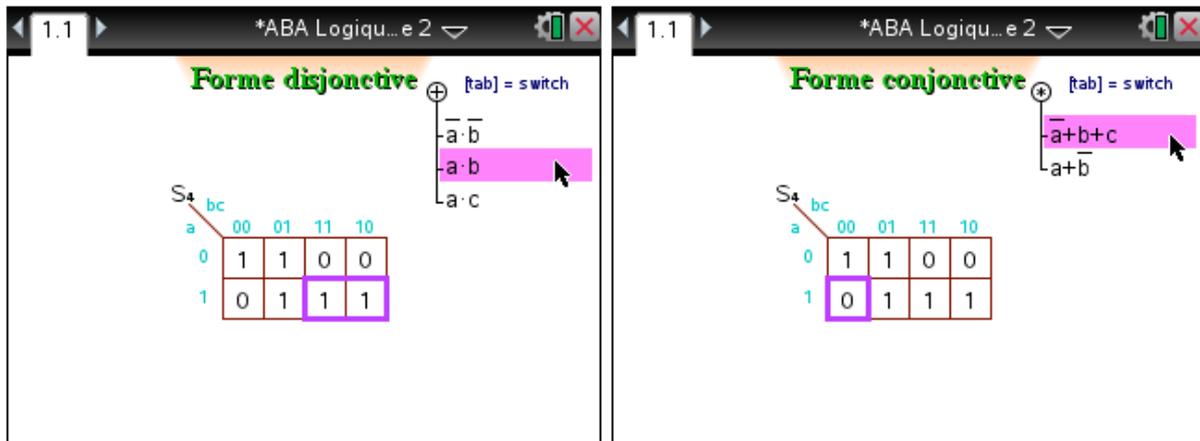
$S_4 = (\bar{a} + b + c) \cdot (a + \bar{b})$

Appuyez sur [tab] pour choisir la forme de la simplification.

Appuyez sur [enter] pour visualiser la méthode de résolution !

Chaque forme et sa factorisation sera stockée dans l'historique, sans redondance (cf : [L'historique](#)), dans cet ordre avec les en-têtes suivants : simpFact+ ; simp_conj ; simpFact* ; simp_disj.

Le tableau de Karnaugh de la fonction est ensuite affiché et les « blocs » de simplification sont visualisables en sélectionnant, avec les flèches ou le curseur, le terme voulu !



- [Transfo pour Ladder :](#)

Pour Ladder, cela veut dire pour circuits électriques... Il s'agit de transformer, grâce au Théorème de De Morgan, les groupes de variables complémentées en variables uniques complémentées. (complémenté veut dire NON) Cela est en effet très utile pour les schémas électriques car on préfère câbler par exemple $/a+/b$ (deux contacts normalement fermés en parallèle) que

$/(a*b)$ (deux contacts normalement ouverts en série qui actionnent un relai qui ouvre le circuit... :|)

Le programme tient compte des subtilités avec les parenthèses, les simplifiant aussi lorsque c'est possible :

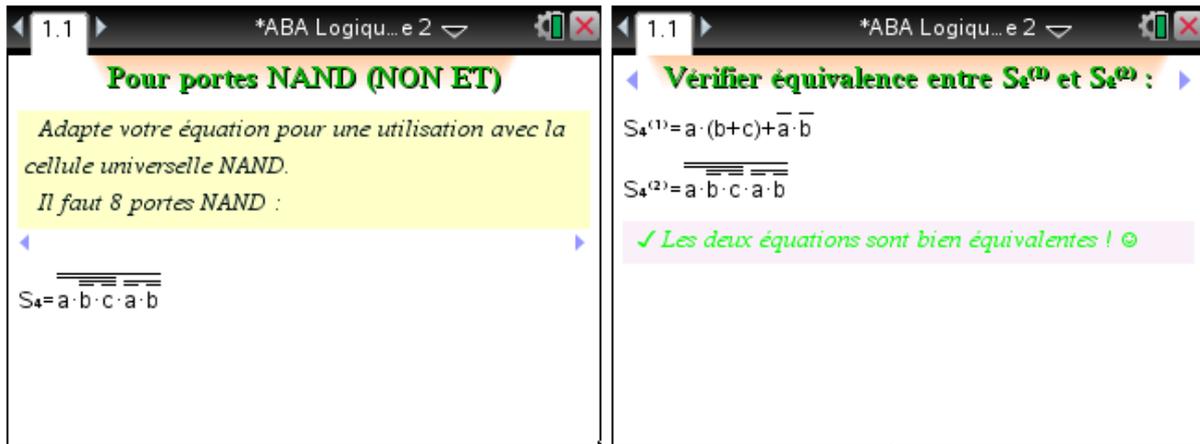
ex : $/(a+b*c) = /a*/(b*c) = /a*/(b+/c)$

ex2 : $/(a+(b+d)*c) = /a*/((b+d)*c) = /a*/((b+d)+/c) = /a*/(b*/d+/c)$

- Transfo pour NAND :

Cela adapte votre équation pour une utilisation avec la cellule universelle NAND. Le programme précise le nombre de portes nécessaires !

Ex : $S = a \cdot (b + c) + \bar{a} \cdot \bar{b}$



- Équation à partir d'une table (tableau de Karnaugh, table de vérité naturelle ou gray)

Cela permet d'obtenir l'équation réduite optimale d'une table de vérité ou d'un tableau de Karnaugh. Il est possible de signaler des états indéterminés (« don't care », notés « ϕ »). L'équation la plus simple sera obtenue par la résolution du tableau de Karnaugh associé.

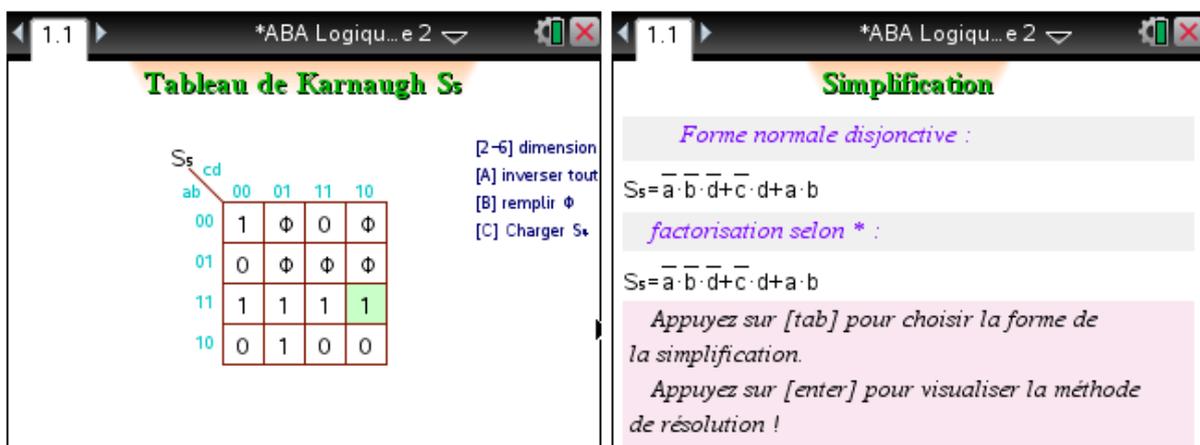
Naviguez dans le tableau avec les flèches ou avec la souris. Pour entrer une valeur, tapez 0, 1 ou n'importe quelle autre touche pour « ϕ » (je recommande « . »). Vous pouvez aussi cliquer à la souris : cela change la case selon le cycle : 0,1, ϕ , ...

Pour choisir les dimensions du tableau (nombre de variables) appuyez à tout moment sur la touche correspondant au nombre de variables voulu. (2,3,4,5,6)

Vous pouvez aussi :

- [A] = inverser tout
- [B] = remplir de « ϕ » (états indéterminés)
- [C] = charger la table de l'équation courante

Exemple pour le tableau de Karnaugh :



Forme disjonctive [tab] = switch

	00	01	11	10
00	1	Φ	0	Φ
01	0	Φ	Φ	Φ
11	1	1	1	1
10	0	1	0	0

$\overline{a} \cdot \overline{b} \cdot \overline{d}$
 $\overline{c} \cdot d$
 $a \cdot b$

Forme conjonctive [tab] = switch

	00	01	11	10
00	1	Φ	0	Φ
01	0	Φ	Φ	Φ
11	1	1	1	1
10	0	1	0	0

$\overline{a+b}$
 $\overline{a+b+d}$
 $b+\overline{c}$

!/ Attention : dans l'exemple ci-dessus, les formes conjonctive et disjonctive ne seront pas équivalentes !!

Exemple pour la table de vérité en binaire naturel :

Table de vérité ★ Binaire naturel

a	b	c	S ₇
0	0	0	1
0	0	1	Φ
0	1	0	0
0	1	1	Φ
1	0	0	Φ
1	0	1	1
1	1	0	1
1	1	1	0

[2-6] dimension
 [A] inverser tout
 [B] remplir Φ
 [C] Charger S₇

Forme disjonctive [tab] = switch

	00	01	11	10
0	1	Φ	Φ	0
1	Φ	1	0	1

$\overline{a} \cdot \overline{c}$
 \overline{b}

Voilà ! Bonne utilisation, et si malgré tout le soin apporté à la réalisation de ce programme, vous avez un souci, des questions ou quelque idée d'amélioration, n'hésitez pas à m'en faire part :

aphone@hotmail.fr

Ou bien rendez vous sur le forum de TI-Planet, j'y suis membre régulier !

ABA Logique Nspire 2

Louis DURAND

Copyright 2009-2013

France 54

Participation au concours Lua TI Planet/Inspired Lua 2011