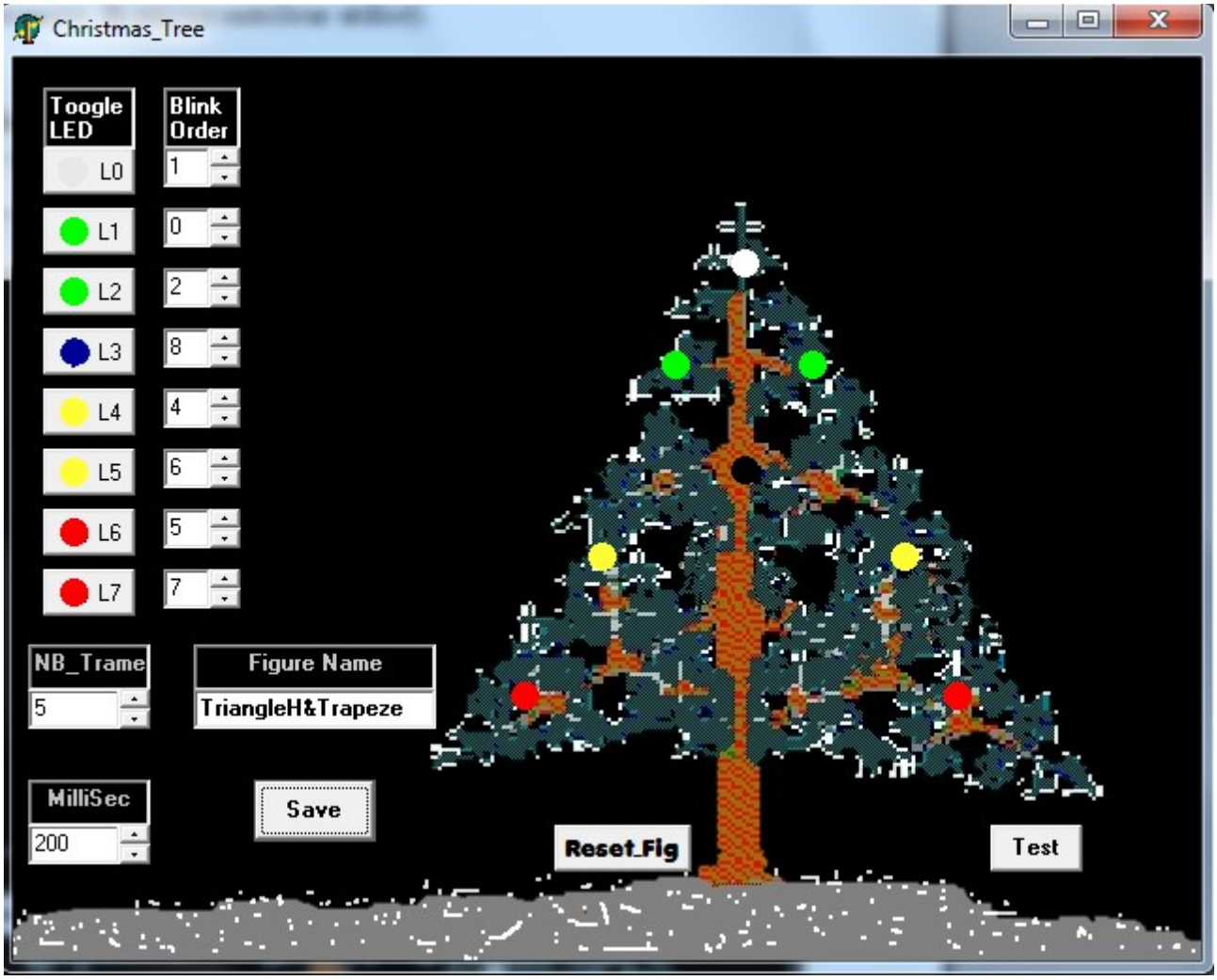
TREE8LED Version 1.0 27/07/2015 (delphi 7)

Le but du programme TREE8LED est de créer des animation pour LED avec la souris ou le clavier chaque séquence d'animation peut être sauvegardée sous forme d'un fichier TXT qui s'intégrera dans un programme C pour Arduino ou AVR Studio destiné à la gamme des microcontrôleurs ATtiny13 /45 /85.
Un Circuit Imprimé en forme de mini sapin de Noël composé de huit LED quatre résistances et un microcontrôleur (au format SMD dans mon cas) a été spécialement conçu pour afficher plusieurs séquences d'animation (limitées à la mémoire du microcontrôleur utilisé).

IMPORTANT :
Mettre toutes les case Blink Order à 8 (avec le bouton Reset_Fig)avant de commencer une figure ou pour remettre à zéro une figure, sinon les figures que vous créerez seront fausses.

Le temps en MilliSec doit être adapté car des microcontrôleur comme l'Attiny13 et de l'Attiny85 ont un oscillateur interne qui fonctionne à une fréquence différente.
En fin de tutoriel, vous trouverez les informations pour les vitesses Oscillateur et la programmation des FusesBit.



screen1

LES BOUTONS :

Reset_Fig
ce bouton remet toutes les cases numériques à zéro (8)
NB_Trame
ce chiffre indique le nombre de fois que la figure sera affichée.

MilliSec
Ce chiffre indique le temps d'allumage par LED en Milliseconde

Figure Name
C'est sous ce nom que la figure sera sauvegardée avec l'extension .txt (TriangleH&Trapeze.txt sur la capture d'écran)

Toogle LED (Basculer la LED)
Ce bouton affiche l'état des huit LED et agit comme un interrupteur qui sert à allumer ou à éteindre une LED, **ces boutons ont comme utilité de donner la position des LED** sur le sapin ce qui donne une idée de ce que sera la figure sur le circuit électronique **mais ne servent pas à la programmation de l'ordre des LED affichées par le microcontrôleur.**
L0 représente la LED 0, L1 la LED 1 …..... L7 la LED 7

Blink Order (Ordre de clignotement)
Ce bouton à flèches (incrémente / décrémente) est important car **c'est lui qui gère l'ordre d'allumage et d'extinction des LED** (0 pour la première et 7 pour la dernière, 8 indique une pose égale au temps en milli Seconde),
5 indique que la figure se répète 5fois
ensuite on voit que L1 s'allume en 1er puis L0 puis L2 puis il y a une pause de 200mS car L3 ne s'allume pas puis L4 s'allume puis L6 puis L5 puis L7).

Quand tous ces paramètres conviennent on peut cliquer sur le bouton **Save** et le fichier est créé dans le répertoire courant.

Voici le fichier généré pour être intégré dans la procédure SomeOne pour Arduino ou AVR Studio : que l'on voit sur la capture d'écran :
SomeOne (5,1,0,2,8,4,6,5,7,200); //TriangleH&Trapeze

la vitesse d'affichage de 200 milli Secondes est pour l'Attiny13, si vous utilisez un Attiny45 ou 85 il faut multiplier la vitesse par 8 ou 10 environ.

Voici le tableau d'exemple pour les Attiny13 et Attiny85 :

Figures sur un Attiny13 :

```
 SomeOne (8,0,1,2,3,4,6,5,7,150); // une par une tres lent
 SomeOne (8,0,1,2,3,4,6,5,7,65); // une par une moyen
 SomeOne (15,0,1,2,3,4,6,5,7,25); // une par une rapide
 SomeOne (90,0,1,2,3,4,5,6,7,1);  // flash
 SomeOne (8,1,2,6,4,3,8,8,0,35);     //trapèze&L4 Rapide
 SomeOne (10,0,1,2,8,4,5,6,7,40);    //grandTriangleparL1
```

Figures sur un Attiny45 ou 85 :

```
 SomeOne (8,0,1,2,3,4,6,5,7,1200); // une par une très lent (150 sur Attiny13)
 SomeOne (8,0,1,2,3,4,6,5,7,600); // une par une moyen
 SomeOne (8,0,1,2,3,4,6,5,7,300); // une par une rapide
 SomeOne (80,0,1,2,3,4,6,5,7,0);  // flash
 SomeOne (8,1,2,6,4,3,8,8,0,600);     //trapèze&L4 Rapide
 SomeOne (10,0,1,2,8,4,5,6,7,500);    //grandTriangleparL1
```

Voici le programme complet pour Attiny13, pour le 85 seules les vitesses d'affichage sont à modifier comme ci-dessus :

```
/*
 * TreeOneByOne.ino For ATtiny 13
 *
 * Created: 29/05/2015 14:42:12 Arduino 1.04
 *  Author: Mic-Josi
 * for Tiny Tree Rev. 1.1 and 1.7
 * pour Tiny Tree Rev. 1.1 et 1.7
 * configuration sélection oscillateur interne 4.8 Mhz et division par 8 coupure alimentation à 1.8 Volts
 * setting internal oscillator 4,8 MHz and division by 8 selected brown-out level detection at 1.8 Volts
 * FuseBits: Low = 0x69 Hight = 0xFD
 */


/*
These inused #define are for using with Amtel Studio 6 only
Ces defines sont à utiliser uniquement avec Amtel Studio 6
# define __DELAY_BACKWARD_COMPATIBLE__
# define F_CPU 8000000UL
# include <avr/io.h>
# include <util/delay.h>
# define byte uint8_t
```

```
*/

//Arduino defines
#include <avr/pgmspace.h>
#define LED_COUNT 9
#define DDR_BYTE 0
#define PORT_BYTE 1


const byte matrix[LED_COUNT][2] PROGMEM = {

        // DDR_BYTE     PORT_BYTE
        {0b00010001     , 0b00010000},//L0-PB4            0
        {0b00010001     , 0b00000001},//L1-PB4          "     "
        {0b00001001     , 0b00001000},//L2-PB3        " 1    2 "  **     Pin-Tree     **
        {0b00001001     , 0b00000001},//L3-PB3       "      3     "       L0+ L1- Pin 3
        {0b00000101     , 0b00000100},//L4-PB2     " 4           6 "     L2+ L3- Pin 2
        {0b00000101     , 0b00000001},//L5-PB2    "                "     L4+ L5- Pin 7
        {0b00000011     , 0b00000010},//L6-PB1    " 5              7  " L6+ L7- Pin 6
        {0b00000011     , 0b00000001},//L7-PB1   ** +- Commun PB0 Pin 5 **

};


//*********************************************************
void turnOn( byte led ) // PBx selection
{
  DDRB = pgm_read_byte (&( matrix[led][DDR_BYTE]));
  PORTB = pgm_read_byte (&(matrix[led][PORT_BYTE]));
}
//*********************************************************


//*********************************************************
// frame- nombre de fois la figure, l1 à l8 = état des LED, spd = temps d'llumage d'une LED en mili-second
void SomeOne (byte frame,byte l1,byte l2,byte l3,byte l4,byte l5,byte l6,byte l7,byte l8,byte spd) // some LEDs ON one by one spd time
{

    int figure [9]= {0,0,0,0,0,0,0,0,0};

        int light = 0;
```

```
int allume = 0;
byte l9 = 8;
while (light < frame)
{
	for( byte l = 0; l < LED_COUNT; l++ )

	{
		switch (l)
		{
		case 0:
		figure [l] = l1;
		break;

		case 1:
		figure [l] = l2;
		break;

		case 2:
		figure [l] = l3;
		break;

		case 3:
		figure [l] = l4;
		break;

		case 4:
		figure [l] = l5;
		break;

		case 5:
		figure [l] = l6;
		break;

		case 6:
		figure [l] = l7;
		break;

		case 7:
		figure [l] = l8;
		break;

		case 8:
		figure [l] = l9;
```

```
                    break;
                    }


                }
                for( byte l = 0; l < LED_COUNT; l++ )
                {
                 allume = figure[l];
                 turnOn(allume);
                 delay(spd);
                }

                light ++;
        }
        delay(50);

}

 //*******************************************************



void setup()
{
}


//*******************************************************

//=======================================================
void loop()
{
  /* 0 à 7 emplacement d'allumage de la LED,8 LED éteinte
    0 to 7 lighting location of the LED, 8 LED off */

  SomeOne (8,0,1,2,3,4,6,5,7,150); // une par une tres lent
  SomeOne (8,0,1,2,3,4,6,5,7,65); // une par une moyen
  SomeOne (15,0,1,2,3,4,6,5,7,25); // une par une rapide
  SomeOne (90,0,1,2,3,4,5,6,7,1);  // flash
  SomeOne (8,0,1,2,8,8,8,8,8,70);// petit triangle haut lent
  SomeOne (8,1,2,3,8,1,2,3,8,49);// petit triangle haut inversé lent0
  SomeOne (8,5,4,1,0,0,2,6,7,42); // grand triangle lent
  SomeOne (8,8,5,4,1,3,2,6,7,46); //grand M rapide
```

```
  SomeOne (8,1,2,6,4,3,8,8,0,35);     //trapèze&L4 Rapide
  SomeOne (10,0,1,2,8,4,5,6,7,40);    //grandTriangleparL1
    //SomeOne (8,0,2,1,8,4,6,5,7,150);     //02184657


}

//============================================================
```

ci-dessous, la copie d'écran pour les FusesBit de ces microcontrôleurs :
Voici une copie d'écran des fusibles de configuration pour les Attiny 13 et 13A Attiny45 et 85 la copie d'écran est celle de l'outil de configuration des AVR dont voici le lien :
http://www.engbedded.com/fusecalc/

FusesBIT pour Attiny13 et 13A

www.engbedded.com/fusecalc/ (141 parts currently listed)

Feature configuration

AVR part name: ATtiny13A Select

This allows easy configuration of your AVR device. All changes will be applied instantly.

Features

Int. RC Osc. 4.8 MHz; Start-up time: 14 CK + 64 ms; [CKSEL=01 SUT=10]

- [x] Divide clock by 8 internally; [CKDIV8=0]
- [ ] Watch-dog Timer always on; [WDTON=0]
- [ ] Preserve EEPROM memory through the Chip Erase cycle; [EESAVE=0]
- [x] Serial program downloading (SPI) enabled; [SPIEN=0]
- [ ] Reset Disabled (Enable PB5 as i/o pin); [RSTDISBL=0]

Brown-out detection level at VCC=2.7 V; [BODLEVEL=01]

- [ ] Debug Wire enable; [DWEN=0]
- [ ] Self Programming enable; [SELFPRGEN=0]

Apply feature settings

Manual fuse bits configuration

This table allows reviewing and direct editing of the AVR fuse bits. All changes will be applied instantly.

Note: ☐ means unprogrammed (1); ☑ means programmed (0).

Bit	*Low*	*High*
7	☑ **SPIEN** SPI programming enable	
6	☐ **EESAVE** Keep EEprom contents during chip erase	
5	☐ **WDTON** Watch dog timer always on	
4	☑ **CKDIV8** Start up with system clock divided by 8	☐ **SELFPRGEN** Self Programming Enable
3	☐ **SUT1** Select start-up time	☐ **DWEN** DebugWire Enable
2	☑ **SUT0** Select start-up time	☑ **BODLEVEL1** Enable BOD and select level
1	☑ **CKSEL1** Select Clock Source	☐ **BODLEVEL0** Enable BOD and select level
0	☐ **CKSEL0** Select Clock Source	☐ **RSTDISBL** Disable external reset

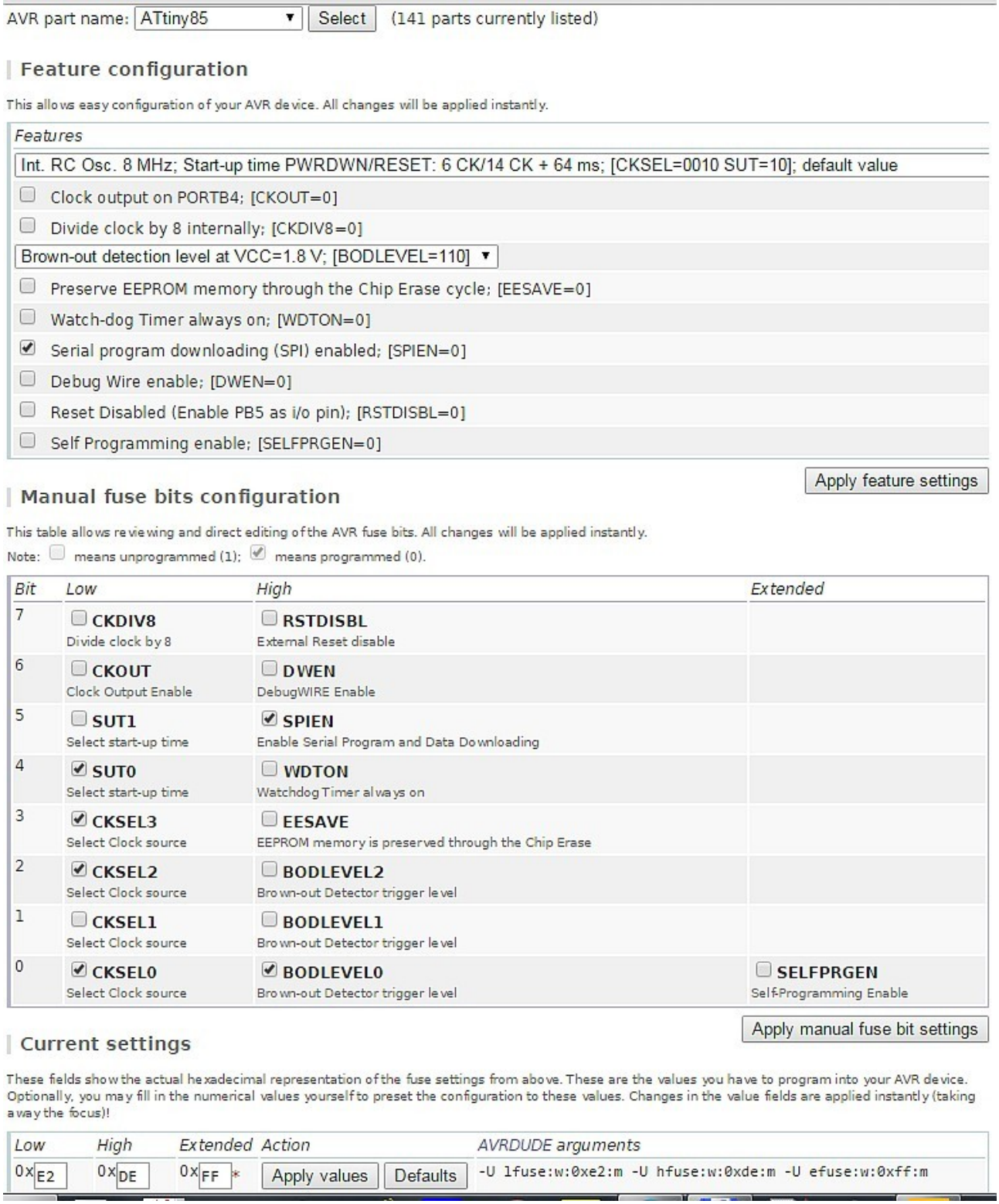
Apply manual fuse bit settings

Current settings

These fields show the actual hexadecimal representation of the fuse settings from above. These are the values you have to program into your AVR device. Optionally, you may fill in the numerical values yourself to preset the configuration to these values. Changes in the value fields are applied instantly (taking away the focus)!

Low	*High*	*Action*	*AVRDUDE arguments*
0x 69	0x FB *	Apply values Defaults	`-U lfuse:w:0x69:m -U hfuse:w:0xfb:m`

FusesBIT pour Attiny45 et 85



Voici le lien pour les téléchargements :

les fichiers Delphi, KiCad, MPLAB, AMTEL STUDIO et la vidéo de démo
http://1drv.ms/1OwZtzi
Le logiciel KiCad :
http://iut-tice.ujf-grenoble.fr/kicad/
librairies additionnelles pour KiCad :
http://www.kicadlib.org/