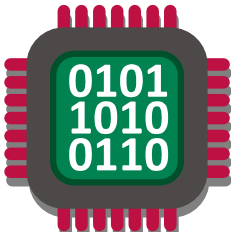


Microcontrôleur ATMEL AVR8

Étalonnage de l'oscillateur RC

http://github.com/epsilonrt/avr-calibrate

1. Présentation

La procédure d'étalonnage de l'oscillateur RC d'un microcontrôleur AVR8 est décrite dans la note d'application ATMEL AVR053.

Cette note d'application n'explique pas la mise en œuvre de cette procédure avec un AVR Dragon et ne prend pas en charge les modèles de microcontrôleur récents comme les tiny441/841, mega168P/328P et mega328PB. Peu d'informations sont disponibles sur le web.

Ce document a donc pour objectif de décrire la procédure d'étalonnage d'un oscillateur RC avec AVR Dragon.

1.1. Principe de l'étalonnage

Le réglage de l'oscillateur RC d'un microcontrôleur AVR8 se fait en modifiant le registre OSCCAL. ATMEL effectue un étalonnage en sortie de chaîne de fabrication. Cette valeur d'étalonnage est celle que l'on peut lire dans le registre OSCCAL après le RESET du microcontrôleur.

Pour améliorer la précision, il est possible de procéder à un nouvel étalonnage en calculant une nouvelle valeur de OSCCAL. Une fois calculée, cette valeur sera stockée en EEPROM et utilisée par le programme du microcontrôleur pour ajuster le réglage.

Deux broches sont utilisées pour l'étalonnage : MOSI et MISO pour une interface ISP, ou TDI et TDO pour une interface JTAG. Pour simplifier, nous utiliserons MOSI et MISO dans l'explication ci-dessous mais elle reste valable pour TDI et TDO.

Le principe de base est que l'AVR Dragon génère une horloge (à 32768 Hz) sur MOSI et que le microcontrôleur utilise cette horloge pour calibrer son oscillateur RC. Quand le microcontrôleur a terminé son étalonnage avec succès, il stocke la valeur calculée de OSCCAL en EEPROM et transmet OK sur MISO.

Pour effectuer un étalonnage, il faut donc :

1. Régler les fusibles du microcontrôleur pour une configuration en oscillateur interne RC, sans division par 8. Il est préférable d'activer le fusible de protection de l'EEPROM
2. Programmer le microcontrôleur avec le *firmware* d'étalonnage (avr053)
3. Effectuer l'étalonnage grâce à un logiciel PC fournit par ATMEL

La valeur d'étalonnage stockée en EEPROM sera utilisée par le programme final du microcontrôleur pour initialiser OSCCAL.

1.2. Installation

Le *firmware* d'étalonnage se trouve dans le fichier AVR053.zip . Il peut être téléchargé sur le site d'ATMEL.

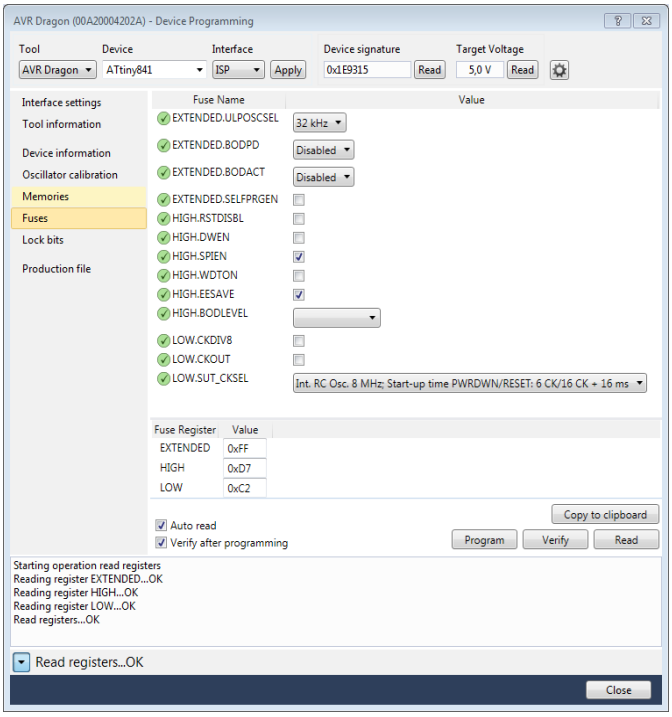
Une fois décompressé, il faut ajouter et modifier certains fichiers pour l'adapter à l'AVR Dragon et aux nouveaux modèles de microcontrôleurs. Pour ce faire, il faut copier tous les fichiers et dossiers se trouvant dans le dossier **`avr-calibrate/AVR053`** dans le dossier où vous avez extrait AVR053.zip, il faudra confirmer la fusion et les remplacements...

2. ÉTALONNAGE AVEC AVR STUDIO 6.2/7.0

- ✓ Ouvrir le fichier solution **`rc_calib.atsln`** avec AvrStudio
- ✓ Choisir votre modèle de microcontrôleur dans **`Project > Properties > Device`**

2.1. Réglage des fusibles du microcontrôleur

- **`Tools`** > **`Device Programming`** puis **`Tool >`** **`AVR Dragon`**, **`Device >`** votre modèle de MCU, **`Interface >`** votre interface de programmation, **`Apply`**.
- Onglet **`Fuses`** :
  - vérifier que les fusible **`SUT_CKSEL`** sont bien réglés pour **`Int. RC Osc. 8 Mhz`**,
  - vérifier que le fusible **`CKDIV8`** est décoché, sinon le décocher et programmer les fusibles. Il faudra sûrement passer l'horloge ISP à 125 kHz puis revenir à 1 MHz,
  - cocher le fusible EESAVE
- Cliquer sur **`Program`**.



2.2. Assemblage et programmation du *firmware* d'étalonnage

Vous trouverez un certain nombre de *firmware* « prêt à l'emploi » à la racine de avr-calibrate, si vous trouvez le *firmware* qui correspond à votre microcontrôleur et à votre programmateur, vous pouvez passer à l'étape suivante, sinon, il faut assembler votre *firmware* :

- ✓ Modifier le fichier **RC_Calibration.asm** afin de valider la ligne correspondant au microcontrôleur utilisé (**.include "Device specific\t841.asm"** par exemple) et au programmateur (**.include "Interface specific\isp_AVRDRAGON_interface.inc"** dans notre cas). Les autres lignes « **Device specific** » et « **Interface specific** » devront être invalidées en mettant un point virgule devant.
- ✓ Cliquer sur **Build > Build rc_calibrate** pour assembler le *firmware*
- ✓ **Tools > Device Programming > Memories**
- ✓ Dans le cadre **Flash** Cliquer sur **Program** (on vérifiera que le fichier **.hex** est bien notre **default\rc_calib.hex**)

Le microcontrôleur est prêt pour l'étalonnage !

2.3. Programmation d'un firmware prêt à l'emploi

Pour les étapes suivantes on utilise la ligne de commande ATMEL qui a été installée en même temps que AVR Studio (**AVR Studio 7.0 Command Prompt** par exemple)

Si vous trouvez le *firmware* qui correspond à votre microcontrôleur et à votre programmateur, vous pouvez effectuer la programmation, par exemple :

```
atprogram -t avrdragon -d attiny841 -i isp program -fl -c -f rc_calib_t841_dragon.hex
     Firmware check OK
     Oscillator calibration sequence succeeded.
```

2.4. Lecture des informations du microcontrôleur

Pour lire la valeur par défaut du registre OSCCAL :

```
atprogram -t avrdragon -i isp -d attiny841 info
```

OSCCAL est donné, par exemple, 0x21

2.5. Étalonnage

```
atprogram -t avrdragon -d attiny841 calibrate
     Firmware check OK
     Oscillator calibration sequence succeeded.
```

2.6. Lecture de la valeur en EEPROM

La valeur calculée de OSCCAL est stockée par le firmware en EEPROM à l'adresse 0x00 :

```
atprogram -t avrdragon -d attiny841 -i isp read -ee -s 1 --format hex
     Firmware check OK
     1c
```

La valeur de OSCCAL est 0x1c.

2.7. Copie de la valeur en haut de l'EEPROM

Dans le cas où le programme C ou C++ tournant sur le microcontrôleur utilise l'EEPROM, il est nécessaire de déplacer cet octet pour la mettre à la fin de l'EEPROM ou d'indiquer à l'éditeur de lien de ne rien écrire à l'adresse 0. En effet, par défaut, l'éditeur de lien du compilateur GCC range les variables en EEPROM en commençant par l'adresse 0. Si rien n'est fait, la première variable C ou C++ en EEPROM écrasera la valeur calculée lors de l'étalonnage.

Il est possible de copier la valeur calculée à une autre adresse de l'EEPROM (ou en FLASH !). Par exemple pour une utilisation avec AvrIO ou un microcontrôleur avec 256 octets d'EEPROM, on copiera la valeur à l'adresse 0x00FF :

```
atprogram -t avrdragon -d attiny841 -i isp write -ee -o 0xFF --values 1c
     Firmware check OK
     Write completed successfully.
```

3. UTILISATION DE LA VALEUR D'ÉTALONNAGE

Avec AvrIO

```
#include <avrio/osc.h>
...
int main() {
    iOscCalibrateFromEE (AVRIO_OSCCAL);
    ....
```

Avec le compilateur avr-gcc (sans AvrIO)

```
#include <avr/io.h>
#include <avr/eeprom.h>
...
int main() {
    OSCCAL = eeprom_read_byte ((const uint8_t *)0x00FF)
    ....
```

4. ÉTALONNAGE AVEC AVR STUDIO 4 (OBSOLÈTE)

Il est préférable d'utiliser la dernière version AVR Studio qui prend en charge les nouveaux microcontrôleurs, néanmoins, ce chapitre décrit la procédure pour les irréductibles qui veulent utiliser AVR Studio 4...

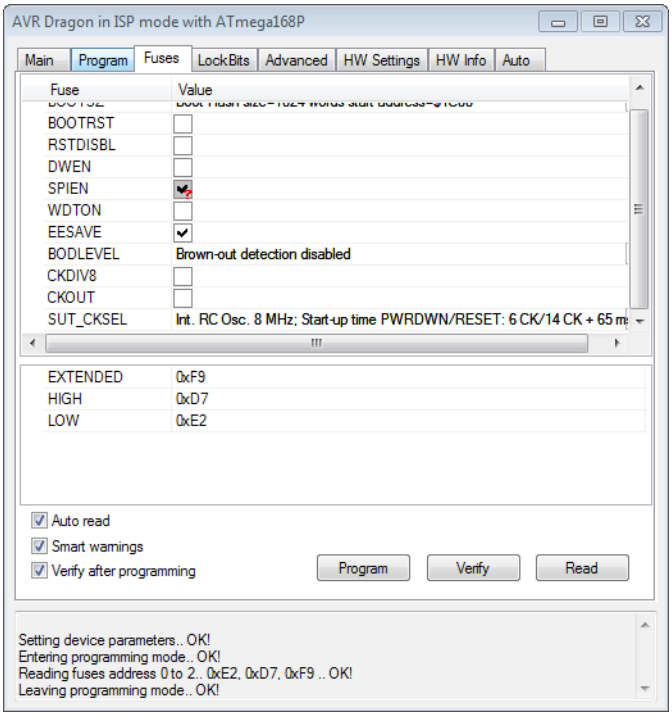
La procédure d'étalonnage utilise l'utilitaire **avrdragon.exe** fournit par ATMEL. Par défaut, il est installé dans **C:\Program Files (x86)\Atmel\AVR Tools\AvrDragon**. Il est préférable d'ajouter ce chemin dans le PATH de Windows.

Le programme **avrdragon.exe** de la version 4.19 Build 730 d'AVR Studio peut effectuer un étalonnage de l'oscillateur RC des modèles :

ATmega168A, ATmega168P, ATmega168PA, ATmega16HVA, ATmega48A, ATmega48PA, ATmega88A, ATmega88PA, ATmega8HVA, ATtiny13A, ATtiny2313A, ATtiny24A, ATtiny261A, ATtiny44A, ATtiny461A, ATtiny84A, ATtiny861A, AT90PWM1, AT90PWM2, AT90PWM216, AT90PWM2B, AT90PWM3, AT90PWM316, AT90PWM3B, AT90PWM81, AT90USB162, AT90USB82, ATA6289, ATmega168, ATmega168P, ATmega16HVA2, ATmega16HVB, ATmega16M1, ATmega16U2, ATmega328, ATmega328P, ATmega32C1, ATmega32HVB, ATmega32M1, ATmega32U2, ATmega48, ATmega48P, ATmega64C1, ATmega64HVE, ATmega64M1, ATmega88, ATmega88P, ATmega8U2, ATtiny13, ATtiny167, ATtiny2313, ATtiny24, ATtiny25, ATtiny261, ATtiny4313, ATtiny43U, ATtiny44, ATtiny45, ATtiny461, ATtiny48, ATtiny84, ATtiny85, ATtiny861, ATtiny87, ATtiny88 .

4.1. Réglage des fusibles du microcontrôleur

- **Tools** > **Program AVR** > **Auto connect**
- Onglet **Fuses** :
  - vérifier que les fusible **SUT_CKSEL** sont bien réglés pour **Int. RC Osc. 8 Mhz**,
  - vérifier que le fusible **CKDIV8** est décoché, sinon le décocher et programmer les fusibles. Il faudra sûrement passer l'horloge ISP à 125 kHz puis revenir à 1 MHz,
  - cocher le fusible EESAVE
- Cliquer sur **Program**.



4.2. Assemblage et programmation du *firmware* d'étalonnage

- ✓ Ouvrir le projet **rc_calib.aps** avec AvrStudio 4
- ✓ Modifier le fichier **RC_Calibration.asm** afin de valider la ligne correspondant au microcontrôleur utilisé (**.include "Device specific\m168p.asm"** par exemple) et au programmateur (**.include "Interface specific\isp_AVRDRAGON_interface.inc"** dans notre cas). Les autres lignes « **Device specific** » et « **Interface specific** » devront être invalidées en mettant un point virgule devant.
- ✓ Cliquer sur **Build** pour assembler le *firmware*
- ✓ **Tools** > **Program AVR** > **Auto connect**
- ✓ Onglet **Program**, indiquer le chemin du fichier **.hex** assemblé à l'étape précédente dans **Input Hex File**

- ✓ Cliquer sur **Program**.

Le microcontrôleur est prêt pour l'étalonnage !

4.3. Étalonnage

Ouvrir une invite de commande DOS :

```
C:\Users\Pascal>avrdragon -d ATmega168P -Y
avrdragon.exe v1.0.3 Copyright (C) Atmel Corporation 2006-2011
Calibration completed successfully.
```

La valeur calculée de OSCCAL est stockée en EEPROM à l'adresse 0, pour la lire :

```
C:\Users\Pascal>avrdragon -d ATmega168P -Z 0
avrdragon.exe v1.0.3 Copyright (C) Atmel Corporation 2006-2011
Value of EEPROM address 0x0000 is 0x6c.
```

Dans le cas où le programme C ou C++ tournant sur le microcontrôleur utilise l'EEPROM, il est nécessaire de déplacer cet octet pour la mettre à la fin de l'EEPROM ou d'indiquer à l'éditeur de lien de ne rien écrire à l'adresse 0. En effet, par défaut, l'éditeur de lien du compilateur GCC range les variables en EEPROM en commençant par l'adresse 0. Si rien n'est fait, la première variable C ou C++ en EEPROM écrasera la valeur calculée lors de l'étalonnage.

Il est possible de copier la valeur calculée à une autre adresse de l'EEPROM (ou en FLASH !). Par exemple pour une utilisation avec AvrIO ou un microcontrôleur avec 256 octets d'EEPROM, on copiera la valeur à l'adresse 0x00FF :

```
C:\Users\Pascal>avrdragon -d ATmega168P -Z 0 -Se 0x00FF
avrdragon.exe v1.0.3 Copyright (C) Atmel Corporation 2006-2011
Value of EEPROM address 0x0000 is 0x6c.
Wrote OSCCAL value 0x6c to EEPROM address 0x00ff.
OSCCAL value in EEPROM address 0x00ff verified.
```

4.4. Upgrade du firmware AVR Dragon

Il sera peut être nécessaire de mettre à jour le firmware de l'AVR Dragon pour utiliser **atprogram** en version 6.2. Il faudra faire un downgrade si vous utilisez AVR Studio 4 !

Lancer la ligne de commande AVR Studio 6.2 et aller dans le dossier AVRDragon de tools :

```
cd .....\tools\AVRDragon
atfw -a dragon_fw.zip -t avrdragon
```

Sous Virtualbox, il faudra valider le périphérique USB AVRBLDR je crois...