

- PROJET RTEL4I -



**Conception détaillée du portage de PREEMPT-RT sur
processeur NIOS2 (L2.3-b)**



SOMMAIRE

1. Présentation du processeur softcore NIOS II d'Altera.....	5
2. Portage de PREEMPT-RT sur le processeur softcore NIOS II.....	9
2.1. Principes du portage.....	9
2.2. Fichiers sources Linux impactés par le portage PREEMPT-RT.....	9
3. Mise en œuvre de PREEMPT-RT sur le processeur softcore NIOS II.....	10
3.1. Carte cible.....	10
3.2. Configuration matérielle : construction du système SoPC.....	11
3.3. Installation des outils Altera sous Linux.....	24
3.4. Configuration logicielle : configuration et compilation de Linux pour NIOS II.....	25
3.5. Configuration logicielle : configuration et compilation de PREEMPT-RT pour NIOS II.....	29
4. Mesures des performances.....	32
4.1. Processus ordinaire.....	32
4.2. Processus Temps Réel.....	33
5. Limitations du portage.....	35
6. Conclusion.....	36
7. Références documentaires.....	37
8. Références logicielles.....	37
9. Annexe 1 : fichiers impactés par le portage de PREEMPT-RT pour l'architecture nios2.....	38
9.1. Liste des fichiers génériques modifiés.....	38
9.2. Liste des fichiers spécifiques modifiés pour l'architecture nios2.....	46
10. Annexe 2 : fichier source squaretest-rt de génération d'un signal périodique.....	48
11. Annexe 3 : Pilote de périphérique pour accéder aux leds de la carte cible.....	50

Fiche synthétique

Matériel	
Processeur cible	NIOS II avec MMU
Fondeur	Altera
Outils de synthèse	Quartus II version 9.1 et supérieur
Carte cible	Altera Stratix 1S10
Logiciel	
Distribution Linux	Fedora 12
Version Linux pour NIOS II	Linux 2.6.31
Version compilateur croisé pour NIOS II	4.1.2
Version patch pour NIOS II	preempt-rt-2.6.31.12-rt21-nios2-1.0-00.patch
Version patch PREEMPT-RT pour Linux	2.6.31.12-rt21

Gestion du document :

Date	Version	Modifications	Relecture	Commentaires
15/05/09	1.0	PK	PK	Création fichier

1. PRÉSENTATION DU PROCESSEUR SOFTCORE NIOS II D'ALTERA

Le processeur NIOS II (2^{ème} génération du processeur NIOS) est un processeur RISC *softcore* entièrement synchrone, son architecture interne étant de type Harvard. Il possède au maximum 6 niveaux de pipeline, cadencé à quelques dizaines de MHz, avec une largeur de bus de 32 bits. Ses performances sont de 30 à 80 MIPS (*Million Instructions per Second*).

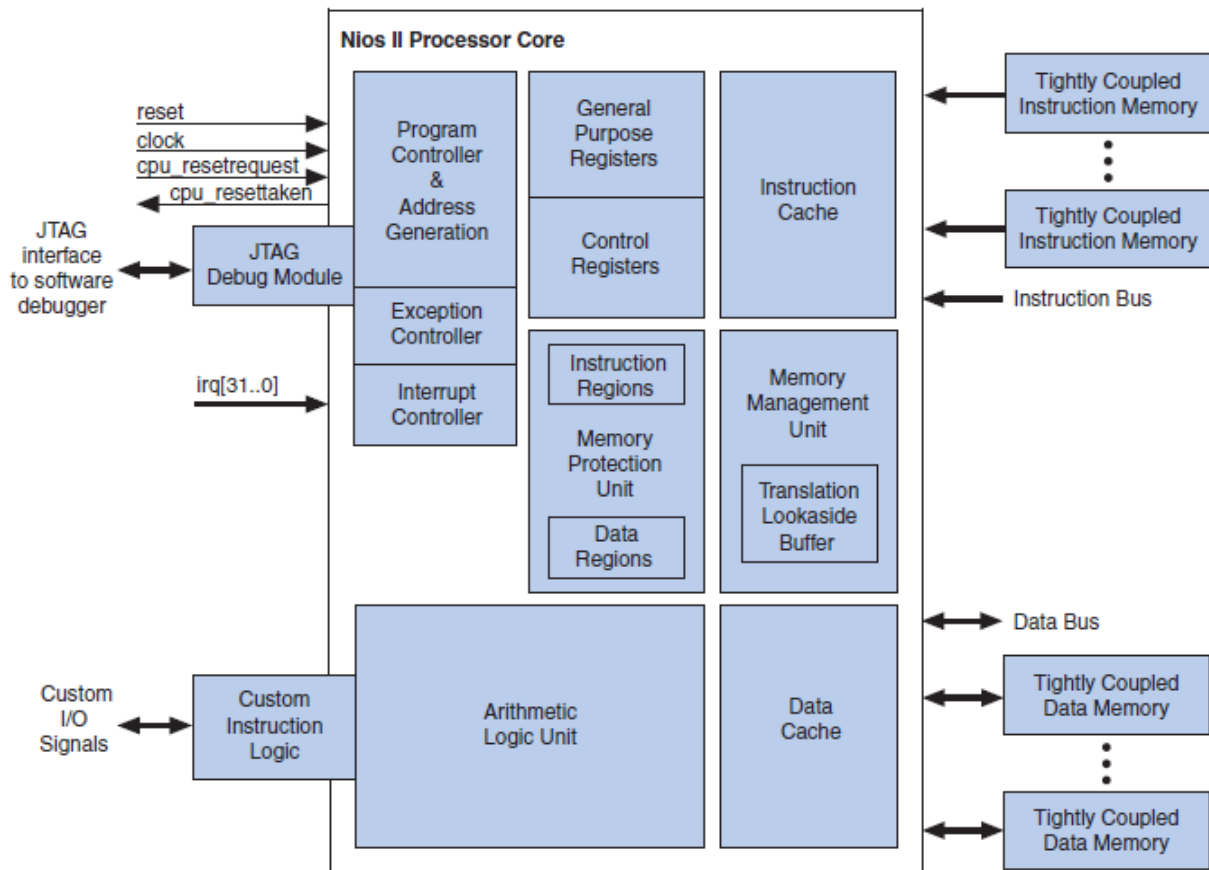


Figure 1 : Architecture du processeur NIOS II

Il est possible d'accélérer certains traitements en ajoutant des instructions personnelles ou *Custom Instructions* (décrites en VHDL) au processeur NIOS II. De cette manière, il est possible de réaliser la surcharge d'opérateurs ou simplement d'étendre le jeu d'instructions.

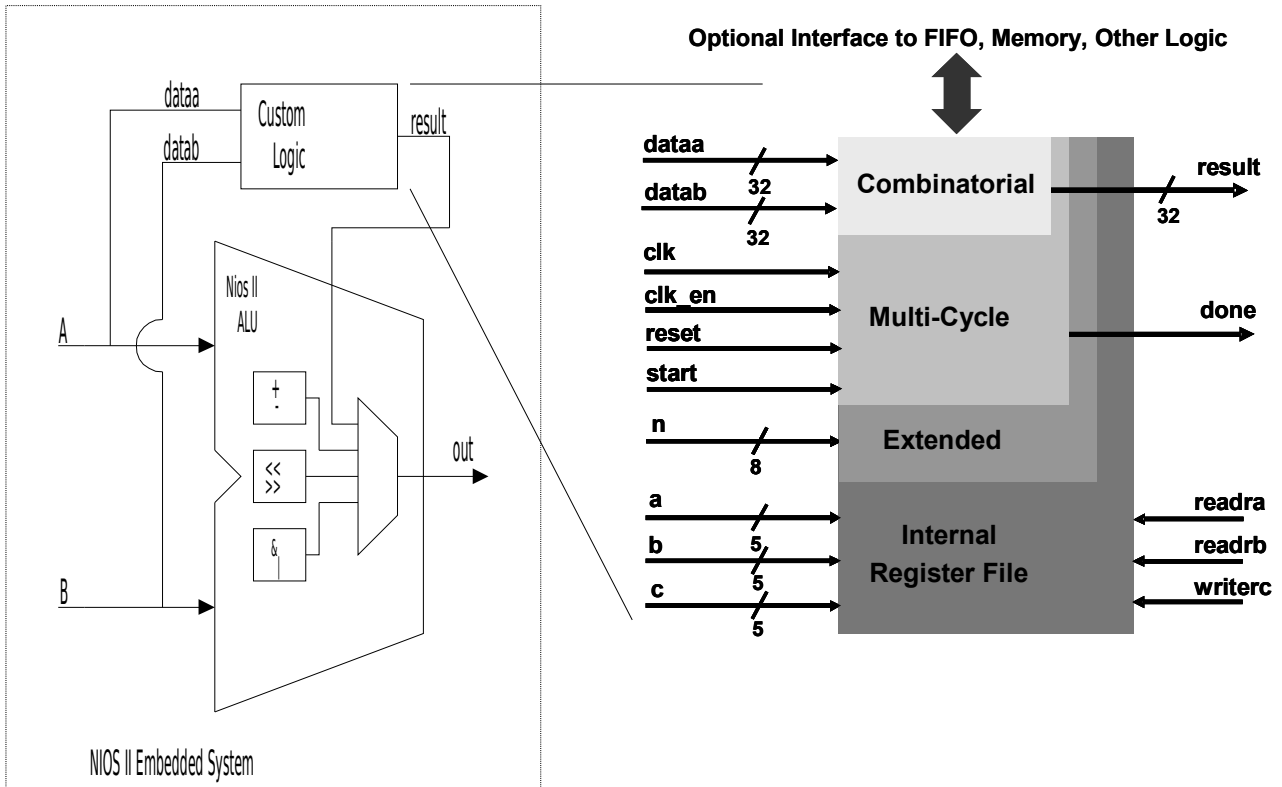


Figure 2 : Instruction personnalisée avec le processeur NIOS II

La table suivante résume les caractéristiques du processeur NIOS II.

NIOS (obsolète)	NIOS II
<ul style="list-style-type: none"> ■ Architecture RISC pipeline ■ Instructions 16 bits ■ Liste de registres avec fenêtrage ■ Données sur 16 ou 32 bits ■ 64 niveaux d'interruption ■ Cache d'instructions et de données ■ Instructions personnalisées 	<ul style="list-style-type: none"> ■ Architecture RISC pipeline ■ Instructions 32 bits ■ Liste de registres fixe ■ Données sur 32 bits ■ 32 niveaux d'interruption ■ Cache d'instructions et de données ■ Instructions personnalisées

Figure 3 : Caractéristiques du processeur NIOS II

Lors de la configuration du processeur NIOS II avec l'outil *SoPC Builder*, il est possible de choisir entre trois versions du processeur NIOS II : une première version *Economy* qui utilise moins de surface de silicium du composant FPGA, une deuxième version *Standard* qui permet un compromis

entre surface et rapidité et enfin une dernière version *Fast* qui est la plus rapide des deux autres versions.

	NIOS II /f	NIOS II /s	NIOS II /e
Pipeline	6 niveaux	5 niveaux	Non
Multiplication Matériel	1 Cycle	3 Cycle	Par logiciel
Branch Prediction	Dynamic	Static	Non
Cache d'Instructions	Configurable	Configurable	Non
Cache de données	Configurable	Non	Non
Instructions Personnalisés	Supérieur à 256		

Figure 4 : Les trois versions du processeur NIOS II

La figure suivante présente les performances en DMIPS (*Dhrystons Million Instructions per Second*) et la surface occupée en éléments logiques Altera (*Logic Element*) des différentes versions du processeur NIOS II sur différentes familles de composants FPGA d'Altera (Stratix, Stratix II, Cyclone, Cyclone II...).

Fast

Standard

Economy

Cyclone II

Figure 5 : Performances du processeur NIOS II sur différents circuits FPGA d'Altera

Dans la phase de construction du circuit SoPC (*System on Programmable Chip*) avec l'outil *SoPC Builder*, il est possible d'inclure différents périphériques standards en utilisant le bus d'adresses/données Altera *Avalon* :

- Mémoire.
- Timer.
- Liaison série UART.
- Interface écran LCD.
- E/S parallèles.
- Interfaces Ethernet.
- Interface Compact Flash.
- JTAG.
- ...

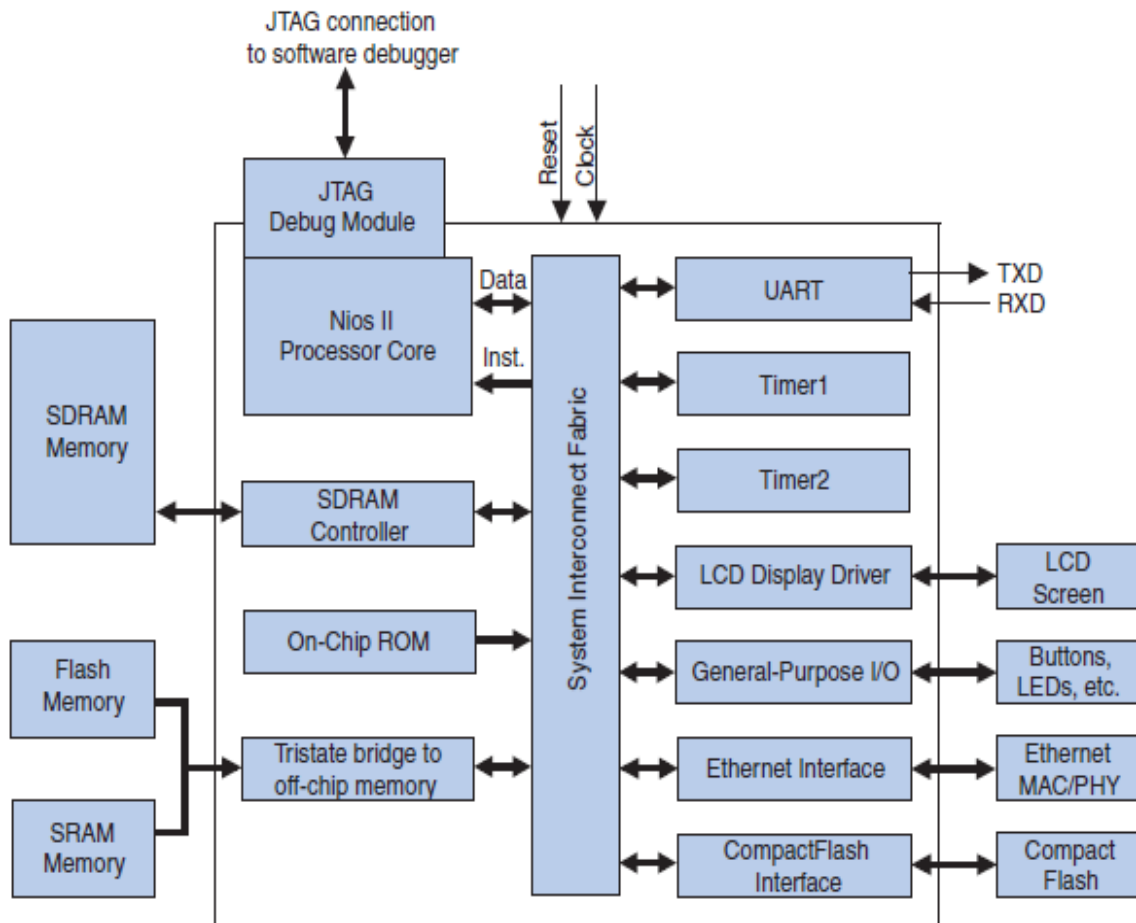


Figure 6 : Bus Avalon pour l'interfaçage des périphériques au processeur NIOS II

2. PORTAGE DE PREEMPT-RT SUR LE PROCESSEUR SOFTCORE NIOS II

2.1. Principes du portage

Le portage de PREEMPT-RT pour le processeur NIOS II avec MMU s'est basé sur celui pour le processeur ARM (PREEMPT-RT version 2.6.31.12) et pour le processeur SH (PREEMPT-RT version 2.6.22.1).

Une analyse fine a pu mettre en évidence le rôle des fichiers potentiellement à modifier pour l'architecture NIOS II dans le répertoire *arch/nios2*.

Les principaux fichiers modifiés sont décrits.

Le fichier *arch/nios2/kernel/entry.S* est modifié pour appeler la fonction du nouvel ordonnanceur PREEMPT-RT *__schedule*.

Le fichier *arch/nios2/kernel/irq.c* est modifié pour utiliser des spinlocks atomiques.

Le fichier *arch/nios2/kernel/process.c* est modifié pour appeler le nouvel ordonnanceur PREEMPT-RT *__schedule()*.

Le fichier *arch/nios2/kernel/signal.c* est modifié pour avoir un noyau préemptible.

Le fichier *arch/nios2/kernel/time.c* est modifié pour une gestion atomique du temps.

Le fichier *arch/nios2/mm/cacheflush.c* est modifié pour avoir un flush du cache MMU atomique.

Le fichier *arch/nios2/mm/mmu_context.c* est modifié pour avoir un changement de contexte atomique par spinlock atomique.

Le fichier *drivers/serial/altjuart.c* est modifié pour un accès atomique par spinlock atomique.

Le fichier *drivers/serial/altuart.c* est modifié pour avoir un accès atomique par spinlock atomique.

Le fichier *include/asm-generic/bitops/atomic.h* est modifié pour utiliser plutôt *raw_local_irq_xxx()* que *local_irq_xxx()* pour homogénéité avec le patch général PREEMPT-RT. Il ne semble pas avoir de différences pour le moment entre les fonctions *raw_local_irq_xxx()* et *local_irq_xxx()*.

2.2. Fichiers sources Linux impactés par le portage PREEMPT-RT

La liste des fichiers sources Linux impactés par le portage de PREEMPT-RT sur l'architecture NIOS II est donnée en annexe 1.

On retrouve globalement un ensemble de fichiers indépendants de l'architecture pour l'implantation de la préemption Temps Réel mou dans le noyau Linux et un ensemble de fichiers modifiés concernant l'architecture NIOS II...

3. MISE EN ŒUVRE DE PREEMPT-RT SUR LE PROCESSEUR SOFTCORE NIOS II

3.1. Carte cible

La carte choisie pour le portage de PREEMPT-RT sur le processeur NIOS II est une carte d'évaluation moyenne gamme d'Altera : carte Stratix 1S10.



Figure 7 : Carte cible Altera Stratix 1S10

La carte Stratix 1S10 possède les caractéristiques suivantes :

- Circuit FPGA Stratix II EP2S30F672C5.
- 1 Mo de SRAM 16 bits.
- 16 Mo de SDRAM 32 bits.
- 16 Mo de mémoire Flash.
- 1 support CompactFlash type I.
- 1 interface Ethernet 10/100 Mb/s.
- 2 ports série (RS-232 DB9).
- 2 ports d'extension pour carte fille maison.
- 1 connecteur JTAG.
- 4 boutons poussoirs.
- 8 leds utilisateurs.
- 2 afficheurs 7 segments.

- 1 afficheur LCD 2x16 caractères.

3.2. Configuration matérielle : construction du système SoPC

Il convient à l'aide de l'outil de synthèse *Quartus II* d'Altera de construire le système SoPC compatible avec le portage PREEMPT-RT sur processeur NIOS II. A l'issue de la synthèse, on récupèrera le fichier *.sof* de programmation du circuit FPGA de la carte cible Stratix 1S10.

1. Design de référence

Le design de référence *Quartus II* pour Linux embarqué avec ou sans l'extension Temps Réel a été construit à partir du design *standard* fourni avec la carte Stratix 1S10 par Altera.

Ce design *standard* est fourni pour toutes les cartes Altera et est une bonne base de travail pour faire tourner dans un premier temps Linux embarqué.

La figure présente le design de référence *standard* :

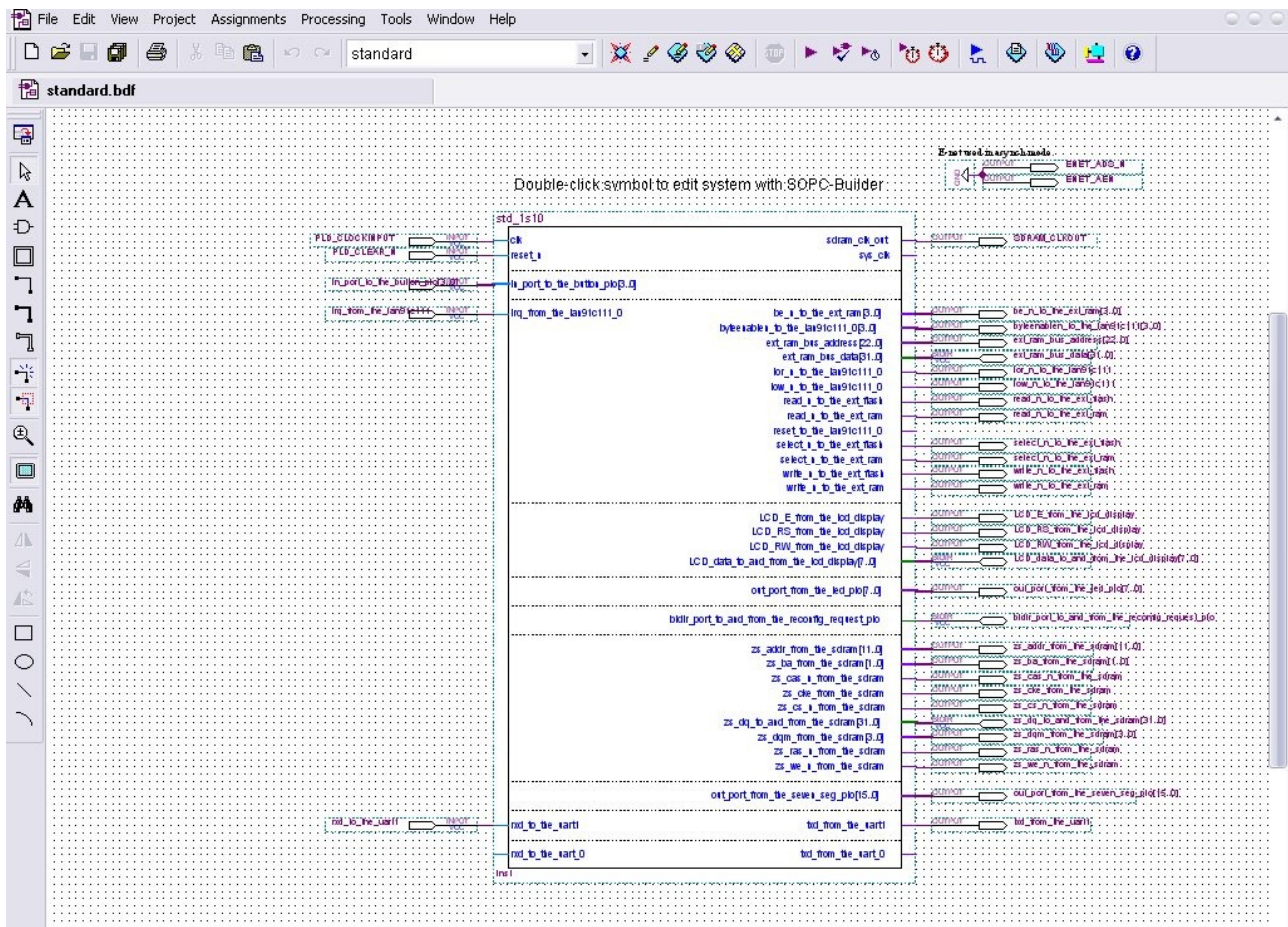


Figure 8 : Design de référence *standard* *Quartus II* de la carte cible Altera Stratix 1S10

Une fois le design de référence adopté, il convient de modifier le système SoPC en utilisant l'outil *SoPC Builder* d'Altera.

Les 3 figures suivantes montrent la configuration du système SoPC.

Pour pouvoir intégrer facilement le patch PREEMPT-RT, le processeur NIOS est configuré avec une MMU (*Memory Management Unit*) car le patch est fait originellement pour des processeurs avec MMU.

Altera a intégré une MMU dans son processeur *softcore* NIOS II en juin 2008. Un portage de Linux standard est disponible depuis septembre 2009. C'est sur cette base qu'a été fait ce travail...

The screenshot displays the Altera SOPC Builder interface. The main window shows the 'System Generation' tab for a project named 'std_1s10.sopc'. The 'Device Family' is set to 'Stratix'. The 'Clock Settings' table is as follows:

Name	Source	MHz
clk	External	50,0
sys_clk	pll.c0	50,0
edram_clk_out	pll.c0	50,0

The main component table is as follows:

Use	Connections	Module Name	Description	Clock	Base	End
<input checked="" type="checkbox"/>		cpu	Nios II Processor			
<input checked="" type="checkbox"/>		instruction_master	Avalon Memory Mapped Master	sys_clk		
<input checked="" type="checkbox"/>		tightly_coupled_instru...	Avalon Memory Mapped Master			
<input checked="" type="checkbox"/>		data_master	Avalon Memory Mapped Master			
<input checked="" type="checkbox"/>		tightly_coupled_data_...	Avalon Memory Mapped Master			
<input checked="" type="checkbox"/>		itag_debug_module	Avalon Memory Mapped Slave			
<input checked="" type="checkbox"/>		fast_tlb_miss_ram_1k	On-Chip Memory (RAM or ROM)			
<input checked="" type="checkbox"/>		s1	Avalon Memory Mapped Slave	sys_clk	0x00810000	0x008107ff
<input checked="" type="checkbox"/>		s2	Avalon Memory Mapped Slave	sys_clk	0x02100000	0x021001ff
<input checked="" type="checkbox"/>		ext_ram_bus	Avalon-MM Tristate Bridge	sys_clk		
<input checked="" type="checkbox"/>		avalon_slave	Avalon Memory Mapped Slave	sys_clk		
<input checked="" type="checkbox"/>		tristate_master	Avalon Memory Mapped Tristate Master			
<input checked="" type="checkbox"/>		ext_flash	Flash Memory Interface (CFI)			
<input checked="" type="checkbox"/>		s1	Avalon Memory Mapped Tristate Slave	sys_clk	0x00000000	0x007fffff
<input checked="" type="checkbox"/>		ext_ram	IDT71V416 SRAM	sys_clk		
<input checked="" type="checkbox"/>		s1	Avalon Memory Mapped Tristate Slave	sys_clk	0x02000000	0x020fffff

At the bottom of the window, there are status messages:

- Info: **ext_flash**: Flash memory capacity: 8,0 MBytes (8388608 bytes).
- Info: **reconfig_request_pio**: PIO inputs are not hardwired in test bench. Undefined values will be read from PIO inputs during simulation.

Altera SOPC Builder - std_1s10.sopc* (D:\tmp\design_preempt-rt_v4\design_preempt-rt_v4\design\std_1s10.sopc)

File Edit Module System View Tools Nios II Help

System Contents System Generation

Component Library

Project
New component...

Library

- Avalon Verification Suite
- Bridges and Adapters
- Interface Protocols
- Legacy Components
- Memories and Memory Control
- Peripherals
 - Debug and Performance
 - Display
 - FPGA Peripherals
 - Microcontroller Peripherals
 - Interval Timer
 - PIO (Parallel I/O)
 - Multiprocessor Coordination
- PLL
- Processor Additions
- Processors
- SLS
- Video and Image Processing

Target
Device Family: Stratix

Clock Settings

Name	Source	MHz
clk	External	50,0
sys_clk	pll.c0	50,0
extram_clk_out	pll.c0	50,0

Remove Add

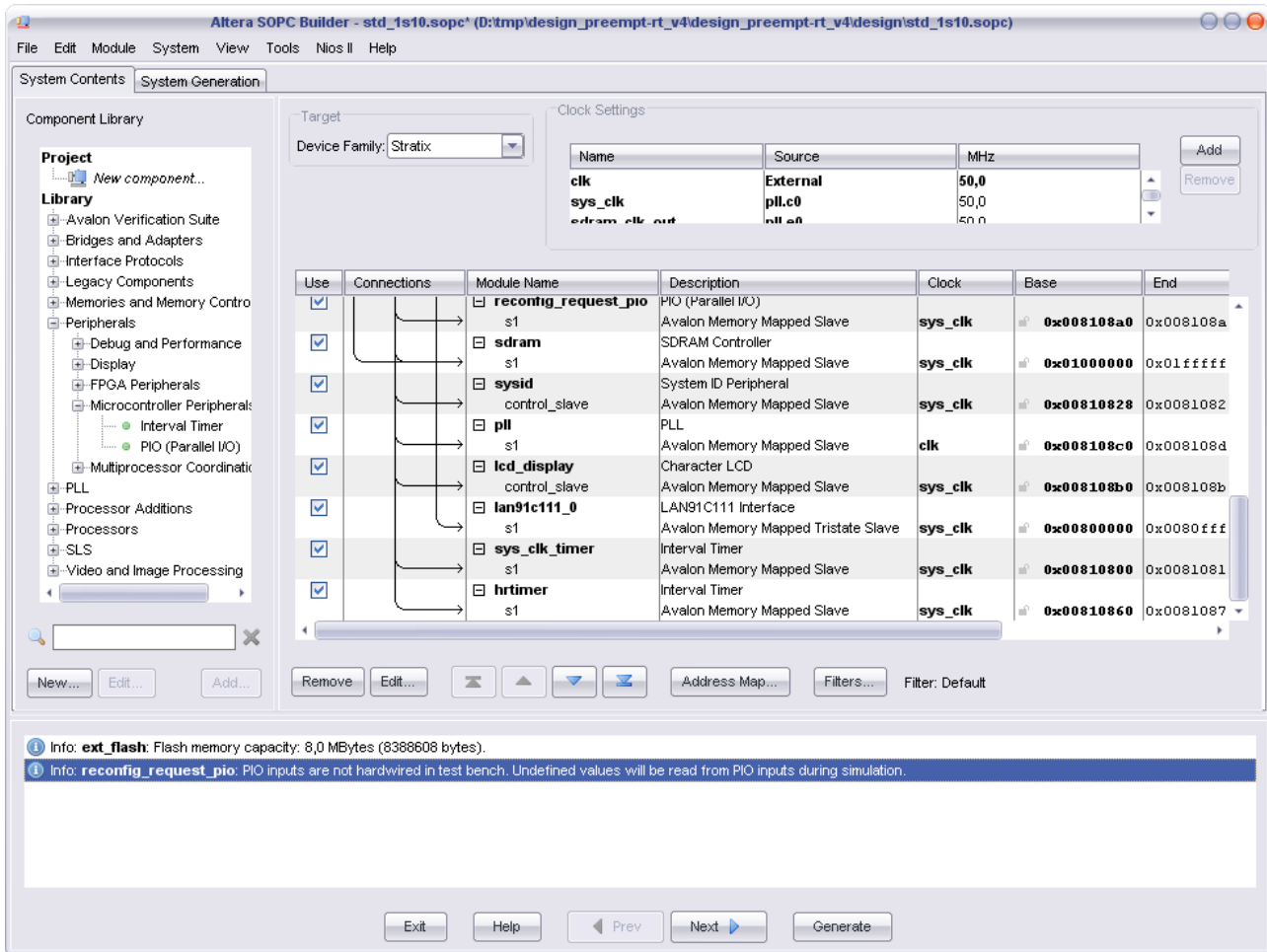
Use	Connections	Module Name	Description	Clock	Base	End
<input checked="" type="checkbox"/>		ext_flash s1	Flash Memory Interface (CFI) Avalon Memory Mapped Tristate Slave	sys_clk	0x00000000	0x007fffff
<input checked="" type="checkbox"/>		ext_ram s1	IDT71V416 SRAM Avalon Memory Mapped Tristate Slave	sys_clk	0x02000000	0x020fffff
<input checked="" type="checkbox"/>		jtag_uart avalon_jtag_slave	JTAG UART Avalon Memory Mapped Slave	sys_clk	0x00810820	0x0081082f
<input checked="" type="checkbox"/>		uart0 s1	UART (RS-232 Serial Port) Avalon Memory Mapped Slave	sys_clk	0x00810840	0x0081085f
<input checked="" type="checkbox"/>		button_pio s1	PIO (Parallel I/O) Avalon Memory Mapped Slave	sys_clk	0x00810830	0x0081083f
<input checked="" type="checkbox"/>		led_pio s1	PIO (Parallel I/O) Avalon Memory Mapped Slave	sys_clk	0x00810880	0x0081088f
<input checked="" type="checkbox"/>		seven_seg_pio s1	PIO (Parallel I/O) Avalon Memory Mapped Slave	sys_clk	0x00810890	0x0081089f
<input checked="" type="checkbox"/>		reconfig_request_pio s1	PIO (Parallel I/O) Avalon Memory Mapped Slave	sys_clk	0x008108a0	0x008108af

Remove Edit... Address Map... Filters... Filter: Default

Info: ext_flash: Flash memory capacity: 8,0 MBytes (8388608 bytes).

Info: reconfig_request_pio: PIO inputs are not hardwired in test bench. Undefined values will be read from PIO inputs during simulation.

Exit Help Prev Next Generate



Figures 9 : Création des différents périphériques dans le système SoPC

Les points importants concernent l'attribution des IRQs et des plages d'adresses des registres des différents périphériques intégrés dans le système SoPC.

Il est important de respecter ces règles pour que le portage de PREEMPT-RT sur processeur NIOS II marche. Tous les numéros d'IRQs doivent être différents de 0 (autodétection au sens Linux sinon).

2. Mapping mémoire

Le mapping mémoire ne doit pas comporter de recouvrement dans l'attribution des plages d'adresses pour les registres des différents périphériques. Il n'y a pas de contraintes sinon, hormis les contraintes des différents types de mémoire de la carte cible.

	cpu.instruction_master	cpu.tightly_coupled_instruction...	cpu.data_master
pll.s1			0x008108c0 - 0x008108c0
cpu.jtag_debug_module	0x00810000 - 0x008107ff		0x00810000 - 0x008107ff
ext_ram_bus.avalon_slave			
ext_flash.s1	0x00000000 - 0x007fffff		0x00000000 - 0x007fffff
ext_ram.s1	0x02000000 - 0x020fffff		0x02000000 - 0x020fffff
fast_tlb_miss_ram_1k.s1		0x02100000 - 0x021001ff	
fast_tlb_miss_ram_1k.s2			
sys_clk_timer.s1			0x00810800 - 0x00810800
jtag_uart.avalon_jtag_slave			0x00810820 - 0x00810820
uart0.s1			0x00810840 - 0x00810840
button_pio.s1			0x00810830 - 0x00810830
led_pio.s1			0x00810880 - 0x00810880
seven_seg_pio.s1			0x00810890 - 0x00810890
reconfig_request_pio.s1			0x008108a0 - 0x008108a0
sdram.s1	0x01000000 - 0x01ffffff		0x01000000 - 0x01ffffff
sysid.control_slave			0x00810828 - 0x00810828
lcd_display_control_slave			0x008108b0 - 0x008108b0
lan91c111_0.s1	0x00800000 - 0x0080ffff		0x00800000 - 0x0080ffff
hrtimer.s1			0x00810860 - 0x00810860

Figure 10 : Mapping mémoire du système SoPC

3. Configuration du processeur NIOS II

Le processeur NIOS II est configuré au minimum dans sa version f (*fast*).

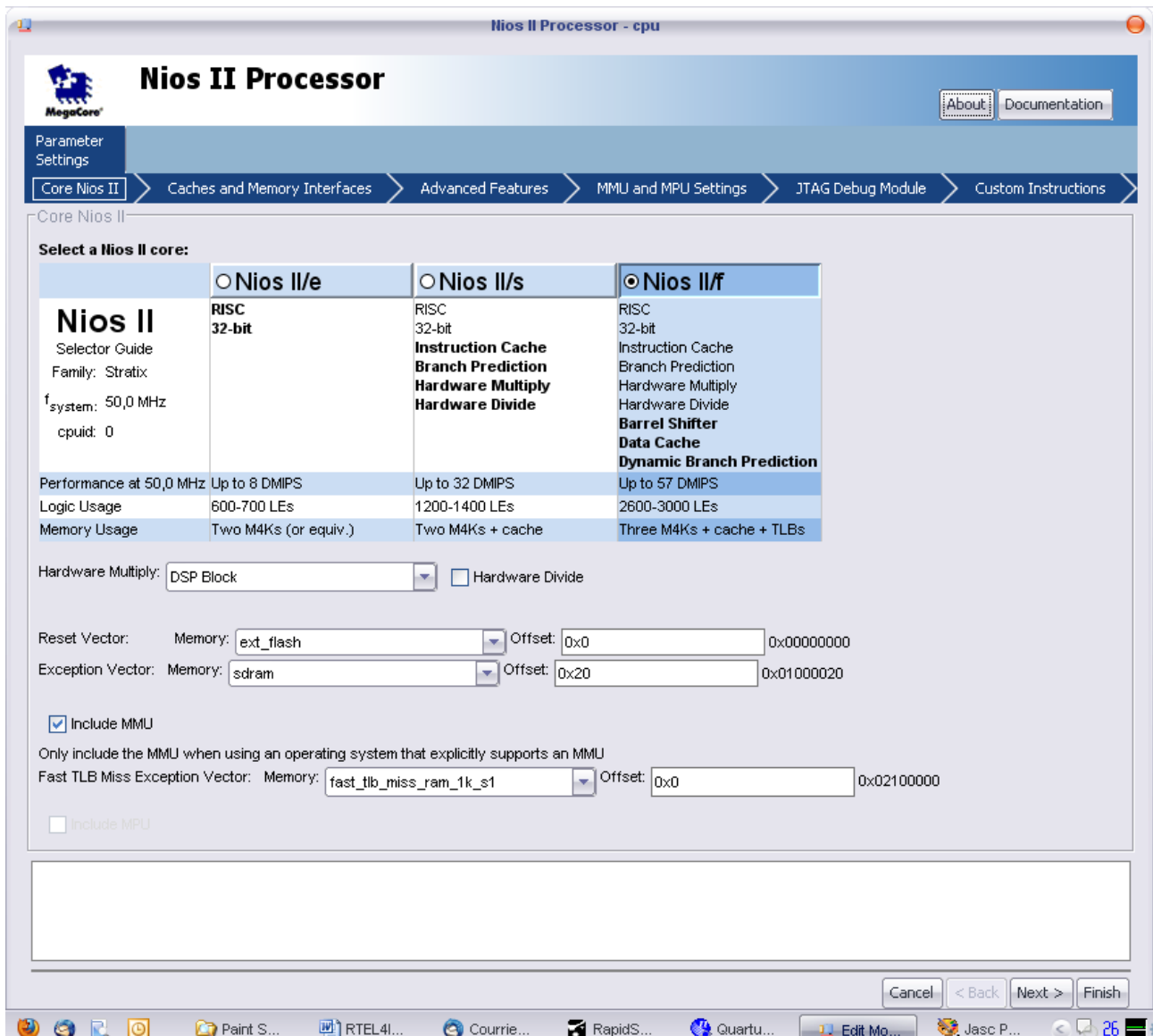


Figure 11 : Configuration du processeur NIOS II dans sa version f

Les caches d'instructions et de données sont validés :

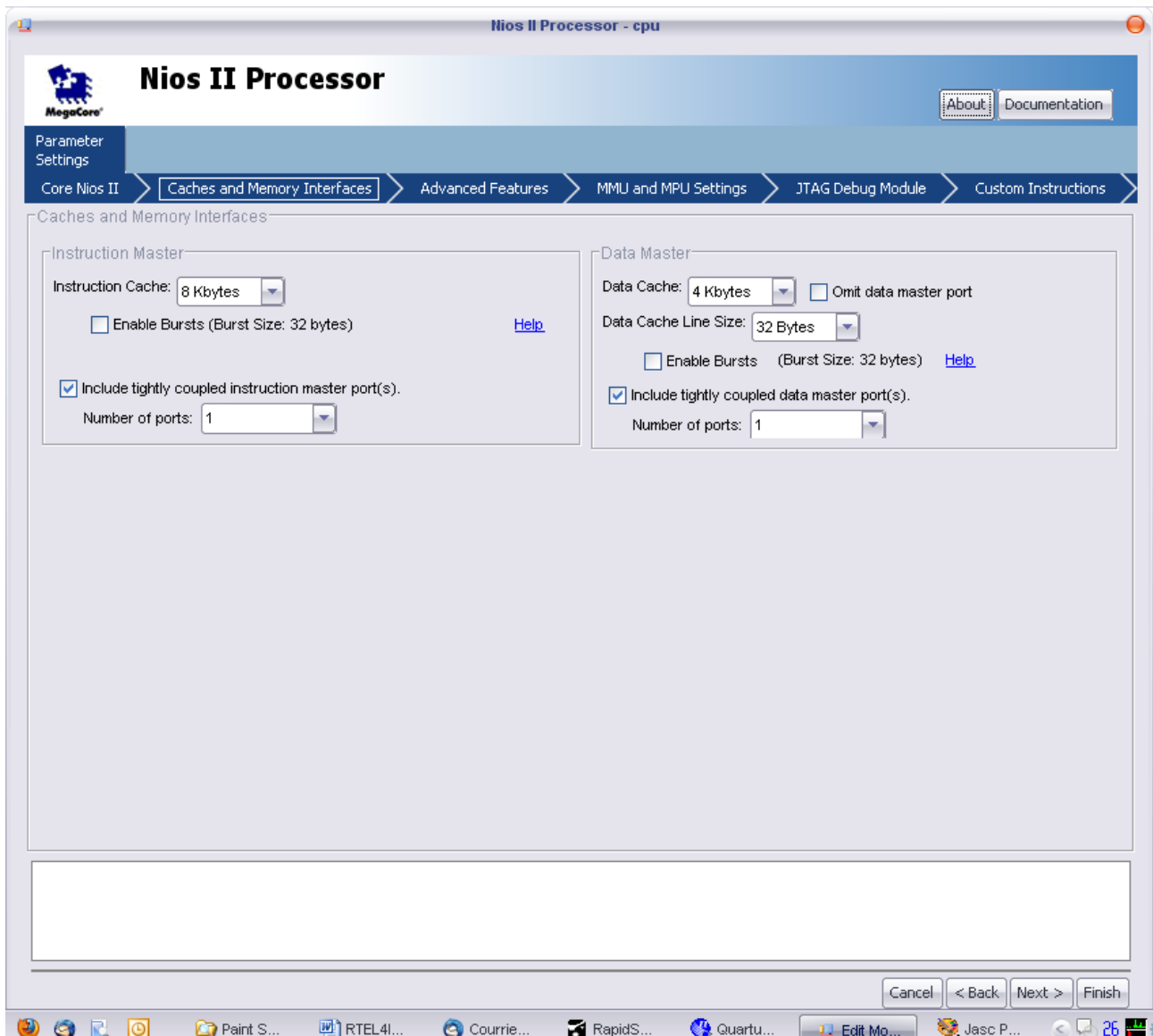
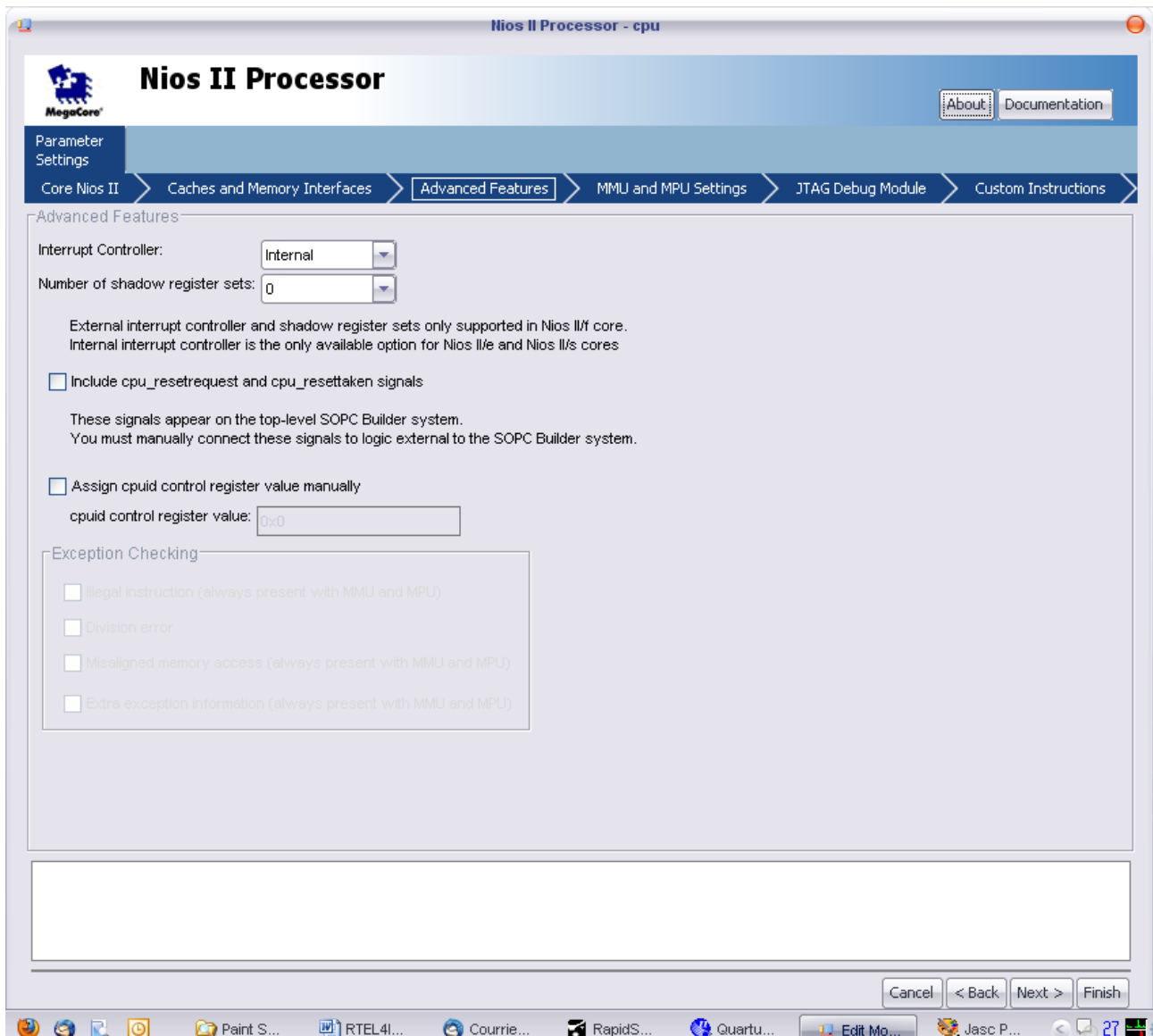
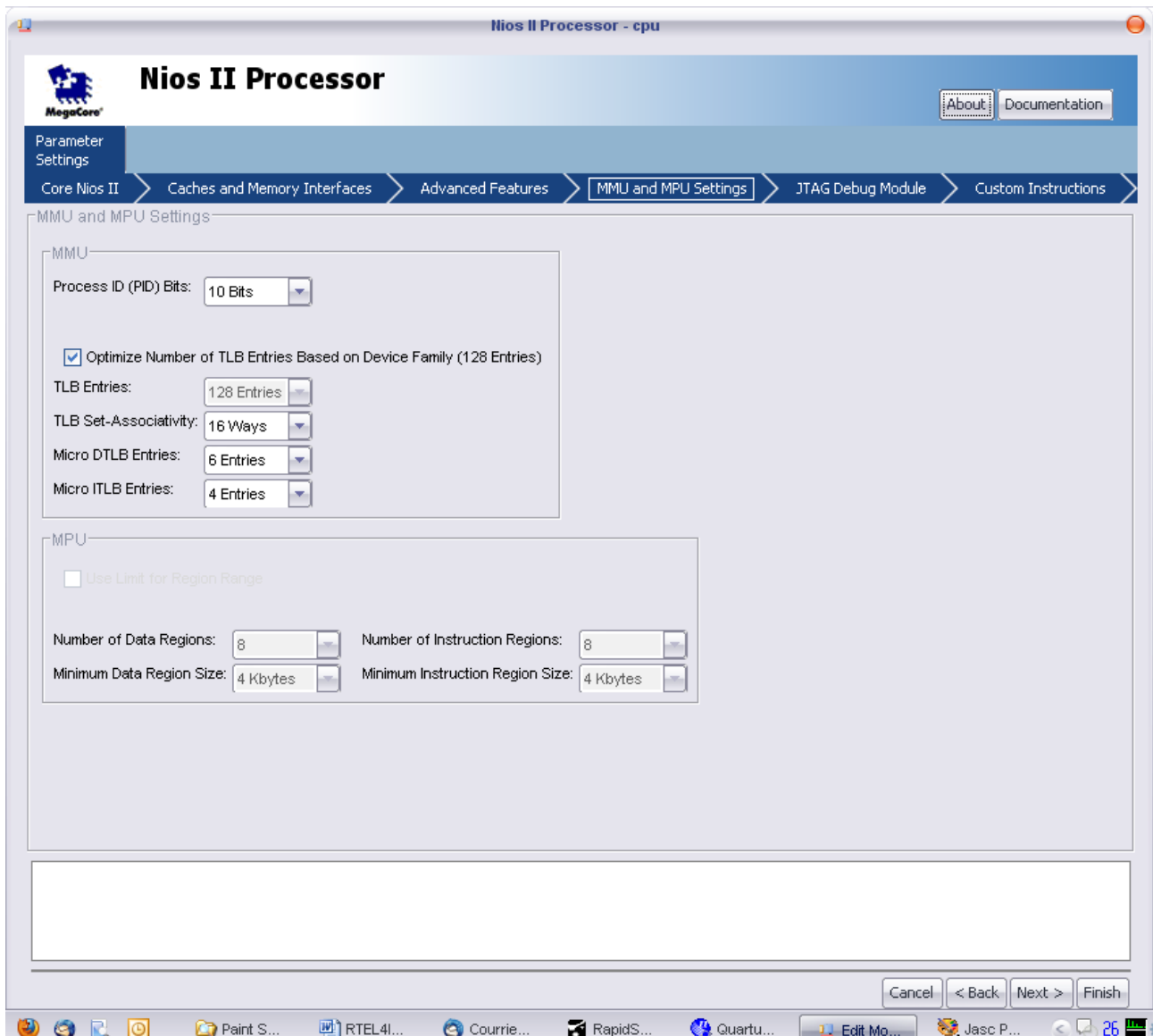


Figure 12 : Validation des caches d'instructions et de données du processeur NIOS II

Le circuit MMU est validé.

En conséquence, **on utilisera la version Linux standard** pour la version Linux embarqué pour le processeur NIOS II...





Figures 13 : Circuit MMU pour le processeur NIOS II

On validera le JTAG en version 1 pour le debug :

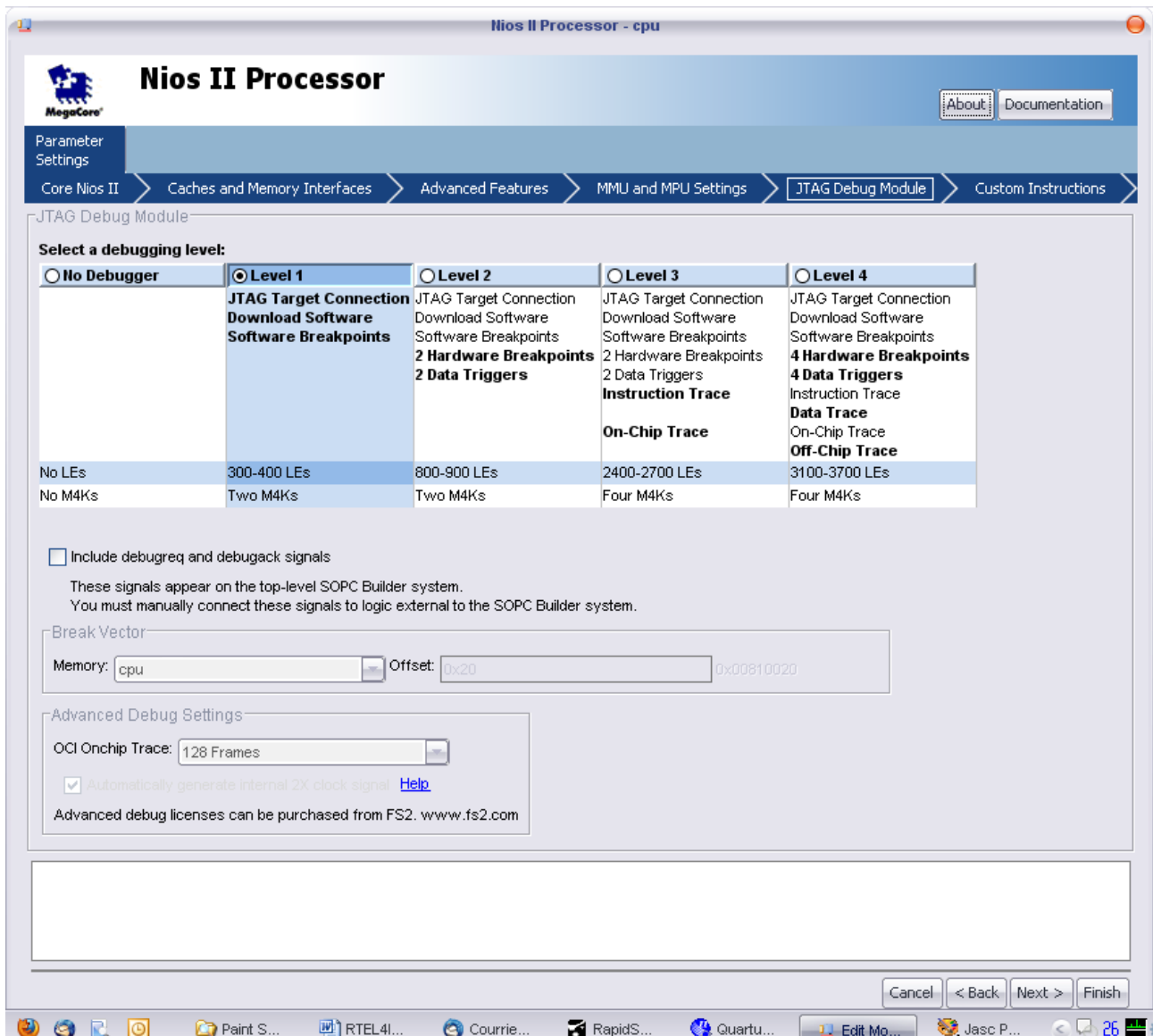


Figure 14 : Validation du JTAG (level 1)

On ne valide rien au niveau des *Custom Instructions* :

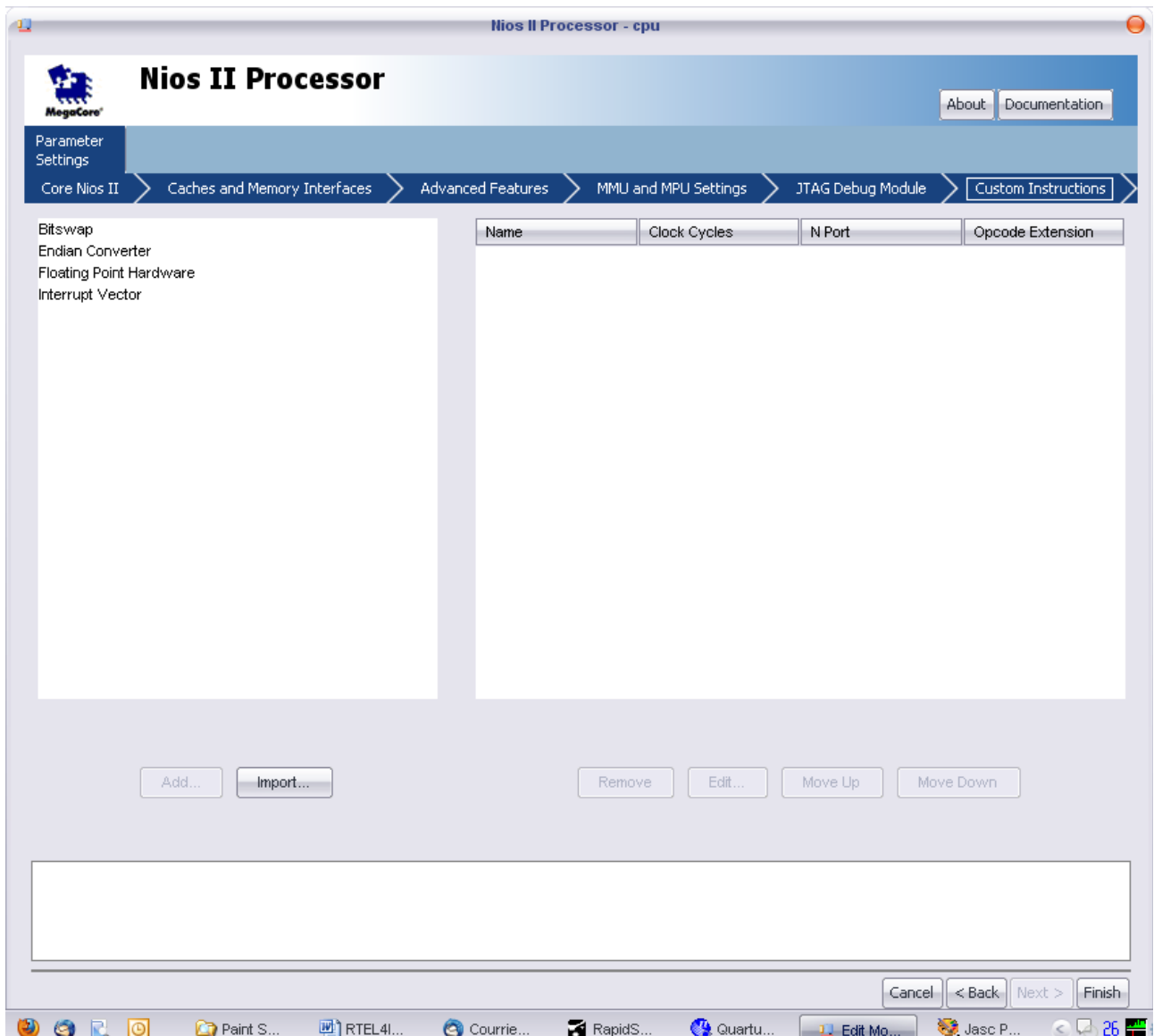


Figure 15 : Dévalidation des *Custom Instructions*

Une particularité importante est de rajouter une mémoire cache de données/instructions pour la MMU.

Il faut rajouter un périphérique appelé ici *fast_tlb_miss_ram_1k* 512 ou 1Ko (512 octets ici) de RAM dual port avec [5] :

- Un port connecté à *tightly_coupled_instruction_master*.
- Un port connecté à *tightly_coupled_data_master*.

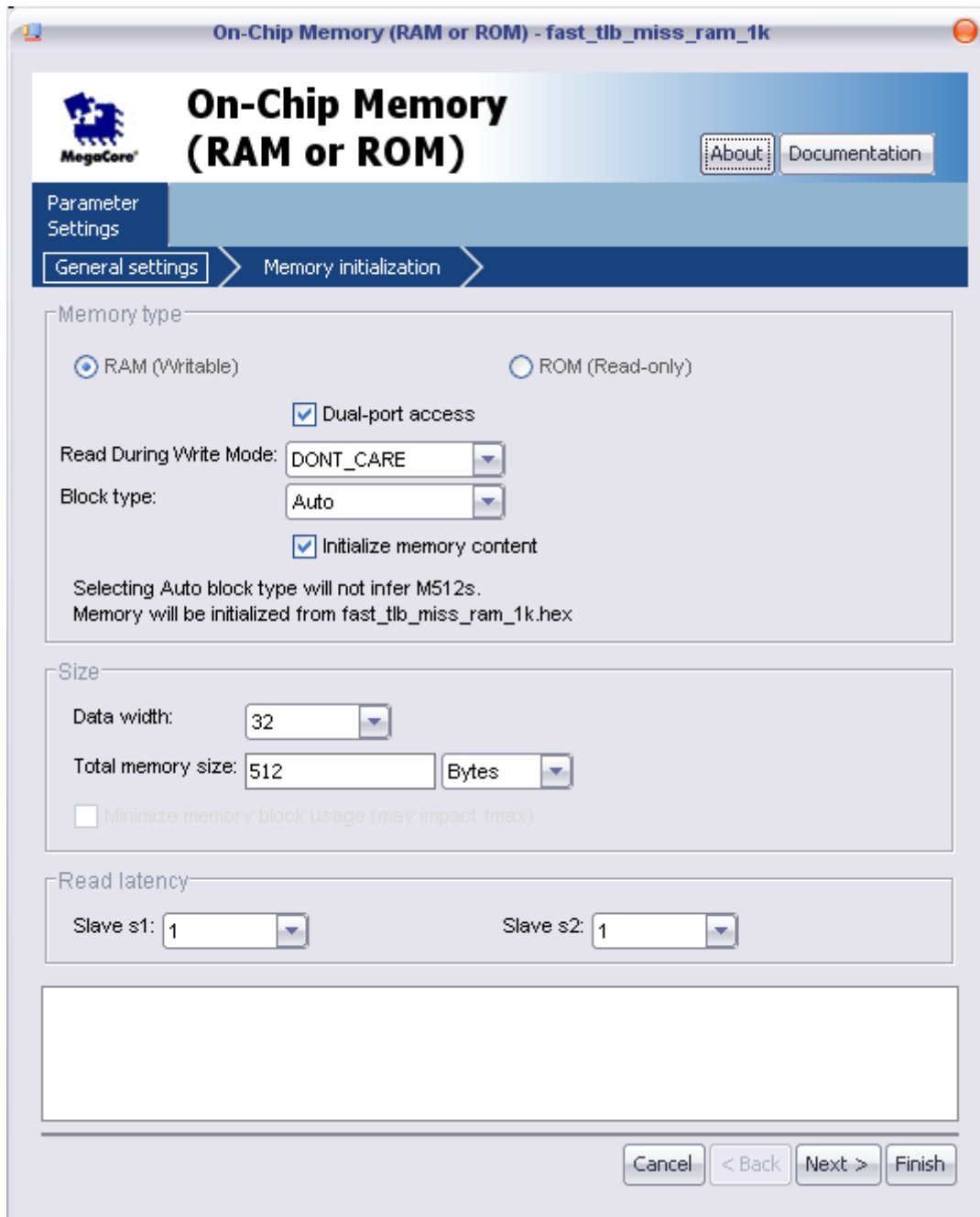
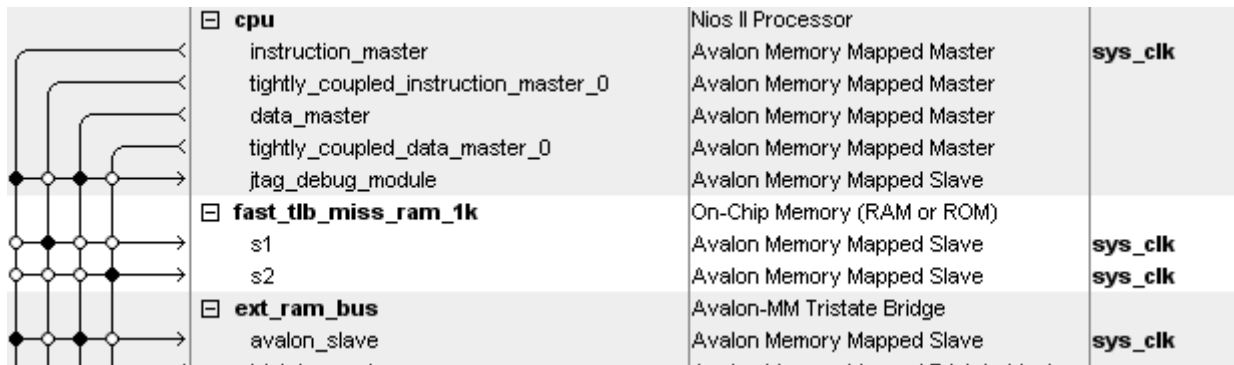


Figure 16 : Configuration et câblage de la MMU

4. Timer `sys_clk_timer`

Le timer `sys_clock_timer` est utilisé par Linux comme *tick timer*. Sa configuration dans *SoPC Builder* est la suivante :

- Timer : 32 bits.
- Période de timeout : 10 ms.
- Preset : custom. Writable period, readable snapshot, Start/Stop control bits.

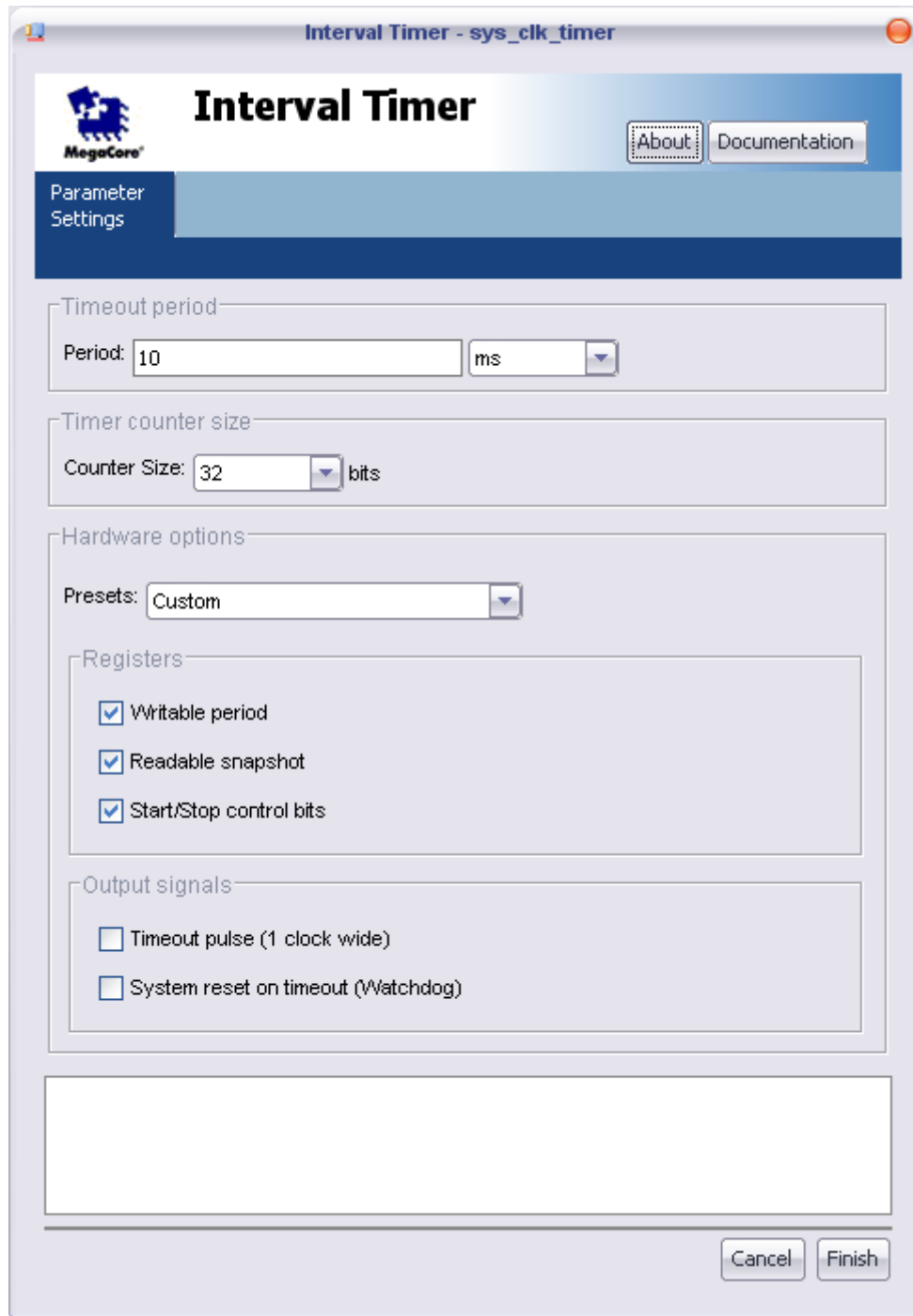


Figure 17 : Configuration du timer Linux `sys_clk_timer`

3.3. Installation des outils Altera sous Linux

Il est possible d'utiliser les outils Altera sous Linux que ce soit pour le design du système SoPC que pour le développement logiciel. Cela permet d'avoir une seule machine sous Linux, ce qui permet de garder une cohérence dans le développement matériel et logiciel.

Le document [3] explique comment faire cette installation. La procédure a été testée sous Fedora 12. On installera les outils Quartus II version 9.1 (à la date d'écriture de ce document) :

```
$ cd
$ wget ftp://ftp.altera.com/outgoing/release/91_quartus_linux.tar
$ wget ftp://ftp.altera.com/outgoing/release/91_nios2eds_linux.tar
$ wget ftp://ftp.altera.com/outgoing/release/91_modelsim_ae_linux.tar
$ tar -xvf 91_nios2eds_linux.tar
$ cd nios2eds
# ./install
$ cd
$ tar -xvf 91_modelsim_ae_linux.tar
$ cd modelsim_ae
# ./install
$ cd
$ tar -xvf 91_quartus_linux.tar
$ cd quartus
# ./install
$ cd
```

L'ensemble des logiciels est installé dans le répertoire */opt/altera9.1*.

On créera le script *n2sdk* que l'on placera sous *~/bin* :

```
#!/bin/bash
# Run this for a Nios II SDK bash shell
export LM_LICENSE_FILE=1700@localhost
SOPC_KIT_NIOS2=/opt/altera9.1/nios2eds
export SOPC_KIT_NIOS2
SOPC_BUILDER_PATH_91=/opt/altera9.1/nios2eds
export SOPC_BUILDER_PATH_91
unset GCC_EXEC_PREFIX
QUARTUS_ROOTDIR=/opt/altera9.1/quartus
export QUARTUS_ROOTDIR
export PERL5LIB=/usr/lib/perl5/5.10.0
bash --rcfile $QUARTUS_ROOTDIR/sopc_builder/bin/nios_bash
```

Pour pouvoir utiliser les outils Altera *Quartus II* et *ModelSim*, il faut bien sûr avoir une licence Altera valable que l'on utilisera en réseau avec un serveur *flexlm*...

Mais l'on peut utiliser les autres outils en ligne de commande.

On installera le JTAG pour le module USB Blaster de la carte cible :

```
# mkdir /etc/jtagd
# cp /opt/altera9.1/quartus/linux/pgm_parts.txt /etc/jtagd/jtagd.pgm_parts
$ n2sdk
[NiosII EDS]$ su
Mot de passe :
```

```
# jtagd
# exit
[NiosII EDS]$ touch ~/.jtag.conf
[NiosII EDS]$ jtagconfig
1) USB-Blaster [USB 2-1.3.2]
   020010DD   EP1S10
```

On vérifie que la commande *nios2-download* fonctionne pour télécharger un fichier dans la cible par le JTAG:

```
[NiosII EDS]$ nios2-download
Using cable "USB-Blaster [USB 2-1.3.2]", device 1, instance 0x00
Pausing target processor: OK
Restarting target processor
[NiosII EDS]$
```

Les outils Altera sont opérationnels sous Linux.

3.4. Configuration logicielle : configuration et compilation de Linux pour NIOS II

Le processeur NIOS II étant configuré avec sa MMU, la **distribution Linux standard** est utilisée.

Il convient dans un premier temps de télécharger la distribution Linux pour NIOS II. On se référera au document [4] pour plus de détails :

```
$ cd
$ wget http://www.niosftp.com/pub/linux/nios2-linux-20090929.tar
```

On décompresse l'archive :

```
$ tar -xvf nios2-linux-20090929.tar
```

On se place dans le répertoire *nios2-linux* :

On suppose que *nios2_linux* représente le chemin du répertoire *nios2-linux*.

```
$ cd $nios2_linux
```

On lance le script *checkout* pour récupérer les différents fichiers sources et la mise à jour par le script *update* :

```
$ ./checkout
$ ./update
```

On installe le compilateur croisé *gcc* pour le processeur NIOS II :

```
$ su
# cp -r toolchain-mmu/x86-linux2/ /opt
```

On ajuste sa variable PATH dans son fichier *profile* (*~/.bash_profile*) :

```
PATH=$PATH: /opt/x86-linux2/bin/
export PATH
```

On vérifie que le compilateur *gcc* pour NIOS II est opérationnel :

```
$ nios2-linux-gnu-gcc -v
```

Using built-in specs.

Target: nios2-linux-gnu

```
Configured with: /opt/nios2gcc4/src/gcc-4.1/configure
--build=i686-pc-linux-gnu --host=i686-pc-linux-gnu --target=nios2-
linux-gnu --enable-threads --disable-libmudflap --disable-libssp
--disable-libstdcxx-pch --with-gnu-as --with-gnu-ld --enable-
languages=c,c++ --enable-shared --enable-symvers=gnu --enable-
__cxa_atexit --disable-nls --prefix=/opt/nios2 --with-
sysroot=/opt/nios2/nios2-linux-gnu/libc --with-build-
sysroot=/opt/nios2gcc4/install/nios2-linux-gnu/libc --disable-
libgomp --enable-poison-system-directories
Thread model: posix
gcc version 4.1.2
```

On suppose que *design* représente le chemin du répertoire contenant le design Quartus II.

Il faut d'abord générer le fichier *custom_fpga.h* qui donnera le lien entre le nom des périphériques et leur mapping mémoire et qui est utilisé par Linux :

```
$ n2sdk
$ cd $design
[NiosII EDS]$ sopc-create-header-files --single custom_fpga.h
```

Le fichier *custom_fpga.h* ainsi créé est à recopier sous *\$nios2_linux/linux2.6/arch/nios2/include/asm* :

```
$ cp custom_fpga.h $nios2_linux/linux-2.6/arch/nios2/include/asm
```

On suppose maintenant que *\$uClinux_dist* représente le chemin du répertoire des fichiers sources de Linux, c'est à dire ici *nios2-linux/uClinux-dist*.

On configure μ Clinux pour le processeur NIOS II :

```
$ cd $uClinux_dist
$ make menuconfig
```

Dans les écrans *menuconfig*, on vérifie que les options suivantes sont bien validées :

```
Vendor/Product Selection --->
  --- Select the Vendor you wish to target
      Vendor (Altera) --->
  --- Select the Product you wish to target
      Altera Products (nios2) --->

Kernel/Library/Defaults Selection --->
  --- Kernel is linux-2.6.x
      Libc Version (None) --->
  [*] Default all settings (lose changes)
  [ ] Customize Kernel Settings
  [ ] Customize Vendor/User Settings
  [ ] Update Default Vendor Settings
```

On configurera ensuite le noyau pour choisir son fichier *custom_fpga.h* :

```
$ make menuconfig
```

```
Menu NiosII Configuration>NiosII FPGA configuration>
```

```
( ) MMU_DEFAULT
( ) MAXIMUM_MMU
(X) CUSTOM_FPGA
```

On peut configurer Linux pour utiliser plutôt la liaison série de la carte cible pour la console Linux au lieu du JTAG émulé en liaison série :

```
$ make menuconfig
```

```
Menu Device Driver>Character devices>Serial drivers
  [ ] 8250/16550 and compatible serial support
      *** Non-8250 serial port support ***
  [ ] Altera JTAG UART support
  [*] Altera UART support
      (4) Maximum number of Altera UART ports
      (115200) Default baudrate for Altera UART ports
  [*] Altera UART console support
```

On compile le noyau Linux :

```
$ make
```

On télécharge ensuite le fichier *zImage* dans la carte cible par le JTAG :

```
$ n2sdk
[NiosII EDS]$ nios2-download -g images/zImage
```

On utilisera *minicom* pour accéder à la liaison série de la carte cible une fois correctement configuré :

L'outil *minicom* étant lancé dans un deuxième terminal, on récupère les traces de boot du noyau μ CLinux :

```
$ minicom
Uncompressing Linux... Ok, booting the kernel.
Linux version 2.6.31-00500-gf604e3a-dirty (kadionik@linux01) (gcc version 4.1.2)
#2 Tue May 11 23:54:12 CEST 2010
console [early0] enabled
Early printk initialized
```

```
Linux/Nios II-MMU
init_bootmem_node(?,0x14f0, 0x1000, 0x2000)
free_bootmem(0x14f0000, 0xb10000)
reserve_bootmem(0x14f0000, 0x200)
Built 1 zonelists in Zone order, mobility grouping off. Total pages: 4064
Kernel command line:
PID hash table entries: 64 (order: 6, 256 bytes)
Dentry cache hash table entries: 2048 (order: 1, 8192 bytes)
Inode-cache hash table entries: 1024 (order: 0, 4096 bytes)
We have 8192 pages of RAM
Memory available: 11172k/5053k RAM, 0k/0k ROM (1805k kernel code, 3247k data)
Hierarchical RCU implementation.
NR_IRQS:32
Calibrating delay loop... 24.57 BogoMIPS (lpj=122880)
Mount-cache hash table entries: 512
NET: Registered protocol family 16
init_BSP(): registering device resources
bio: create slab <bio-0> at 0
NET: Registered protocol family 2
IP route cache hash table entries: 1024 (order: 0, 4096 bytes)
TCP established hash table entries: 512 (order: 0, 4096 bytes)
```

```
TCP bind hash table entries: 512 (order: -1, 2048 bytes)
TCP: Hash tables configured (established 512 bind 512)
TCP reno registered
NET: Registered protocol family 1
msgmni has been set to 21
io scheduler noop registered
io scheduler anticipatory registered
io scheduler deadline registered
io scheduler cfq registered (default)
ttyJ0 at MMIO 0x810820 (irq = 4) is a Altera JTAG UART
console handover: boot [early0] -> real [ttyJ0]
smc91x.c: v1.1, sep 22 2004 by Nicolas Pitre <nico@cam.org>
eth0: SMC91C11xFD (rev 1) at e0800300 IRQ 7 [nowait]
eth0: Invalid ethernet MAC address. Please set using ifconfig
TCP cubic registered
NET: Registered protocol family 17
RPC: Registered udp transport module.
RPC: Registered tcp transport module.
eth0: link down
Welcome to
```



For further information check:
<http://www.uclinux.org/>

BusyBox v1.15.3 (2010-05-11 23:53:41 CEST) hush - the humble shell
Enter 'help' for a list of built-in commands.

```
root: /> ps
  PID USER      VSZ STAT COMMAND
    1 root        1832 S   /init
    2 root          0 SW<  [kthreadd]
    3 root          0 SW<  [ksoftirqd/0]
    4 root          0 SW<  [watchdog/0]
    5 root          0 SW<  [events/0]
    6 root          0 SW<  [khelper]
    9 root          0 SW<  [async/mgr]
   41 root          0 SW<  [kblockd/0]
   47 root          0 SW<  [kseriod]
   63 root          0 SW   [khungtaskd]
   64 root          0 SW   [pdflush]
   65 root          0 SW   [pdflush]
   66 root          0 SW<  [kswapd0]
   67 root          0 SW<  [aio/0]
   68 root          0 SW<  [nfsiod]
  596 root          0 SW<  [rpciod/0]
  614 root        1976 S   -/bin/sh
  615 root        1692 S   /sbin/inetd
  616 root        1968 R   ps
```

3.5. Configuration logicielle : configuration et compilation de PREEMPT-RT pour NIOS II

Il convient de resynchroniser les versions du noyau Linux pour NIOS II et du patch général PREEMPT-RT. On met déjà à jour le noyau Linux pour NIOS II en passant sur la branche *unstable-nios2mmu* :

```
$ cd $nios_linux/linux-2.6
$ git branch unstable-nios2mmu origin/unstable-nios2mmu
$ git checkout -f unstable-nios2mmu
$ git pull
```

On crée une nouvelle branche pour avoir un noyau Linux 2.6.31 pour NIOS II :

```
$ git checkout -b v2.6.31 f604e3aec6e4a5c05ce3b69f172957d92a605ac5
```

On récupère le patch général version 2.6.31.12-rt21 de PREEMPT-RT que l'on applique aux sources du noyau :

```
$ cd ..
$ wget http://www.kernel.org/pub/linux/kernel/projects/rt/patch-2.6.31.12-rt21.bz2
$ cd linux-2.6/
$ bzipcat ../patch-2.6.31.12-rt21.bz2 | patch -p
```

Les 4 fichiers suivants sont rejetés mais il seront corrigés par le patch PREEMPT-RT pour NIOS II :

```
./Makefile.rej
./kernel/perf_counter.c.rej
./kernel/futex.c.rej
./kernel/time/clockevents.c.rej
```

On récupère le patch PREEMPT-RT 2.6.31.12-rt21-nios2-1.0-00 pour NIOS II que l'on applique aux sources du noyau :

```
$ http://www.enseirb.fr/~kadionik/nios2-preempt-rt/patches/preempt-rt-2.6.31.12-rt21-nios2-1.0-00.patch
$ patch -p1 <../preempt-rt-2.6.31.12-rt21-nios2-1.0-00.patch
```

On recompile enfin le noyau Linux en validant PREEMPT-RT dans le menu du choix du mode de préemption :

```
$ cd $uClinux_dist
$ make
```

On télécharge ensuite le fichier *zImage* dans la carte cible par le JTAG :

```
$ n2sdk
[NiosII EDS]$ nios2-download -g images/zImage
```

On utilisera *minicom* pour accéder à la liaison série de la carte cible une fois correctement configuré :

L'outil *minicom* étant lancé dans un deuxième terminal, on récupère les traces de boot du noyau μ Clinux :

BusyBox v1.15.3 (2010-05-04 17:30:42 CEST) hush - the humble shell
Enter 'help' for a list of built-in commands.

```
root: /> ps
  PID USER      VSZ STAT COMMAND
    1 root        1832 S    /init
    2 root          0 SW<  [kthreadd]
    3 root          0 SW<  [sirq-high/0]
    4 root          0 SW<  [sirq-timer/0]
    5 root          0 SW<  [sirq-net-tx/0]
    6 root          0 SW<  [sirq-net-rx/0]
    7 root          0 SW<  [sirq-block/0]
    8 root          0 SW<  [sirq-tasklet/0]
    9 root          0 SW<  [sirq-sched/0]
   10 root          0 SW<  [sirq-hrtimer/0]
   11 root          0 SW<  [sirq-rcu/0]
   12 root          0 SW<  [posixcputmr/0]
   13 root          0 SW<  [watchdog/0]
   14 root          0 SW<  [desched/0]
   15 root          0 SW<  [rcu_sched_grace]
   16 root          0 SW<  [events/0]
   17 root          0 SW<  [khelper]
   20 root          0 SW<  [async/mgr]
   52 root          0 SW<  [kblockd/0]
   58 root          0 SW<  [kseriod]
   74 root          0 SW   [khungtaskd]
   75 root          0 SW   [pdflush]
   76 root          0 SW   [pdflush]
   77 root          0 SW<  [kswapd0]
   78 root          0 SW<  [aio/0]
   79 root          0 SW<  [nfsiod]
  595 root          0 SW<  [irq/4-JTAGUART]
  605 root          0 SW<  [irq/7-eth%d]
  609 root          0 SW<  [rpciod/0]
  627 root        1976 S    -/bin/sh
  628 root        1692 S    /sbin/inetd
  629 root        1968 R    ps
```

Avec la commande *ps*, on voit bien l'action de PREEMPT-RT, les interruptions étant maintenant considérées comme des threads...

4. MESURES DES PERFORMANCES

Les mesures de performances ont été effectuées sur la carte cible Stratix 1S10 d'Altera.

Il faut rappeler que le processeur a une fréquence de fonctionnement de 50 MHz. Les expériences menées ont consisté en la génération d'un signal carré périodique sur un port d'E/S parallèle de la carte cible grâce à un programme écrit en langage C (programme *squaretest-rt* donné en annexe 2).

La précision du signal de sortie est donc soumise aux contraintes de gestion du temps par défaut du système d'exploitation Linux car le portage effectué n'offre pas encore de timers haute résolution (API *hrtimer*). Le but est de mettre en évidence à la fois l'intérêt du système Temps Réel vis-à-vis du système à temps partagé et la conservation du bornage du temps d'exécution que le processeur ait une forte charge de travail ou non.

4.1. Processus ordinaire

On essaie de générer un créneau périodique par le processeur sans charge processeur. La période du créneau demandé est de 50 ms.

On utilise le programme de test *squaretest-rt* avec la priorité la plus faible, soit 1 :

```
$ squaretest-rt 50 1
```

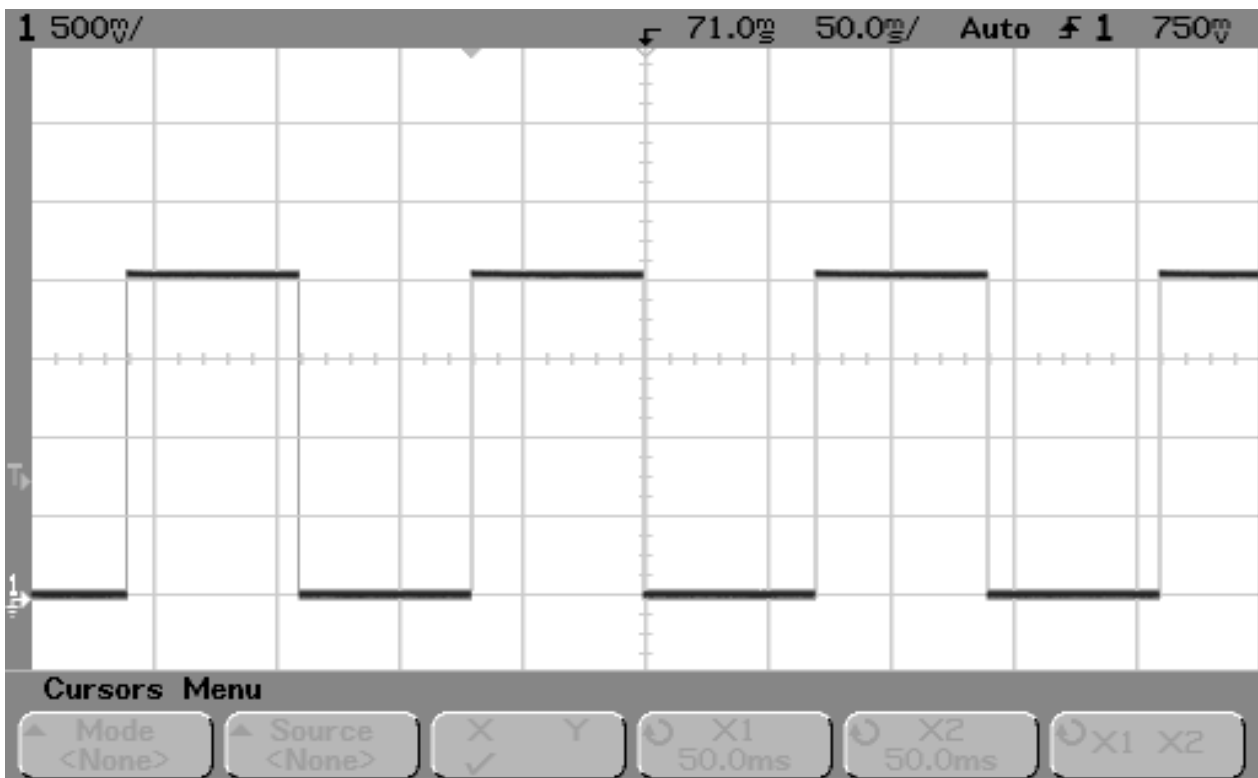


Figure 18 : Signal carré périodique généré sous Linux sans charge

Par la suite, l'expérience est renouvelée en chargeant le processeur NIOS II avec MMU par *ping flooding* de l'interface réseau.

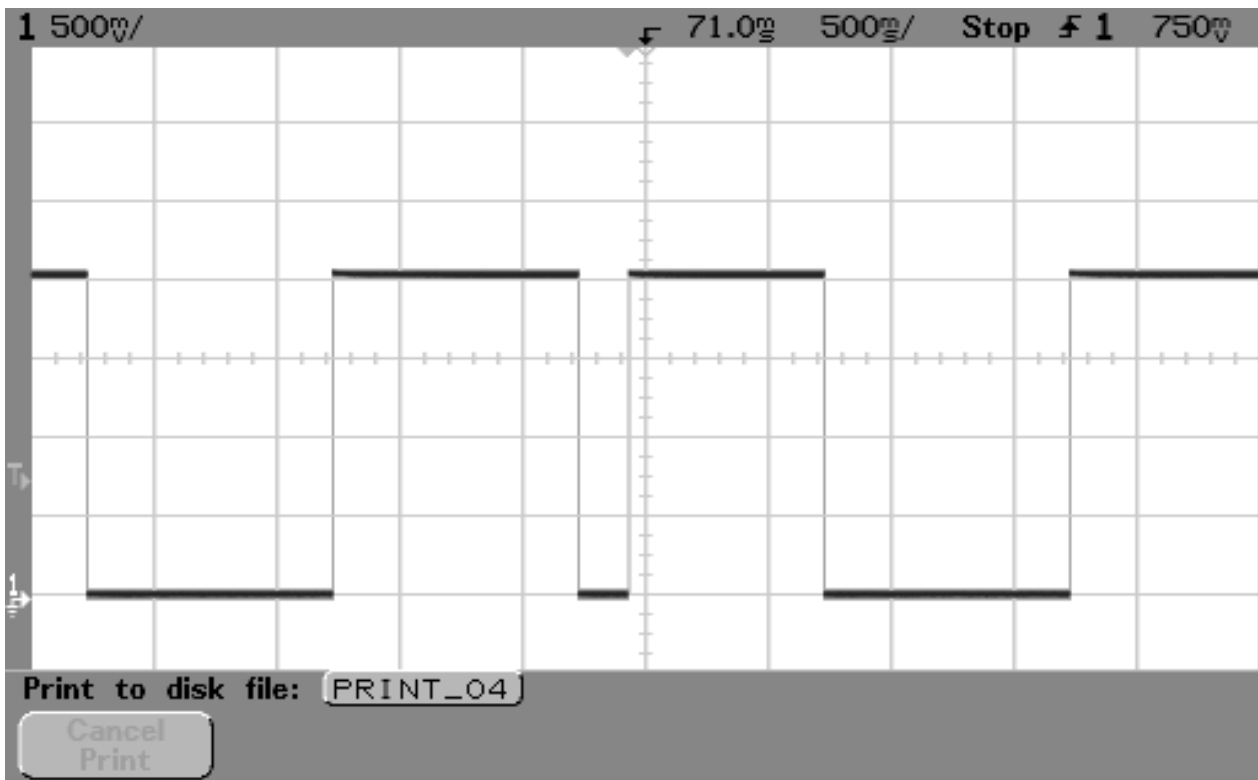


Figure 19 : Signal carré périodique généré sous Linux standard chargé

On voit que le signal généré n'est plus valide du tout.

4.2. Processus Temps Réel

On recommence maintenant mais on utilise le programme de test *squaretest-rt* avec la priorité Temps Réel mou la plus forte, soit 99 :

```
$ squaretest-rt 50 99
```

La mesure est faite directement avec charge du processeur par *ping flooding*.

Le signal est maintenant périodique.

Si l'on fait un zoom, la jige maximale mesurée durant l'intervalle de mesure est de 442 μ s.

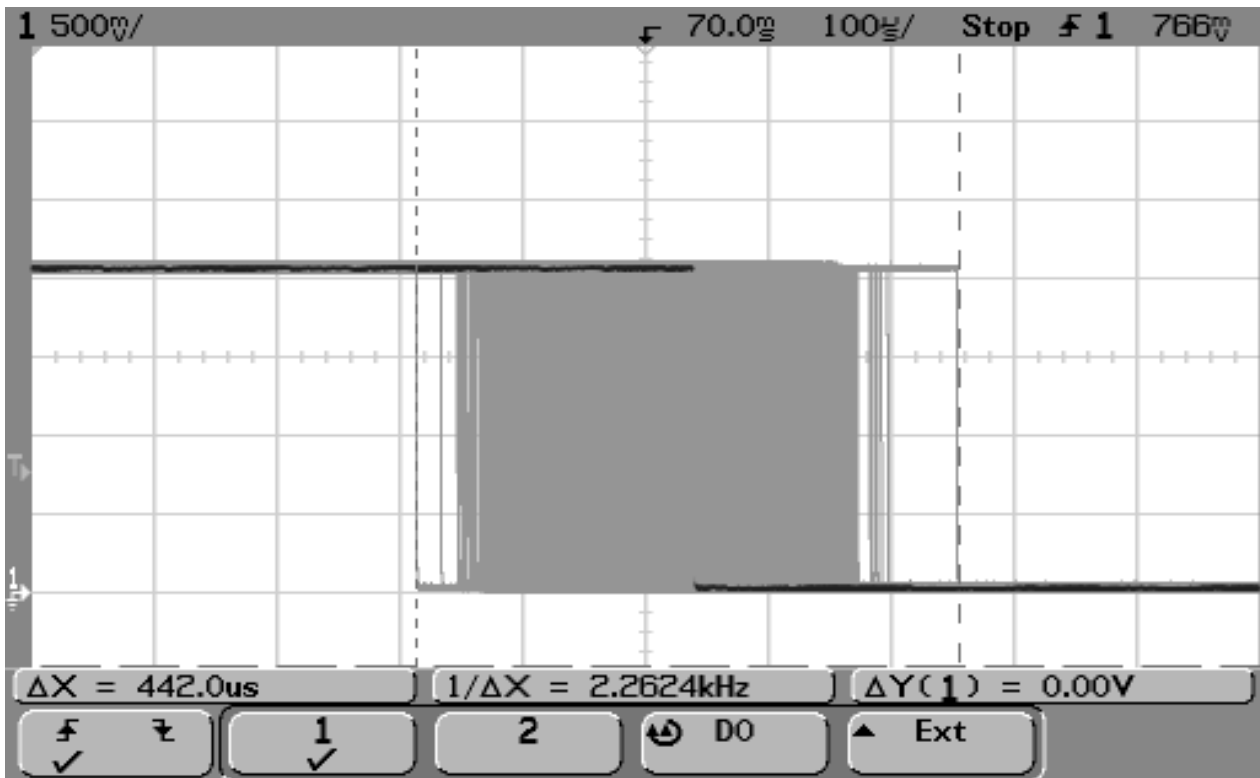


Figure 20 : Signal carré périodique généré sous Linux avec PREEMPT-RT avec charge (zoom)

Avec la priorité 99 et l'ordonnancement Temps Réel SCHED_FIFO, le patch PREEMPT-RT pour NIOS II permet de répondre au mieux à la génération périodique du signal.

5. LIMITATIONS DU PORTAGE

Le portage de PREEMPT-RT est fonctionnel pour le processeur NIOS avec MMU.

Il permet d'avoir des temps de latence mesurées de quelques centaines de μs pour une approche Temps Réel mou pour le processeur NIOS II cadencé ici à 50 MHz.

A l'heure actuelle, le patch proposé n'intègre pas encore de services *clock event service* ou *hrtimer*, ce qui ne permet pas d'avoir accès pour le moment aux timers haute résolution...

6. CONCLUSION

Le portage de PREEMPT-RT pour le processeur NIOS II dans l'environnement Linux (avec MMU) est opérationnel.

Nous avons défini le design matériel qui a permis le portage de PREEMPT-RT.

Le portage est fonctionnel sur une carte cible d'Altera (Stratix 1S10) fonctionnant à 50 MHz. Avec cette fréquence relativement basse pour un processeur embarqué, nous avons des temps de latence de l'ordre de quelques centaines de μ s avec cette solution Temps Réel mou.

7. RÉFÉRENCES DOCUMENTAIRES

- [1] Projet PREEMPT-RT. https://rt.wiki.kernel.org/index.php/Main_Page
- [2] Projet Linux (avec MMU) pour processeur NIOS II. <http://www.nioswiki.com/>
- [3] Page QuartusforLinux. <http://www.nioswiki.com/OperatingSystems/UCLinux/QuartusforLinux>
- [4] Page InstallNios2Linux. <http://www.nioswiki.com/InstallNios2Linux>
- [5] Page sur l'ajout de la MMU dans le processeur NIOS II.
http://www.nioswiki.com/Linux/Creating_a_Nios_II_Design_with_an_MMU
- [6] Page des ressources ENSEIRB-MATMECA de Linux embarqué et du portage de PREEMPT-RT sur processeur NIOS II. <http://www.enseirb.fr/~kadionik/nios2-preempt-rt/>

8. RÉFÉRENCES LOGICIELLES

- [7] Outils Altera sous Linux : <ftp://ftp.altera.com/outgoing/release>
- [8] Portage Linux pour le processeur NIOS II :
http://www.nioswiki.com/Linux/Downloading_Linux_Distribution/
<http://www.niosftp.com/pub/linux/nios2-linux-20090929.tar>
- [9] Compilateur croisé gcc pour processeur NIOS II : <http://www.niosftp.com/pub/linux/nios2-linux-20090929.tar>
- [10] Ces briques logicielles sont en secours sur la page PREEMPT-RT pour le processeur NIOS II de l'ENSEIRB-MATMECA : <http://www.enseirb.fr/~kadionik/nios2-preempt-rt/>

9. ANNEXE 1 : FICHIERS IMPACTÉS PAR LE PORTAGE DE PREEMPT-RT POUR L'ARCHITECTURE NIOS2

9.1. Liste des fichiers génériques modifiés

La liste des fichiers modifiés par le patch PREEMPT-RT concernant les architectures déjà supportées a été supprimée (répertoire *arch/*).

patching file Documentation/kernel-parameters.txt
patching file Documentation/trace/events.txt
patching file Documentation/trace/ftrace.txt
patching file Documentation/trace/function-graph-fold.vim
patching file Documentation/trace/histograms.txt
patching file Documentation/trace/ring-buffer-design.txt
patching file MAINTAINERS
patching file Makefile
patching file arch/Kconfig
patching file block/blk-core.c
patching file drivers/acpi/acpica/acglobal.h
patching file drivers/acpi/acpica/hwregs.c
patching file drivers/acpi/acpica/hwxface.c
patching file drivers/acpi/acpica/utmutex.c
patching file drivers/acpi/ec.c
patching file drivers/acpi/osl.c
patching file drivers/acpi/processor_idle.c
patching file drivers/ata/libata-sff.c
patching file drivers/base/bus.c
patching file drivers/base/core.c
patching file drivers/base/dd.c
patching file drivers/base/power/main.c
patching file drivers/block/hd.c
patching file drivers/block/paride/pseudo.h
patching file drivers/char/random.c
patching file drivers/char/rtc.c
patching file drivers/char/tty_audit.c
patching file drivers/char/tty_buffer.c
patching file drivers/char/tty_ldisc.c
patching file drivers/char/vt.c
patching file drivers/firewire/core-device.c
patching file drivers/gpu/drm/r128/r128_cce.c
patching file drivers/gpu/drm/r128/r128_drv.h
patching file drivers/gpu/drm/r128/r128_state.c
patching file drivers/ide/alim15x3.c
patching file drivers/ide/hpt366.c
patching file drivers/ide/ide-io-std.c
patching file drivers/ide/ide-io.c
patching file drivers/ide/ide-iops.c
patching file drivers/ide/ide-probe.c
patching file drivers/ide/ide-taskfile.c
patching file drivers/ieee1394/nodemgr.c

patching file drivers/infiniband/core/user_mad.c
patching file drivers/infiniband/ulp/ipoib/ipoib_multicast.c
patching file drivers/input/gameport/gameport.c
patching file drivers/input/joystick/analog.c
patching file drivers/input/keyboard/hil_kbd.c
patching file drivers/input/misc/hp_sdc_rtc.c
patching file drivers/input/misc/pcspkr.c
patching file drivers/input/mouse/hil_ptr.c
patching file drivers/input/serio/hil_mlc.c
patching file drivers/input/serio/hp_sdc.c
patching file drivers/macintosh/adb.c
patching file drivers/media/dvb/dvb-core/dvb_frontend.c
patching file drivers/mfd/twl4030-irq.c
patching file drivers/mfd/ucb1x00-core.c
patching file drivers/misc/Kconfig
patching file drivers/misc/Makefile
patching file drivers/misc/hwlat_detector.c
patching file drivers/mmc/card/queue.c
patching file drivers/net/3c527.c
patching file drivers/net/3c59x.c
patching file drivers/net/8139too.c
patching file drivers/net/arm/at91_ether.c
patching file drivers/net/at11c/at11c_main.c
patching file drivers/net/at11e/at11e_main.c
patching file drivers/net/bnx2.c
patching file drivers/net/bnx2x_main.c
patching file drivers/net/chelsio/sge.c
patching file drivers/net/hamradio/6pack.c
patching file drivers/net/hamradio/mkiss.c
patching file drivers/net/irda/sir_dev.c
patching file drivers/net/ixp2000/enp2611.c
patching file drivers/net/ixp2000/ixpdev.c
patching file drivers/net/loopback.c
patching file drivers/net/mlx4/mlx4.h
patching file drivers/net/mv643xx_eth.c
patching file drivers/net/netxen/netxen_nic_init.c
patching file drivers/net/niu.c
patching file drivers/net/ppp_async.c
patching file drivers/net/rionet.c
patching file drivers/net/s2io.c
patching file drivers/net/sungem.c
patching file drivers/net/tehuti.c
patching file drivers/net/tulip/tulip_core.c
patching file drivers/net/wan/cosa.c
patching file drivers/of/base.c
patching file drivers/oprofile/event_buffer.c
patching file drivers/oprofile/oprofilefs.c
patching file drivers/parport/ieee1284.c
patching file drivers/parport/share.c
patching file drivers/pci/access.c
patching file drivers/pci/bus.c
patching file drivers/pci/hotplug/ibmphp_hpc.c

patching file drivers/pci/pci.c
patching file drivers/pcmcia/ds.c
patching file drivers/s390/cio/crw.c
patching file drivers/scsi/aacraid/aacraid.h
patching file drivers/scsi/aacraid/commctrl.c
patching file drivers/scsi/aacraid/commsup.c
patching file drivers/scsi/aacraid/dpcsup.c
patching file drivers/serial/8250.c
patching file drivers/staging/comedi/drivers/dt9812.c
patching file drivers/staging/comedi/drivers/usbdx.c
patching file drivers/staging/comedi/drivers/usbdxfast.c
patching file drivers/staging/cpc-usb/cpc-usb_drv.c
patching file drivers/staging/frontier/alphatrack.c
patching file drivers/staging/frontier/tranzport.c
patching file drivers/staging/go7007/go7007-driver.c
patching file drivers/staging/go7007/go7007-i2c.c
patching file drivers/staging/go7007/go7007-usb.c
patching file drivers/staging/go7007/go7007-v4l2.c
patching file drivers/staging/go7007/s2250-loader.c
patching file drivers/staging/mimio/mimio.c
patching file drivers/staging/octeon/ethernet-mdio.c
patching file drivers/staging/otus/wwrap.c
patching file drivers/staging/p9auth/p9auth.c
patching file drivers/staging/rspiusb/rspiusb.c
patching file drivers/staging/rt2870/common/2870_rtmp_init.c
patching file drivers/usb/core/devio.c
patching file drivers/usb/core/driver.c
patching file drivers/usb/core/hcd.c
patching file drivers/usb/core/message.c
patching file drivers/usb/gadget/inode.c
patching file drivers/usb/misc/ftdi-elan.c
patching file drivers/uwb/umc-bus.c
patching file drivers/uwb/uwb-internal.h
patching file drivers/video/console/fbcon.c
patching file drivers/video/console/vgacon.c
patching file drivers/watchdog/davinci_wdt.c
patching file fs/affs/super.c
patching file fs/aio.c
patching file fs/attr.c
patching file fs/btrfs/locking.c
patching file fs/buffer.c
patching file fs/cifs/cifsglob.h
patching file fs/cifs/cifssmb.c
patching file fs/cifs/connect.c
patching file fs/cifs/misc.c
patching file fs/dcache.c
patching file fs/direct-io.c
patching file fs/exec.c
patching file fs/ext4/inode.c
patching file fs/fat/inode.c
patching file fs/file.c
patching file fs/hfs/bfind.c

patching file fs/hfs/btree.c
patching file fs/hfs/btree.h
patching file fs/hfsplus/bfind.c
patching file fs/hfsplus/btree.c
patching file fs/hfsplus/hfsplus_fs.h
patching file fs/hpfs/buffer.c
patching file fs/hpfs/hpfs_fn.h
patching file fs/hpfs/super.c
patching file fs/inode.c
patching file fs/ioprio.c
patching file fs/jbd/transaction.c
patching file fs/jbd2/transaction.c
patching file fs/namespace.c
patching file fs/nilfs2/mdt.c
patching file fs/ntfs/aops.c
patching file fs/ntfs/file.c
patching file fs/ocfs2/aops.c
patching file fs/ocfs2/file.c
patching file fs/pipe.c
patching file fs/proc/array.c
patching file fs/proc/stat.c
patching file fs/proc/task_mmu.c
patching file fs/reiserfs/xattr.c
patching file fs/smbfs/inode.c
patching file fs/xfs/linux-2.6/mrlock.h
patching file fs/xfs/linux-2.6/xfs_buf.c
patching file fs/xfs/linux-2.6/xfs_buf.h
patching file fs/xfs/xfs_alloc.c
patching file fs/xfs/xfs_bmap.c
patching file fs/xfs/xfs_filestream.c
patching file fs/xfs/xfs_fsops.c
patching file fs/xfs/xfs_ialloc.c
patching file fs/xfs/xfs_itable.c
patching file fs/xfs/xfs_mount.c
patching file fs/xfs/xfs_mount.h
patching file include/acpi/acpiosxf.h
patching file include/asm-generic/bug.h
patching file include/asm-generic/cmpxchg-local.h
patching file include/asm-generic/percpu.h
patching file include/asm-generic/tlb.h
patching file include/asm-generic/vmlinux.lds.h
patching file include/linux/bottom_half.h
patching file include/linux/buffer_head.h
patching file include/linux/console.h
patching file include/linux/device.h
patching file include/linux/fs.h
patching file include/linux/ftrace_event.h
patching file include/linux/hardirq.h
patching file include/linux/hrtimer.h
patching file include/linux/init_task.h
patching file include/linux/interrupt.h
patching file include/linux/irq.h

patching file include/linux/jbd.h
patching file include/linux/jbd2.h
patching file include/linux/kernel.h
patching file include/linux/kernel_stat.h
patching file include/linux/kprobes.h
patching file include/linux/list.h
patching file include/linux/lockdep.h
patching file include/linux/mm.h
patching file include/linux/mm_types.h
patching file include/linux/module.h
patching file include/linux/mutex.h
patching file include/linux/netdevice.h
patching file include/linux/netfilter/x_tables.h
patching file include/linux/netpoll.h
patching file include/linux/oprofile.h
patching file include/linux/page_cgroup.h
patching file include/linux/pagevec.h
patching file include/linux/parport.h
patching file include/linux/percpu-defs.h
patching file include/linux/percpu.h
patching file include/linux/percpu_counter.h
patching file include/linux/perf_counter.h
patching file include/linux/plist.h
patching file include/linux/preempt.h
patching file include/linux/profile.h
patching file include/linux/proportions.h
patching file include/linux/quicklist.h
patching file include/linux/radix-tree.h
patching file include/linux/ring_buffer.h
patching file include/linux/rt_lock.h
patching file include/linux/rtmutex.h
patching file include/linux/rwlock.h
patching file include/linux/rwlock_types.h
patching file include/linux/rwsem-spinlock.h
patching file include/linux/rwsem.h
patching file include/linux/sched.h
patching file include/linux/semaphore.h
patching file include/linux/seqlock.h
patching file include/linux/signal.h
patching file include/linux/skbuff.h
patching file include/linux/smb_fs_sb.h
patching file include/linux/smp.h
patching file include/linux/smp_lock.h
patching file include/linux/spinlock.h
patching file include/linux/spinlock_api_smp.h
patching file include/linux/spinlock_api_up.h
patching file include/linux/spinlock_types.h
patching file include/linux/srcu.h
patching file include/linux/syscalls.h
patching file include/linux/time.h
patching file include/linux/timer.h
patching file include/linux/timex.h

patching file include/linux/topology.h
patching file include/linux/tracepoint.h
patching file include/linux/tty.h
patching file include/linux/uaccess.h
patching file include/linux/usb.h
patching file include/linux/vmstat.h
patching file include/linux/workqueue.h
patching file include/net/bluetooth/hci_core.h
patching file include/trace/define_trace.h
patching file include/trace/events/block.h
patching file include/trace/events/hist.h
patching file include/trace/events/latency_hist.h
patching file include/trace/events/module.h
patching file include/trace/events/sched.h
patching file include/trace/events/syscalls.h
patching file include/trace/ftrace.h
patching file include/trace/syscall.h
patching file init/Kconfig
patching file init/Makefile
patching file init/main.c
patching file ipc/mqueue.c
patching file ipc/msg.c
patching file ipc/sem.c
patching file kernel/Kconfig.preempt
patching file kernel/Makefile
patching file kernel/audit.c
patching file kernel/capability.c
patching file kernel/cgroup.c
patching file kernel/exit.c
patching file kernel/fork.c
patching file kernel/futex.c
patching file kernel/futex_compat.c
patching file kernel/hrtimer.c
patching file kernel/irq/autoprobe.c
patching file kernel/irq/chip.c
patching file kernel/irq/handle.c
patching file kernel/irq/internals.h
patching file kernel/irq/manage.c
patching file kernel/irq/migration.c
patching file kernel/irq/numa_migrate.c
patching file kernel/irq/pm.c
patching file kernel/irq/proc.c
patching file kernel/irq/spurious.c
patching file kernel/itimer.c
patching file kernel/kmod.c
patching file kernel/kprobes.c
patching file kernel/latencytop.c
patching file kernel/lock-internals.h
patching file kernel/lockdep.c
patching file kernel/module.c
patching file kernel/mutex-debug.h
patching file kernel/mutex.c

patching file kernel/notifier.c
patching file kernel/perf_counter.c
patching file kernel/posix-cpu-timers.c
patching file kernel/posix-timers.c
patching file kernel/printk.c
patching file kernel/rcupreempt.c
patching file kernel/rcutorture.c
patching file kernel/relay.c
patching file kernel/res_counter.c
patching file kernel/rt.c
patching file kernel/rtmutex-debug.c
patching file kernel/rtmutex-debug.h
patching file kernel/rtmutex.c
patching file kernel/rtmutex_common.h
patching file kernel/rwlock.c
patching file kernel/rwsem.c
patching file kernel/sched.c
patching file kernel/sched_cpupri.c
patching file kernel/sched_cpupri.h
patching file kernel/sched_debug.c
patching file kernel/sched_fair.c
patching file kernel/sched_idletask.c
patching file kernel/sched_rt.c
patching file kernel/sched_stats.h
patching file kernel/semaphore.c
patching file kernel/signal.c
patching file kernel/smp.c
patching file kernel/softirq.c
patching file kernel/softlockup.c
patching file kernel/spinlock.c
patching file kernel/srcu.c
patching file kernel/stop_machine.c
patching file kernel/sys.c
patching file kernel/sysctl.c
patching file kernel/time.c
patching file kernel/time/clockevents.c
patching file kernel/time/clocksource.c
patching file kernel/time/ntp.c
patching file kernel/time/tick-broadcast.c
patching file kernel/time/tick-common.c
patching file kernel/time/tick-internal.h
patching file kernel/time/tick-sched.c
patching file kernel/time/timekeeping.c
patching file kernel/time/timer_list.c
patching file kernel/time/timer_stats.c
patching file kernel/timer.c
patching file kernel/trace/Kconfig
patching file kernel/trace/Makefile
patching file kernel/trace/blktrace.c
patching file kernel/trace/ftrace.c
patching file kernel/trace/kmemtrace.c
patching file kernel/trace/latency_hist.c

patching file kernel/trace/ring_buffer.c
patching file kernel/trace/trace.c
patching file kernel/trace/trace.h
patching file kernel/trace/trace_boot.c
patching file kernel/trace/trace_entries.h
patching file kernel/trace/trace_event_profile.c
patching file kernel/trace/trace_event_types.h
patching file kernel/trace/trace_events.c
patching file kernel/trace/trace_events_filter.c
patching file kernel/trace/trace_export.c
patching file kernel/trace/trace_functions.c
patching file kernel/trace/trace_functions_graph.c
patching file kernel/trace/trace_irqsoff.c
patching file kernel/trace/trace_mmio.c
patching file kernel/trace/trace_output.c
patching file kernel/trace/trace_output.h
patching file kernel/trace/trace_power.c
patching file kernel/trace/trace_sched_switch.c
patching file kernel/trace/trace_sched_wakeup.c
patching file kernel/trace/trace_selftest.c
patching file kernel/trace/trace_stack.c
patching file kernel/trace/trace_stat.c
patching file kernel/trace/trace_stat.h
patching file kernel/trace/trace_syscalls.c
patching file kernel/trace/trace_workqueue.c
patching file kernel/tracepoint.c
patching file kernel/user.c
patching file kernel/workqueue.c
patching file lib/Kconfig
patching file lib/Kconfig.debug
patching file lib/Makefile
patching file lib/debugobjects.c
patching file lib/dec_and_lock.c
patching file lib/kernel_lock.c
patching file lib/locking-selftest.c
patching file lib/percpu_counter.c
patching file lib/plist.c
patching file lib/proportions.c
patching file lib/radix-tree.c
patching file lib/ratelimit.c
patching file lib/rwsem-spinlock.c
patching file lib/rwsem.c
patching file lib/scatterlist.c
patching file lib/spinlock_debug.c
patching file mm/bounce.c
patching file mm/filemap.c
patching file mm/highmem.c
patching file mm/memcontrol.c
patching file mm/memory.c
patching file mm/mmap.c
patching file mm/oom_kill.c
patching file mm/page_alloc.c

patching file mm/page_cgroup.c
patching file mm/quicklist.c
patching file mm/slab.c
patching file mm/swap.c
patching file mm/vmscan.c
patching file mm/vmstat.c
patching file net/bluetooth/hci_core.c
patching file net/core/dev.c
patching file net/core/flow.c
patching file net/core/netpoll.c
patching file net/core/skbuff.c
patching file net/core/sock.c
patching file net/ipv4/icmp.c
patching file net/ipv4/netfilter/arp_tables.c
patching file net/ipv4/netfilter/ip_tables.c
patching file net/ipv4/route.c
patching file net/ipv4/tcp.c
patching file net/ipv6/netfilter/ip6_tables.c
patching file net/netfilter/core.c
patching file net/netlink/af_netlink.c
patching file net/sched/sch_generic.c
patching file scripts/Kbuild.include
patching file scripts/checkpatch.pl
patching file scripts/mkcompile_h
patching file scripts/recordmcount.pl
patching file security/keys/permission.c
patching file security/keys/proc.c
patching file sound/drivers/pcsp/pcsp.h
patching file sound/drivers/pcsp/pcsp_input.c
patching file sound/drivers/pcsp/pcsp_lib.c
patching file sound/soc/s3c24xx/s3c2443-ac97.c
patching file tools/perf/util/parse-events.c
patching file virt/kvm/kvm_main.c

9.2. Liste des fichiers spécifiques modifiés pour l'architecture nios2

patching file arch/nios2/include/asm/system.h
patching file arch/nios2/kernel/entry.S
patching file arch/nios2/kernel/irq.c
patching file arch/nios2/kernel/process.c
patching file arch/nios2/kernel/signal.c
patching file arch/nios2/kernel/syscalltable.S
patching file arch/nios2/kernel/time.c
patching file arch/nios2/mm/cacheflush.c
patching file arch/nios2/mm/fault.c
patching file arch/nios2/mm/mmu_context.c
patching file drivers/serial/altjuart.c
patching file drivers/serial/altuart.c
patching file include/asm-generic/bitops/atomic.h
patching file kernel/futex.c
patching file kernel/perf_counter.c

patching file kernel/time/clockevents.c
patching file Makefile

10. ANNEXE 2 : FICHIER SOURCE SQUARETEST-RT DE GÉNÉRATION D'UN SIGNAL PÉRIODIQUE

```
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <unistd.h>
#include <pthread.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <sys/ioctl.h>
#include <signal.h>
#include <sys/mman.h>
#include <time.h>
#include <sched.h>
#include <string.h>

#define MAX_SAFE_STACK (8*1024) /* The maximum stack size which is
                                guaranteed safe to access without
                                faulting */

void stack_pdefault(void) {
    unsigned char dummy[MAX_SAFE_STACK];

    memset(&dummy, 0, MAX_SAFE_STACK);
}

int main(int argc, char **argv) {
    int fd;
    char value;
    int period;
    int priority;
    struct sched_param param;

    if(argc != 3) {
        printf("%% squaretest-rt period_in_ms priority\n");
        exit(-1);
    }

    period = atoi(argv[1]);
    period *= 1000;
    priority = atoi(argv[2]);

    printf("squaretest-rt: period=%d us, priority=%d\n", period, priority);

    /* Declare ourself as a real time task */
    param.sched_priority = priority;
    if(sched_setscheduler(0, SCHED_FIFO, &param) == -1) {
        perror("sched_setscheduler failed");
        exit(-1);
    }

    /* Lock memory */
    if(mlockall(MCL_CURRENT|MCL_FUTURE) == -1) {
        perror("mlockall failed");
        exit(-2);
    }
}
```

```
/* Pre-fault our stack */
stack_pEFAULT();

fd=open("/dev/leds", O_RDWR | O_SYNC);
if(fd < 0) {
    printf("Failed to open /dev/leds\n");
    exit(-1);
}

while(1) {
    value = 0xff;
    write(fd, &value, 1);
    fsync(fd);
    usleep(period);

    value = 0;
    write(fd, &value, 1);
    fsync(fd);
    usleep(period);
}

close(fd);
exit(0);
}
```

Usage :

Génération d'un signal périodique de 50 ms avec un processus de priorité non Temps Réel :

```
$ squaretest-rt 50 1
```

Génération d'un signal périodique de 50 ms avec un processus de priorité Temps Réel PREEMPT-RT de priorité la plus forte :

```
$ squaretest-rt 50 99
```

11. ANNEXE 3 : PILOTE DE PÉRIPHÉRIQUE POUR ACCÉDER AUX LEDS DE LA CARTE CIBLE

Pour observer le signal électrique, un oscilloscope est branché sur une des 8 leds de la carte.

```
#include <linux/kernel.h>      /* We're doing kernel work */
#include <linux/module.h>      /* Specifically, a module */
#include <linux/fs.h>
#include <asm/uaccess.h>      /* for get_user and put_user */

MODULE_AUTHOR("Patrice Kadionik");
MODULE_LICENSE("GPL");
MODULE_DESCRIPTION("Module for led access");
MODULE_SUPPORTED_DEVICE("none");

#include <asm/custom_fpga.h>
#define LED_BASE LED_PIO_BASE // 0x810880

#define MAJOR_NUM          100
#define DEVICE_FILE_NAME  "leds"
#define DEVICE_NAME        "leds"

#undef DEBUG
#undef DEBUG2

static int Device_Open = 0;
static volatile char *ptr;
static char buf[1];

static int device_open(struct inode *inode, struct file *file)
{
#ifdef DEBUG
    printk(KERN_ALERT "device_open(%p)\n", file);
#endif

    if (Device_Open)
        return -EBUSY;

    Device_Open++;

    try_module_get(THIS_MODULE);
    return 0;
}

static int device_release(struct inode *inode, struct file *file)
{
#ifdef DEBUG
    printk(KERN_ALERT "device_release(%p,%p)\n", inode, file);
#endif

    Device_Open--;

    module_put(THIS_MODULE);
}
```

```
        return 0;
    }

static ssize_t device_write(struct file *file, const char __user * buffer,
size_t count, loff_t * offset)
{
#ifdef DEBUG
    printk(KERN_ALERT "device_write(%p,%s,%d)", file, buffer, count);
#endif
    if (copy_from_user(buf, buffer, count))
        return -EFAULT;

#ifdef DEBUG2
    printk(KERN_ALERT "data=%x", buf[0]);
#endif

    local_irq_disable();
    *ptr = buf[0];
    local_irq_enable();

    return count;
}

struct file_operations Fops = {
    .read = NULL,
    .write = device_write,
    .ioctl = NULL,
    .open = device_open,
    .release = device_release,    /* a.k.a. close */
};

static int device_init(void)
{
    int ret_val;

    ptr = (char *)ioremap_nocache(LED_BASE, 1);

    ret_val = register_chrdev(MAJOR_NUM, DEVICE_NAME, &Fops);

    if (ret_val < 0) {
        printk(KERN_ALERT "%s failed with %d\n",
            "Sorry, registering the character device ", ret_val);
        return ret_val;
    }

    printk(KERN_ALERT "Leds registration success.The major device number is
%d.\n", MAJOR_NUM);
    printk(KERN_ALERT "mknod /dev/%s c %d 0\n", DEVICE_FILE_NAME, MAJOR_NUM);
    printk(KERN_ALERT "Leds remapped in VM at %x\n", ptr);

    return 0;
}

static void device_exit(void)
{
    unregister_chrdev(MAJOR_NUM, DEVICE_NAME);
}

module_init(device_init);
```

```
module_exit(device_exit);
```

Usage :

```
# mknod /dev/leds c 100 0
```