

Tutorial 4

Realizzazione di processore NIOS-2 e di alcune interfacce di I/O

Descrizione: L'FPGA montata sulla DE1 ha risorse sufficienti per poter realizzare al suo interno un microprocessore. In questo tutorial verranno analizzati i rudimenti per realizzare, configurare e programmare opportunamente tale processore.

Scopo: Realizzazione di una semplice architettura di un microprocessore, e di alcune interfacce dedicate di I/O.

Apprendimenti previsti:

- Utilizzo del tool "SOPC Builder" per la configurazione dell'architettura del processore
- Utilizzo del tool "Altera Monitor Program" per la programmazione ed il debugging del software
- Impiego di memorie esterne all'FPGA e problematiche di timing

Procedimento:

Si inizi un nuovo progetto per Ciclone II - EP2C20F484C7N

Definizione dell'architettura del Processore.

Si apra il tool SOPC Builder

Tool > SOPC Builder

Si definisca un nome da assegnare al sistema composto dal processore e le sue periferiche ed il linguaggio con cui verrà descritto.

Si definiscano i blocchi che compongono l'architettura:

1. Una CPU:
 - Processor > Nios II Processor - **ADD**
 - Si configuri nella versione Nios II/e
2. Una memoria interna all'FPGA:
 - Memories ... > On Chip > On Chip Memories (RAM or ROM)
 - Si configuri come RAM da 16384 Bytes ed un "Data Width" da 32 bits
3. Un' Interfaccia UART - JTAG:
 - Interface Protocol > Serial > JTAG UART
 - Configurazione come da default
4. Un' Interfaccia Parallela di I/O come ingresso:
 - Peripherals > Microcontroller Peripherals > PIO
 - Si configuri come Ingresso a 10 bits
5. Un' Interfaccia Parallela di I/O come Uscita:
 - Peripherals > Microcontroller Peripherals > PIO

- Si configuri come Uscita a 10 bits

6. Un' identificativo del sistema:

- Peripherals > Debug and Performance > System ID Peripheral
- Questa periferica non richiede alcuna configurazione

A questo punto si possono modificare i nomi delle varie periferiche, è consigliabile dare un nome mnemonico che evidenzi la loro futura funzione (ed esempio la PIO in ingresso si potrebbe nominare SWITCHES e quella in uscita LED). Per cambiare nome si evidenzi la periferica e si digiti `ctl-r`. Analogamente si può modificare il nome del clock. Sebbene in questa esercitazione non venga usata, per il futuro è consigliabile denominare la periferica identificativa del sistema "sysid".

Gli indirizzi delle varie periferiche possono essere modificati uno alla volta oppure automaticamente:

System > Auto Assign Base Addresses

Riaprire la CPU (doppio click) e modificare la configurazione sia per quanto riguarda "reset vector" che "exception vector" e scegliere come memoria alla quale fare riferimento la memoria realizzata al punto 2.

Da un punto di vista globale l'architettura del processore è riportata in figura

The screenshot shows the 'Clock Settings' window in SOPC Builder. The 'Device Family' is set to 'Cyclone II'. A table lists the clock source 'clk' as 'External' at '50,0' MHz. Below this is a table of modules and their memory addresses:

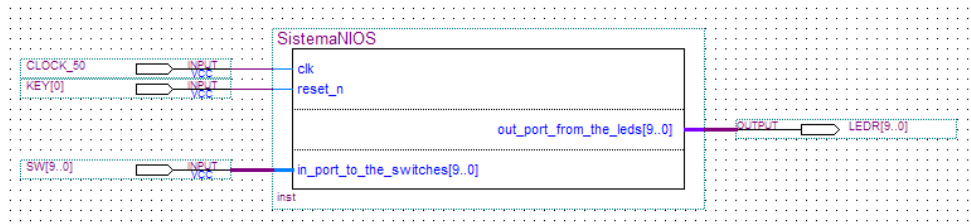
Use	Conn...	Module Name	Description	Clock	Base	End	Tags
<input checked="" type="checkbox"/>		cpu	Nios II Processor				
		instruction_master	Avalon Memory Mapped Master	clk			
		data_master	Avalon Memory Mapped Master	clk			
		jtag_debug_module	Avalon Memory Mapped Slave	clk	# 0x00008800	0x00008fff	IRQ 0 IRQ 31
<input checked="" type="checkbox"/>		mem	On-Chip Memory (RAM or ROM)	clk	# 0x00004000	0x00007fff	
<input checked="" type="checkbox"/>		s1	Avalon Memory Mapped Slave	clk	# 0x00009020	0x00009027	
		jtag_uart_0	JTAG UART	clk	# 0x00009020	0x00009027	
		avalon_jtag_slave	Avalon Memory Mapped Slave	clk	# 0x00009020	0x00009027	
<input checked="" type="checkbox"/>		switches	PIO (Parallel IO)	clk	# 0x00009000	0x0000900f	
		s1	Avalon Memory Mapped Slave	clk	# 0x00009000	0x0000900f	
<input checked="" type="checkbox"/>		leds	PIO (Parallel IO)	clk	# 0x00009010	0x0000901f	
		s1	Avalon Memory Mapped Slave	clk	# 0x00009010	0x0000901f	
<input checked="" type="checkbox"/>		sysid	System ID Peripheral	clk	# 0x00009028	0x0000902f	
		control_slave	Avalon Memory Mapped Slave	clk	# 0x00009028	0x0000902f	

Si salvi la configurazione e si prenda nota degli indirizzi ai quali sono state mappate le varie periferiche; si clicchi su "Generate". A processo ultimato si può chiudere la finestra dell'SOPC Builder.

Realizzazione del sistema completo

Il processore così realizzato può trovare posto in un sistema completo ove può essere integrato con eventuali ulteriori blocchi hardware.

- Si crei un nuovo schematico
- Si importi il Sistema appena generato
- Si colleghino tutti gli ingressi ed uscite ad opportuni PIN
- Si assegnino dei nomi coerenti col file di vincoli che si verrà successivamente ad assegnare al progetto



- Si importi un opportuno file di vincoli
- Ci si assicuri che lo schematico appena generato sia configurato come “TOP Level ENTITY”
- Si compili l’intero progetto
- Si esegua il download su FPGA

Sviluppo Software

A questo punto il sistema è pronto per funzionare, ma vi si deve caricare l’opportuno software. Per fare questo esistono vari sistemi, in questo tutorial utilizzeremo l’“ALTERA monitor Program” : un sistema utile per la compilazione del sorgente, il download del codice nella memoria del processore ed il debugging.

Si supponga che si voglia far girare sul processore il seguente semplice programma:

```
#define Switches (volatile int *) 0x0009000
#define LEDR (int *) 0x0009010

int main()
{
  int a;
  while (1)
  {a = *Switches;
  *LEDR = a;}
}
```

Ovviamente gli indirizzi rispettivamente di “Switches” e “LEDR” devono corrispondere a quelli del sistema hardware sviluppato al passo precedente. Utilizzando un normale text editor si editi questo testo e lo si salvi all’interno di un opportuno direttorio. (E’ consigliabile adottare un opportuno direttorio dedicato al software all’interno della cartella in cui si è sviluppato il progetto hardware)

Si lanci il programma “Altera Monitor Program” (se non sono presenti collegamenti sul desktop un probabile percorso è il seguente)

```
Start > Programmi > Altera > University Program > Altera Monitor Program > Altera Monitor Program
```

Si apra un nuovo progetto

```
File > New Project
```

Si scelga il direttorio in cui far risiedere il progetto (è consigliabile usare lo stesso direttorio nel quale è stato salvato il file in C) ed il nome del progetto stesso. (NEXT)

Si selezioni il sistema su cui far risiedere il progetto, si scelga il sistema

```
Custom
```

Si indichi al sistema il file `.ptf` (che contiene le informazioni sulla struttura e sull'architettura del sistema realizzato in HW) che risiede nel direttorio in cui si è sviluppato il progetto HW.
Si indichi eventualmente il file `.sof` con cui configurare l'FPGA. (NEXT)

Si selezioni la tipologia di linguaggio che si desidera adottare

C Program

E si includa il file C che è stato salvato nei passi precedenti. (NEXT)

Se vi fossero più CPU disponibili o più periferiche da usarsi come STDIO si potrebbero indicare in questa pagina, altrimenti (NEXT)

Se vi fossero più memorie a disposizione si potrebbe scegliere in quale far risiedere il testo ed i dati (NEXT)

Il sistema chiede se si vuole riconfigurare l'FPGA col file `.sof` indicato in precedenza (è un metodo comodo per configurare l'FPGA direttamente dall'interno di "Altera Monitor Program" senza dover utilizzare il programma quartus).

A questo punto non rimane che da compilare il sorgente

Action > Compile (ctrl-Shift-C)

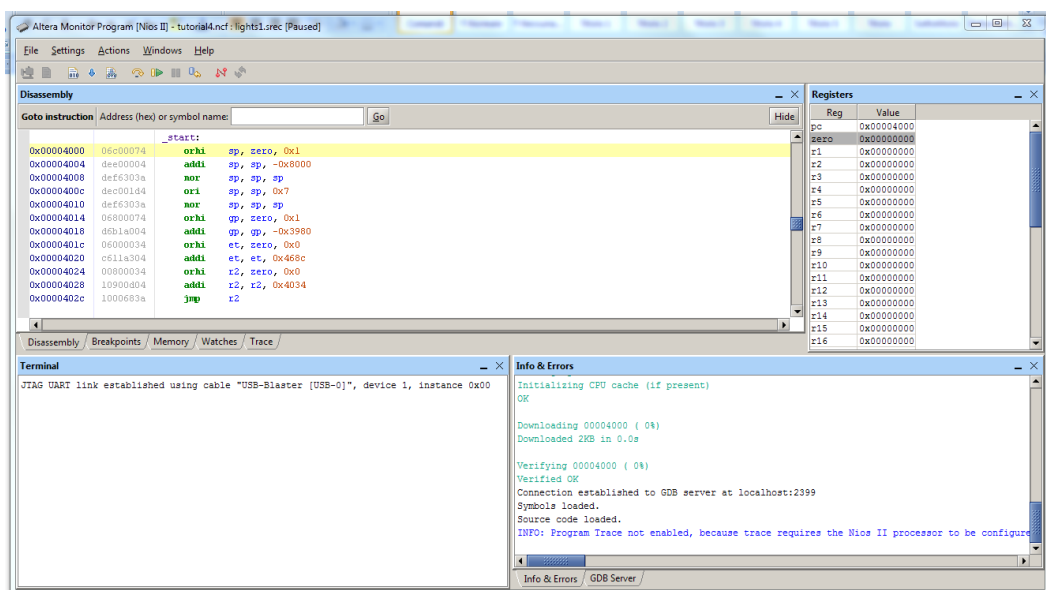
E fare il download del codice nella memoria del processore

Action > Load (ctrl-shift-L)

Alternativamente le due azioni possono essere svolte in rapida sequenza una di seguito all'altra con

Action > Compile & Load (F5)

Se tutto ha funzionato correttamente si dovrebbe ottenere una schermata simile



A questo punto il sistema è pronto e programmato, per avviarlo

Action > Continue (F3) – oppure si usino gli opportuni tasti

Nota 1: Si provi ad usare il tasto di “pause” o in alternativa il tasto di reset del processore (nell’esempio usato è KEY[0])

Nota 2: Si Noti che il LEDR[0] si accende contemporaneamente a LEDR[8] e LEDR[1] a LEDR[9]. Si provi ora a definire i puntatori non a “int” bensì a “long int”

Nota 3: si noti che sebbene il ciclo while si sarebbe potuto scrivere più semplicemente come

```
while (1)
    {*LEDR = *Switches;}
```

Questo può comportare problemi di interpretazione al compilatore, ovvero ad ogni passo del ciclo si deve aggiornare la posizione di memoria occupata dai LED, ma questo deve essere fatto in base all’attuale dato presente negli switches oppure in base al precedente?

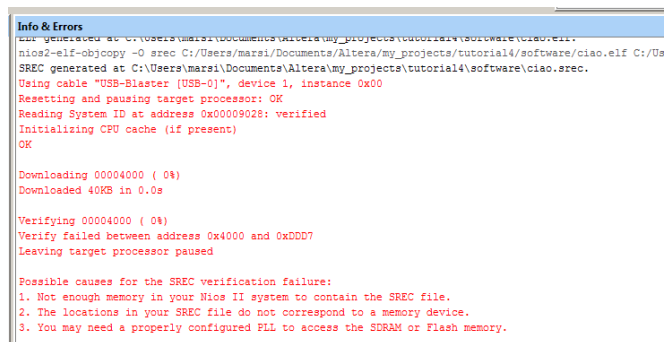
Nota 4: si provi ora qualche altro programma, ad esempio:

```
for (a=1;a<34235;a++) *LEDR = a;
```

Nota 5: si provi ora con:

```
#include <stdio.h>
int main()
    {printf("CIAO/n");}
```

Sebbene sintatticamente corretto, in fase di download su dispositivo si ottiene il seguente errore:



```
Info & Errors
SREC generated at C:\Users\mars1\Documents\Altera\my_projects\tutorial4\software\ciao.elf
nios2-elf-objcopy -O srec C:\Users\mars1\Documents\Altera\my_projects\tutorial4\software\ciao.elf C:/Us
SREC generated at C:\Users\mars1\Documents\Altera\my_projects\tutorial4\software\ciao.srec.
Using cable "USB-Blaster (USB-0)", device 1, instance 0x00
Resetting and pausing target processor: OK
Reading System ID at address 0x00009028: verified
Initializing CPU cache (if present)
OK
Downloading 00004000 ( 0%)
Downloaded 40KB in 0.0s
Verifying 00004000 ( 0%)
Verify failed between address 0x4000 and 0xDDD7
Leaving target processor paused
Possible causes for the SREC verification failure:
1. Not enough memory in your Nios II system to contain the SREC file.
2. The locations in your SREC file do not correspond to a memory device.
3. You may need a properly configured PLL to access the SDRAM or Flash memory.
```

Si può infatti notare che la memoria interna all’FPGA da dedicare al processore risulta abbastanza limitata 16384 bits sono in pratica 512 parole da 32 bits, e pertanto essa non riesce a contenere le risorse utili a gestire le funzioni di input/output su STDIO.

Si deve pertanto far ricorso alle risorse esterne all’FPGA ovvero alla SRAM ed alla SDRAM montate sulla DE1.

Utilizzo di memorie esterne.

SRAM

La DE1 monta una memoria SRAM IS61LV25616 da 256K x 16 bits.

Per includere questa tra le periferiche del sistema da sviluppare si può utilizzare il controllore già sviluppato in ambito dell'“University Program” che mette gratuitamente a disposizione degli utenti una serie di interfacce dedicate appunto ai componenti montati sulle schede DE1, DE2, DE2-70 e DE3.

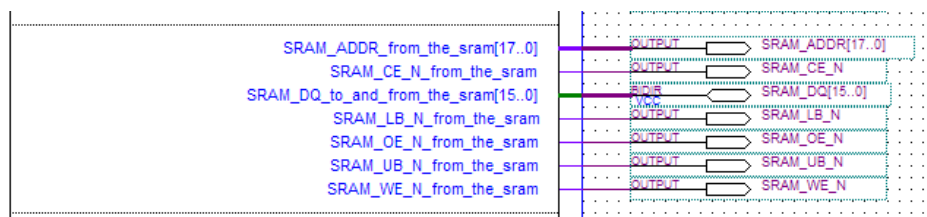
All'interno dell'SOPC Builder selezionare:

University Program > Memory > SRAM/SSRAM Controller

Come configurazione si scelga la scheda su cui si sta lavorando (DE1) ed eventualmente se questa memoria volesse essere impiegata come Frame Buffer in un sistema di elaborazione video (in questo caso no).

Dopo aver rilanciato “GENERATE” il sistema complessivo comprenderà anche le linee per il collegamento con la memoria esterna. Si dovrà pertanto collegare queste ad opportuni PIN, e dare a questi ultimi un nome uguale a quello assegnato all'interno del file di vincoli per garantire il corretto collegamento.

NOTA BENE: mentre tutte le linee sono configurate come uscite, la linea dati deve essere configurata come bidirezionale (infatti normalmente in una memoria i dati si possono sia scrivere che leggere)



SDRAM

La DE1 monta una memoria SDRAM IS42S16400 da 1Mword x 16 bit x 4 Banks (64 Mbits) ove gli indirizzi sono organizzati in matrici di 12 righe x 8 colonne. La gestione dei segnali di una SDRAM, se fatta direttamente a livello hardware risulta abbastanza problematica (si ricordi ad esempio che necessita di opportuni cicli di refresh). Però per l'uso che se ne fa in questo tutorial torna comodo l'impiego di un controller già opportunamente sviluppato da terze parti che rende praticamente trasparente tutta la gestione dei segnali.

All'interno dell'SOPC Builder selezionare:

Memories & Memory Controller > SDRAM > SDRAM Controller

Successivamente si configuri il blocco come

- Preset: Custom
- Architecture: 1 chip select, 4 banks
- Address width: row 12, column 16

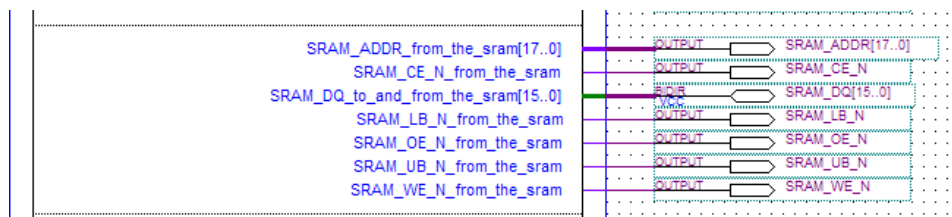
Si lascino inalterati tutti gli altri parametri di configurazione.

Si riassegnino automaticamente tutti gli indirizzi. Il sistema completo dei controllori per le memorie SDRAM e SRAM dovrebbe più o meno coincidere con quanto sotto riportato.

Use	Conn...	Module Name	Description	Clock	Base	End	Te
<input checked="" type="checkbox"/>		cpu	Nios II Processor				
		instruction_master	Avalon Memory Mapped Master	clk			
		data_master	Avalon Memory Mapped Master			IRQ 0 IRQ 31	
		jtag_debug_module	Avalon Memory Mapped Slave		0x01108800	0x01108fff	
<input checked="" type="checkbox"/>		mem	On-Chip Memory (RAM or ROM)				
		s1	Avalon Memory Mapped Slave	clk	0x01104000	0x01107fff	
<input checked="" type="checkbox"/>		jtag_uart_0	JTAG UART				
		avalon_jtag_slave	Avalon Memory Mapped Slave	clk	0x01109020	0x01109027	
<input checked="" type="checkbox"/>		switches	PIO (Parallel I/O)				
		s1	Avalon Memory Mapped Slave	clk	0x01109000	0x0110900f	
<input checked="" type="checkbox"/>		leds	PIO (Parallel I/O)				
		s1	Avalon Memory Mapped Slave	clk	0x01109010	0x0110901f	
<input checked="" type="checkbox"/>		sysid	System ID Peripheral				
		control_slave	Avalon Memory Mapped Slave	clk	0x01109028	0x0110902f	
<input checked="" type="checkbox"/>		sram	SRAM/SSRAM Controller				
		avalon_sram_slave	Avalon Memory Mapped Slave	clk	0x01080000	0x010fffff	
<input checked="" type="checkbox"/>		sdram	SDRAM Controller				
		s1	Avalon Memory Mapped Slave	clk	0x00800000	0x00ffffff	

Dopo aver rilanciato "GENERATE" il blocco complessivo comprenderà, analogamente a quanto accaduto per la SRAM anche le linee per il collegamento con la memoria esterna. Si dovrà pertanto collegare queste ad opportuni PIN, e dare a questi ultimi un nome uguale a quello assegnato all'interno del file di vincoli per garantire il corretto collegamento.

NOTA BENE: mentre tutte le linee sono configurate come uscite, la linea dati deve essere configurata come bidirezionale (infatti normalmente in una memoria i dati si possono sia scrivere che leggere)



SDRAM-PLL

A differenza della SRAM che non è controllata da alcun clock, la SDRAM necessita un clock opportuno e la sincronizzazione di questo è particolarmente critica, si deve infatti riuscire a compensare i ritardi introdotti a livello di scheda dalle linee di collegamento tra l'FPGA e la SDRAM. Per fare questo, torna utile l'impiego di un elemento specifico per la ri-sincronizzazione dei segnali, una PLL (Phase Lock Loop).

All'interno di Quartus:

Tools > Mega Wizard Plugin Manager
(Create a New Custom Megafunction)

Installed Plugins > I/O > ALTPLL

Si dia un nome opportuno (ad esempio SDRAM_PLL)

Si scelga il linguaggio di descrizione (Verilog HDL) e la famiglia di FPGA (Ciclone II).

Si configuri opportunamente il blocco

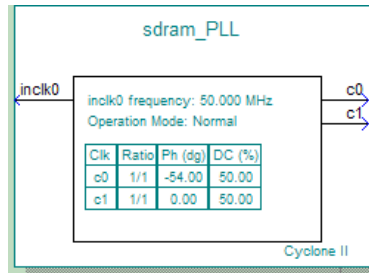
Parameter Settings|General/Modes – Si imposti il clock in ingresso a 50 MHz

Parameter Settings|Inputs/Locks – Si disattivi "create pllenna" e "create locked output"

Output clocks |clk0 – si attivi "use this clock" e si fissi *clock phase shift* = -3 ns

Output clocks |clk1 – si attivi "use this clock"

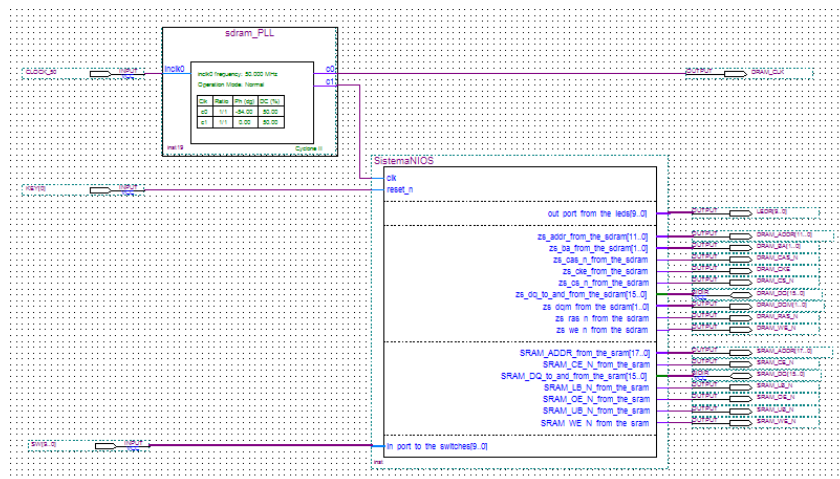
Alla fine il blocco deve assumere questa forma:



A questo punto si può generare il blocco e lo si farà entrare a far parte del progetto in essere.

SCHEMA finale

La PLL così generata fornirà i clock tanto per il processore quanto per la SDRAM. Lo schema finale da realizzare, comprensivo di tutti i pin sarà il seguente:



Si compili il progetto complessivo e si effettui il download sulla DE1

Sviluppo software

Con questo nuovo sistema, la memoria a disposizione è di molto aumentata rispetto il sistema precedente, si può pertanto provare a far girare, utilizzando “Altera Monitor Program” anche programmi che richiedano maggior risorse in termini di memoria, come ad esempio

```
#include <stdio.h>
int main()
{printf("CIAO/n");}
```

L’uscita testuale, indirizzata di default STDIO ed avendo provveduto a reindirizzare quest’ultimo su UART-JTAG (in fase di configurazione del progetto all’interno di Altera Monitor Program), viene visualizzata sulla finestra “TERMINAL” del tool predetto.