Seconda Università
degli Studi di Napoli

Altera University
Program

A Watermark-based Solution for Digital Authentication

Team Advisor:
Prof. Gianmarco Romano

Team members:
Francesco Pizzo
Luigi di Grazia
Anna Di Benedetto
Francesco Ledda
Raffaele Giulio Sperandeo

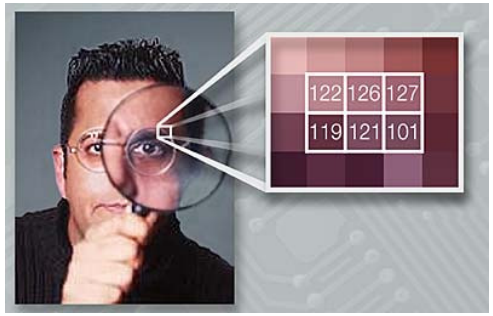
1 Dicembre 2011
Villa Trivulzio – Omate di Agrate Brianza (MB)

Sommario

- Intro Watermarking
- Obiettivi
- Un'idea semplice (LSB algorithm)
- Algoritmo migliorato
- Esempio immagini e audio
- Versioni HW e SW realizzate
- Impercettibilità
- Prestazioni
- Punti di forza
- Conclusioni

Intro Watermarking

"***Watermarking*** *is the process of embedding information into a digital signal* "



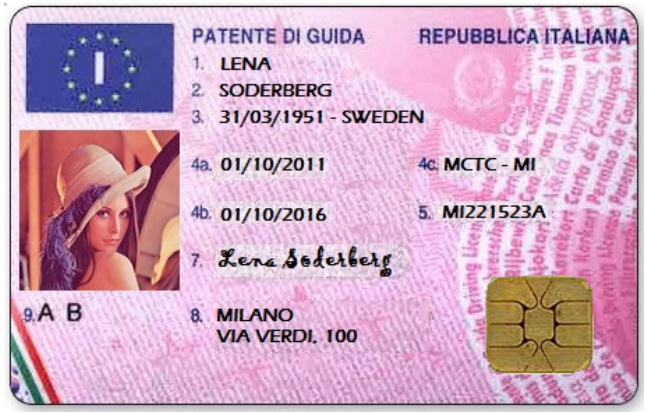
"***Steganography*** *is the art and science of writing hidden messages in such a way that no one suspects the existence of it*"

Possibili campi d'applicazione:

- ***Digital Rights Managment (DRM) –*** *gestione del copyright, identificazione, protezione e tracciamento dei contenuti*
- ***Digital Authentication –*** *verifica delle autorizzazioni per l'accesso ai file, identificazione dell'owner, firme digitali*
- ***Covert Communications –*** *usate in realtà diverse, a partire da quelle militari (intelligence) fino alla diffusione di informazioni nei paesi a forte censura.*

Obiettivi

- Sviluppare un sistema basato su watermarking steganografico che permetta di nascondere (e recuperare) informazioni all'interno di immagini e file audio
  - Sfruttare la flessibilità dei dispositivi FPGA per migliorare le prestazioni ed ottenere un sistema embedded ma allo stesso tempo facilmente riprogrammabile
- Realizzare (ove possibile) custom peripherals per migliorare ulteriormente le prestazioni del sistema



Lo scopo è realizzare una o più applicazioni in grado di operare in campi eterogenei, nascondendo informazioni in immagini e audio

Algoritmo LSB

Un'idea semplice:

Inserimento dell'informazione nel bit meno significativo di ogni pixel/campione

ESEMPIO 1

11000111	00010100	11111111	01001010
01000100	10111001	00101000	11010010
10101010	01011011	00101011	11011001
11100011	11111111	11010010	00101010

Immagine "contenitore", matrice 4x4 pixel

Profondità : 8 bit/pixel

Si desidera nascondere:

W=10011010

1100011**0**	0001010**1**	11111111	01001010
0100010**1**	1011100**0**	00101000	11010010
1010101**0**	0101101**0**	00101011	11011001
1110001**1**	1111111**1**	11010010	00101010

Inserimento per colonne

Si comincia dal bit meno significativo del watermark

Inserimento di una stringa di testo in un'immagine

Come riuscire a comprendere in fase di estrazione dove termina la stringa?

Supporto ai soli caratteri della tabella ASCII STANDARD codificati su 8 bit, e utilizzo di un "carattere tappo" (11111111) che non codifica alcun carattere standard.

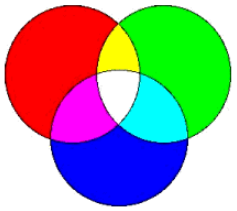
Occupazione Memoria

(Nc+1)*8 bit

Nc = numero di caratteri della stringa

Dimensioni minime immagine contenitore

24 bit = 8 bit + 8 bit + 8 bit
red green blue



Utilizzeremo come contenitore immagini bitmap in formato RGB a 24 bit, per cui:

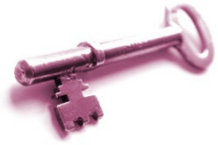
Almeno (Nc+1)*8/3 pixel

Utilizzo di una chiave di cifratura (1)

Necessità di robustezza: è facile risalire al watermark se esso è inserito in maniera sequenziale

Ordinamento *pseudocasuale* per l'inserimento dei bit del watermark

Generatore di numeri random (e.g. rand in C)



Il seme (condizione iniziale) del generatore diventa la nostra chiave di cifratura

Solo conoscendo la chiave è possibile risalire alla sequenza corretta dei bit del watermark!

Problema: generare due volte lo stesso indice porterebbe a sovrascrivere bit di informazione!

Soluzione: implementazione della funzione randperm

•Permutazione pseudocasuale dei possibili indici, senza ripetizioni

Utilizzo di una chiave di cifratura (2)

ESEMPIO 2

Stesso watermark e stessa immagine dell'Esempio 1

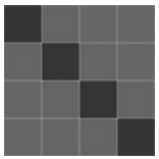
Invece di inserire i bit per colonna, utilizziamo la funzione randperm per calcolare l'indice.

Matrice 4x4: 16 possibili indici **randperm(16)=[6 3 16 11 7 14 8 15 5 1 2 4 13 9 10 12]**

W=10011010

11000111	00010100	11111111	01001010
01000100	10111001	00101000	11010010
10101010	01011011	00101011	11011001
11100011	11111111	11010010	00101010

11000111	00010100	11111111	01001010
01000100	10111000	00101000	11010010
10101011	01011011	00101011	11011001
11100011	11111110	11010010	00101010



Le diverse permutazioni per un'immagine di soli 4x4 pixel ad una componente di colore sono

16! = 20.922.789.888.000

Per un'immagine di 800x600 pixel RGB tale valore cresce a dismisura...

Una chiave a 32 bit genera 4.294.967.296 combinazioni diverse

File audio

Stesso principio usato per l'**Image Watermarking**

Contenitori audio: files con estensione ".wav" di tipo RIFF (Resource Interchange File Format).

Differenze:

- Campioni codificati su un numero di bit pari ad un multiplo di 8

- Header a lunghezza fissa (44 byte)

Lettura del file header:
- Verifica della compatibilità del formato
- Estrazione di informazioni utili per l'elaborazione (dimensione bitmap, profondità, numero di canali e codifica campioni audio...)

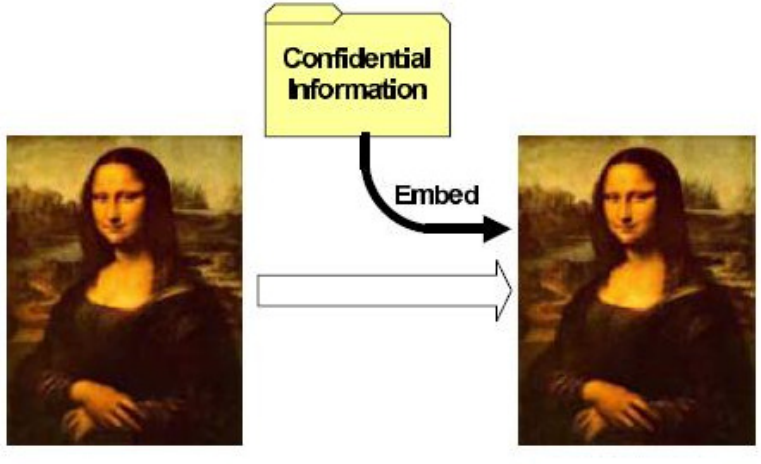
L'algoritmo LSB permette di utilizzare come contenitore svariati formati di file audio e immagini

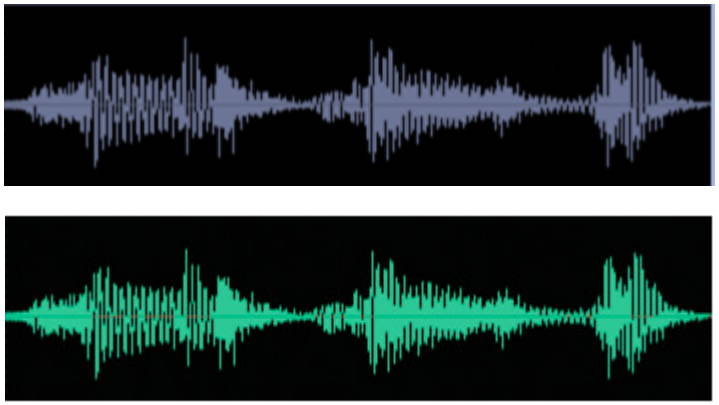
Non solo testo: il watermark può essere un file qualsiasi, col solo vincolo della dimensione

Impercettibilità

Commutazione del bit meno significativo → **Passaggio ad un livello di colore R, G o B (o a un livello audio) adiacente**

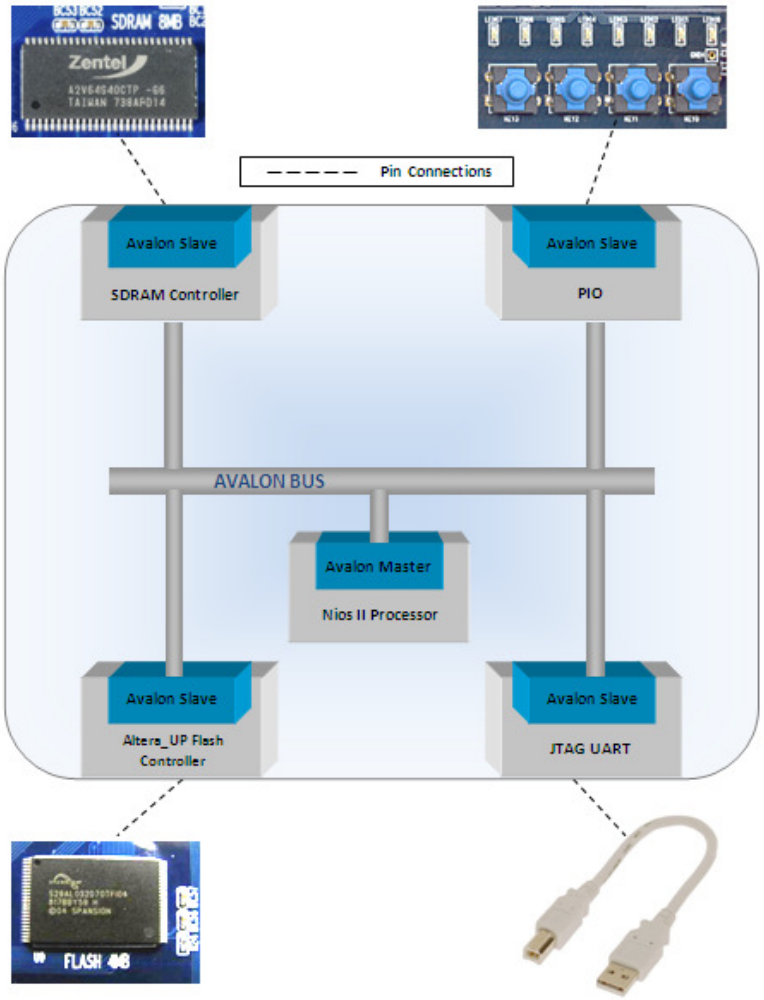
Incapacità per l'occhio e l'orecchio umano di rilevare le differenze tra file originale e file watermarkato!





Maggiore profondità = Maggiore risoluzione = Maggiore impercettibilità

Architettura Hardware del sistema



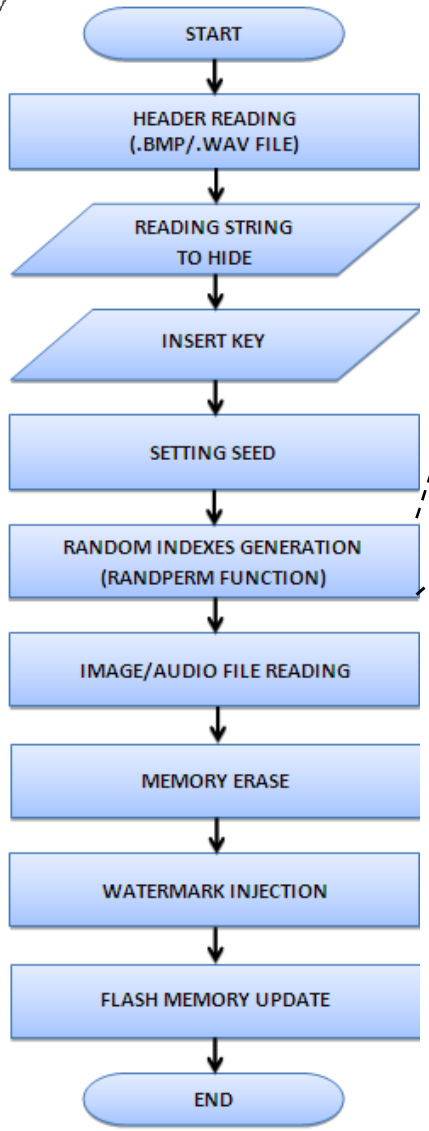
Architettura di base composta da:

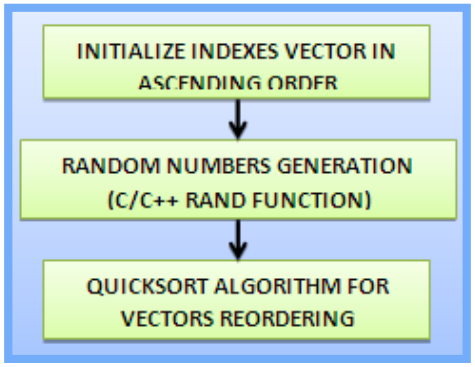
- Soft Processor **Nios-II** versione standard;
- **SDRAM** come memoria per l'elaborazione;
- **Flash memory**, come memoria di massa, accessibile tramite **Altera_UP controller;**
- **Controller JTAG** per la comunicazione con l'host PC;
- **Pio** collegate ai led della Board per scandire le fasi dell'elaborazione;
- Componenti collegati tramite l'**Avalon Bus**.

*Composizione del sistema tramite l'utilizzo di **Qsys**.*

Watermarking & Dewatermarking Flow-Charts

Versione Software Base



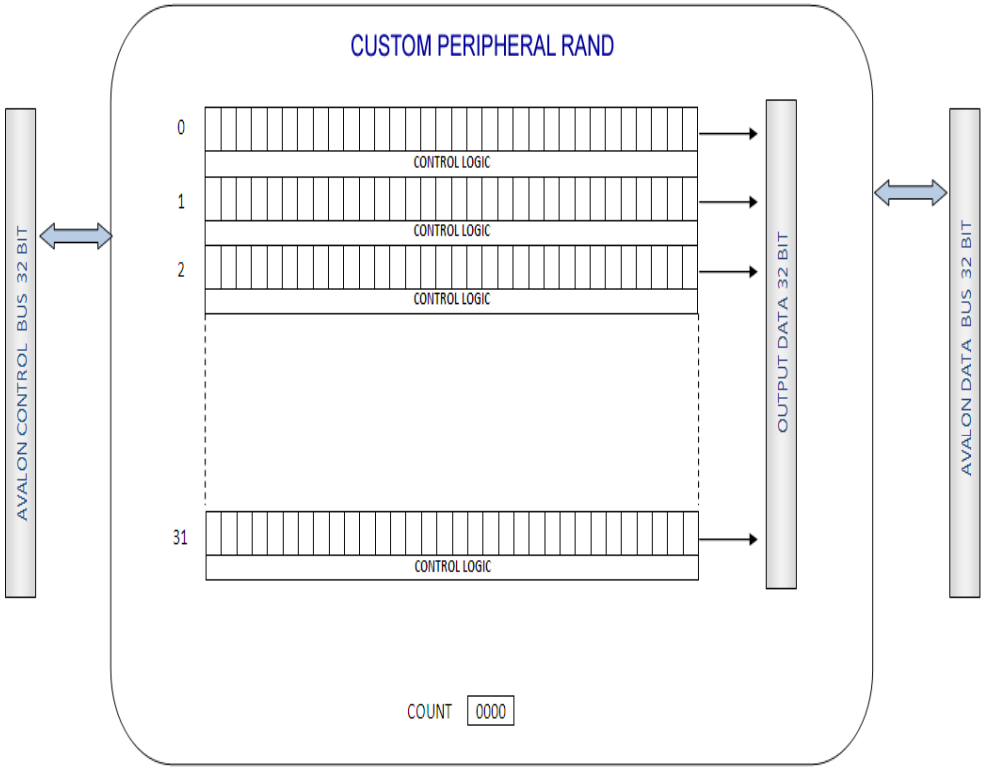


AMBIENTE SVILUPPO SOFTWARE «NIOS II SBT» (Eclipse)

- **Lettura dell' header file** per l'estrapolazione di dati relativi al file stesso.
- **Acquisizione Key** per il settaggio del *seme* nel *generatore di numeri casuali.*
- **Funzione «randperm»** per la generazione *pseudo-casuale* di una *permutazione di interi.*

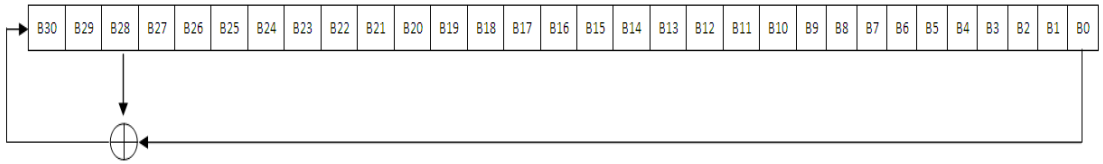
La principale differenza col diagramma di flusso del dewatermarking, è che la stringa non viene acquisita ma estratta.

Custom Peripheral - Rand

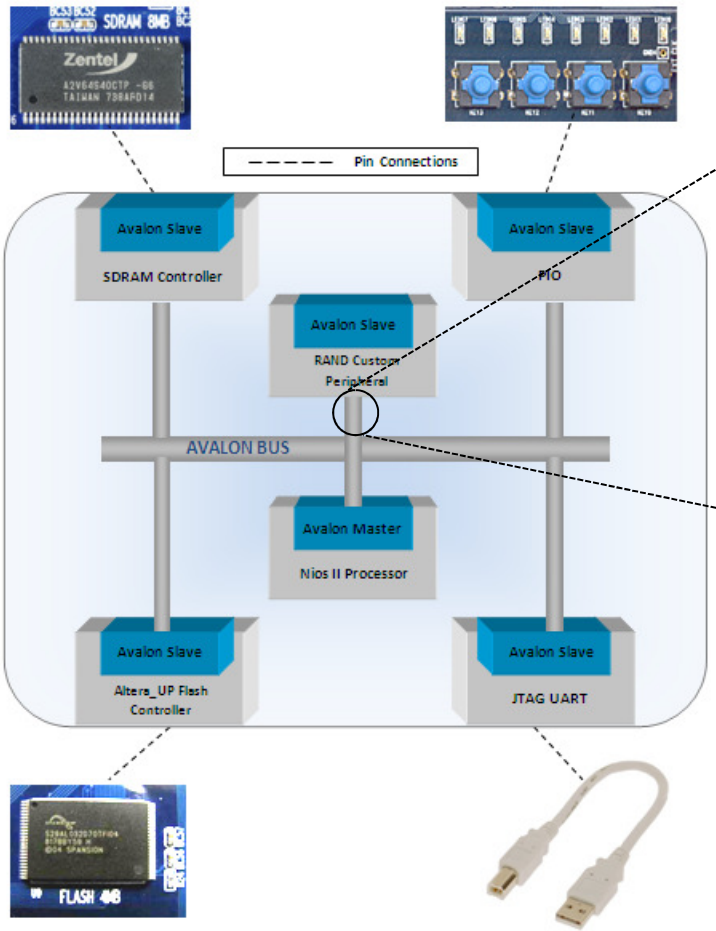


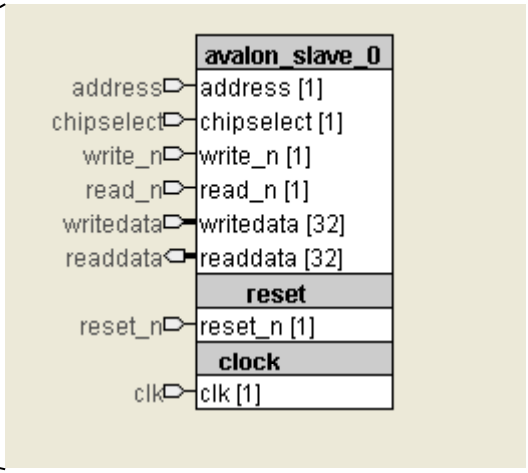
- Generatore Hardware di interi a 32 bit;
- Costituito da 32 PRBS in parallelo, ciascuno a 31 bit con polinomio generatore $x^{31}+x^{28}+1$;
- Scritta in VHDL;
- Migliori prestazioni rispetto alla rand del C;
- Periodo di $2^{31}-1$ interi;
- Riutilizzabile in altri progetti.

J-th PRBS



Interfacciamento della CP con il sistema (1)

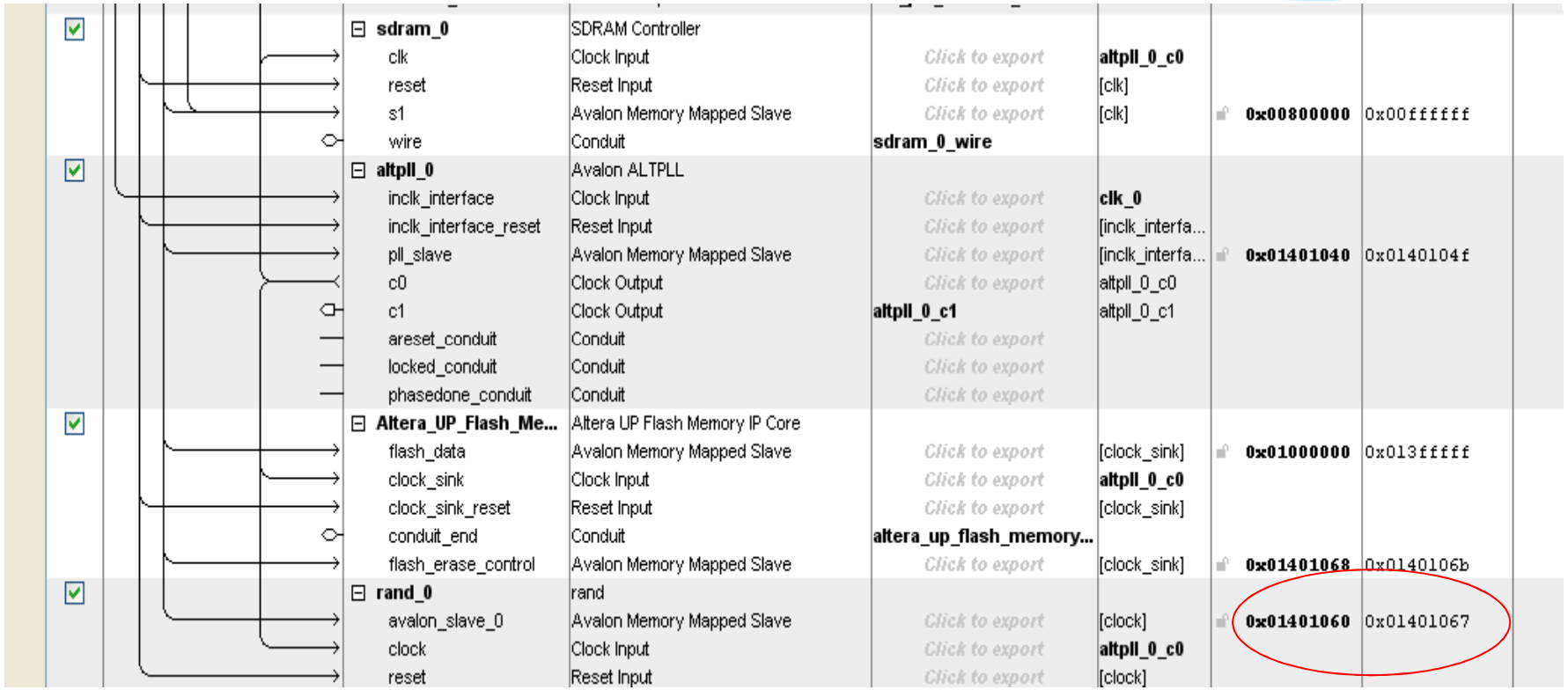




La CP si interfaccia col sistema tramite Bus dati di ingresso e di uscita a 32 bit e tramite segnali di controllo dell'Avalon BUS.

La periferica è accessibile in modalità Memory-Mapped.

Interfacciamento della CP con il sistema (2)

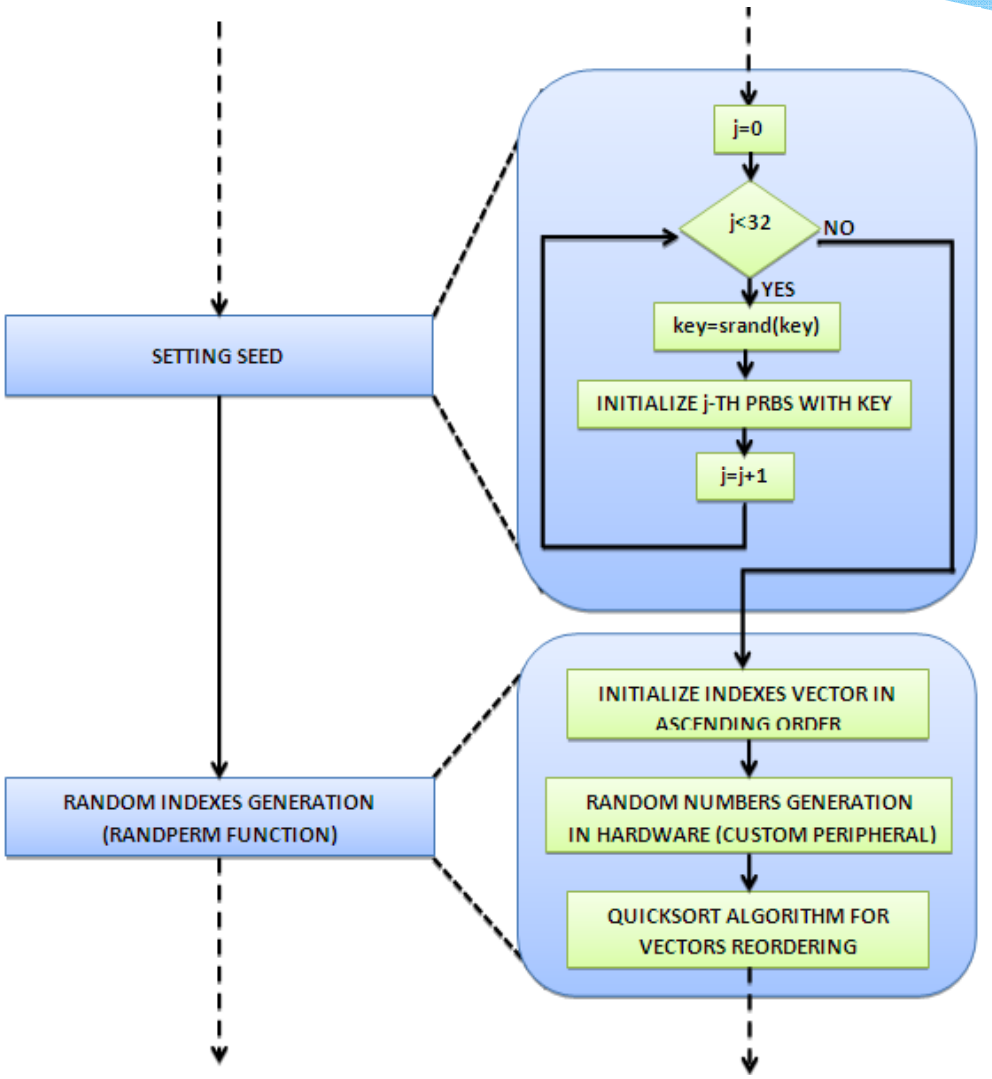


sdram_0	SDRAM Controller					
clk	Clock Input	Click to export	**altpll_0_c0**			
reset	Reset Input	Click to export	[clk]			
s1	Avalon Memory Mapped Slave	Click to export	[clk]	**0x00800000**	0x00ffffff	
wire	Conduit	**sdram_0_wire**				
altpll_0	Avalon ALTPLL					
inclk_interface	Clock Input	Click to export	**clk_0**			
inclk_interface_reset	Reset Input	Click to export	[inclk_interfa...			
pll_slave	Avalon Memory Mapped Slave	Click to export	[inclk_interfa...	**0x01401040**	0x0140104f	
c0	Clock Output	Click to export	altpll_0_c0			
c1	Clock Output	**altpll_0_c1**	altpll_0_c1			
areset_conduit	Conduit	Click to export				
locked_conduit	Conduit	Click to export				
phasedone_conduit	Conduit	Click to export				
Altera_UP_Flash_Me...	Altera UP Flash Memory IP Core					
flash_data	Avalon Memory Mapped Slave	Click to export	[clock_sink]	**0x01000000**	0x013fffff	
clock_sink	Clock Input	Click to export	**altpll_0_c0**			
clock_sink_reset	Reset Input	Click to export	[clock_sink]			
conduit_end	Conduit	**altera_up_flash_memory...**				
flash_erase_control	Avalon Memory Mapped Slave	Click to export	[clock_sink]	**0x01401068**	0x0140106b	
rand_0	rand					
avalon_slave_0	Avalon Memory Mapped Slave	Click to export	[clock]	**0x01401060**	0x01401067	
clock	Clock Input	Click to export	**altpll_0_c0**			
reset	Reset Input	Click to export	[clock]			

Tramite Qsys possiamo assegnare gli indirizzi utilizzati per accedere alla CP.

Watermarking & Dewatermarking Flow-Charts

Versione Software con Custom Peripheral



«SETTING SEED»

Inizializzazione dei 32 PRBS della CP con 32 diverse chiavi pseudo-casuali ottenute da quella acquisita da input.

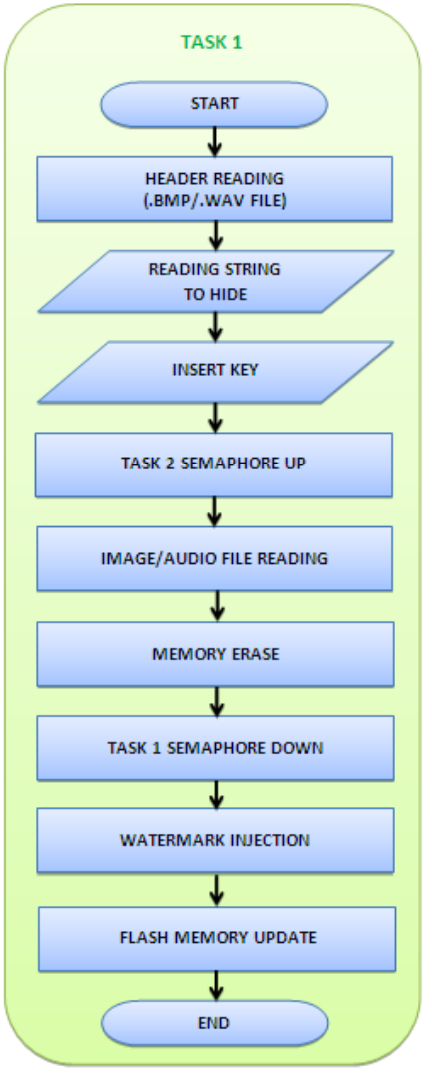
«RANDPERM FUNCTION»

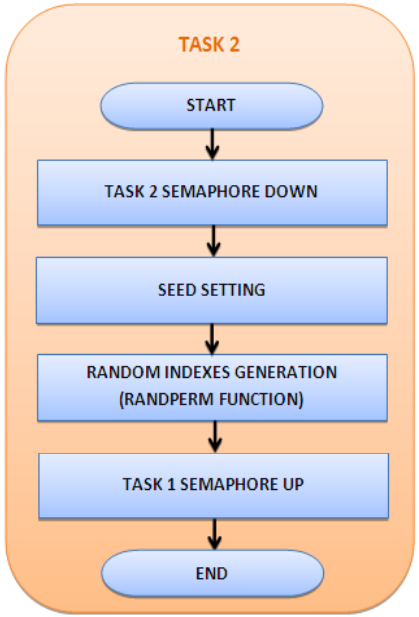
Generazione dei numeri casuali con lettura in memoria all'indirizzo della CP tramite puntatori.

Per il dewatermarking valgono analoghe considerazioni.

Watermarking & Dewatermarking Flow-Charts

Versione Software con Multithreading





Multithreading (RTOS «MicroC/OS-II») per la suddivisione dell'elaborazione in due *tasks*.

- ***Task 1:*** principali operazioni dell'algoritmo

- ***Task 2:*** generazione dei numeri casuali

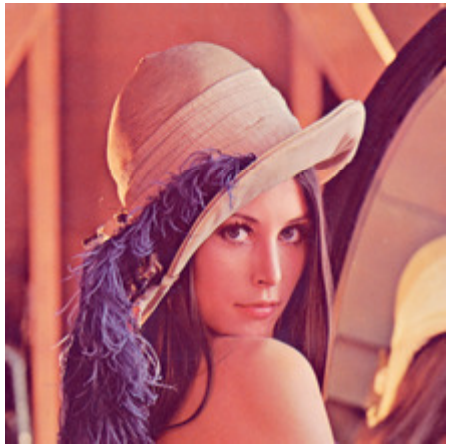
- Utilizzo di **semafori** per la *sincronizzazione* e per la presenza di *strutture dati condivise* tra i tasks

I flow-chart dell'algoritmo di dewatermarking con multi-threading sono simili a quelli dell'operazione di watermarking.

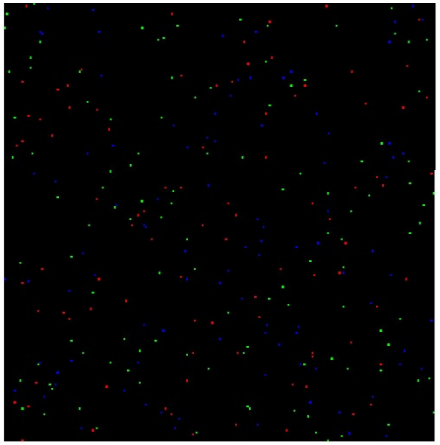
Video


A Watermark-based
Solution for Digital
Authentication
(IT003)
00:02

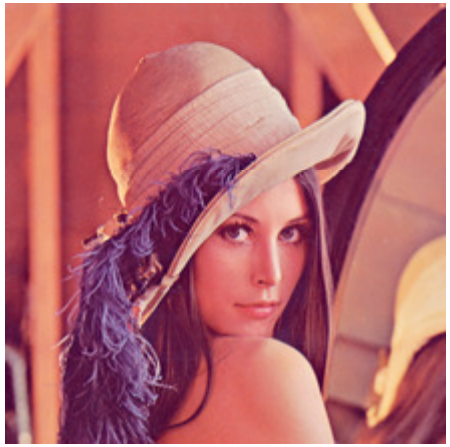
Esempio – Stringa inserita:
"Nome: Lenna; Cognome: Soderberg; Nata il: 31/03/1951; Nazionalità: Svedese; Indirizzo: Via Verdi 100, Milano; ID code:IT003-adgyfg3432g11"



Originale



Pixel modificati



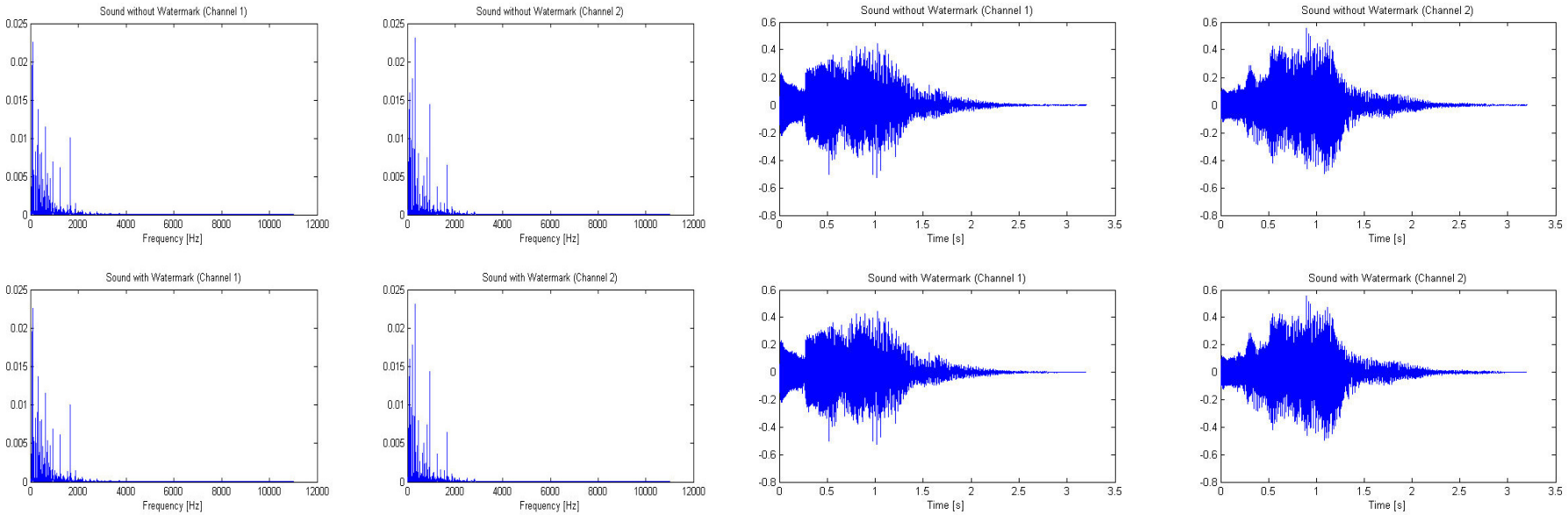
Watermarked

L'*impercettibilità visiva* del watermark è stata analizzata ricorrendo a strumenti messi a disposizione dall'ambiente Matlab.

Prestazioni del sistema

File audio

Nel caso di file audio l'impercettibilità è ancora più evidente.
I campioni possono essere rappresentati anche su stringhe formate da multipli di 8 bit.



Spettro e forme d'onda relativi ad un breve suono della durata di circa 3 secondi, con profondità di 16 bit per campione a due canali, campionato alla velocità di 22kHz.

Confronto tra le diverse implementazioni

Le diverse versioni puntano a migliorare le prestazioni in termini di tempi di esecuzione con i seguenti risultati (image watermarking):

Version	# of clocks	Time [s]
watermark_0x1	67730	67,73
watermarkCP_0x2	62497	62,497
MTwatermark_0x1	47197	47,197
MTwatermark_0x2	47128	47,128

Version	# of clocks	Time [s]
dewatermark_0x1	22415	22,415
dewatermarkCP_0x1	16628	16,628
MTdewatermark_0x1	23959	23,959
MTdewatermark_0x2	17899	17,899

Version	Multi-threading	Custom Peripheral
watermark_0x1	NO	NO
watermarkCP_0x2	NO	YES
MTwatermark_0x1	YES	NO
MTwatermark_0x2	YES	YES
dewatermark_0x1	NO	NO
dewatermarkCP_0x1	NO	YES
MTdewatermark_0x1	YES	NO
MTdewatermark_0x2	YES	YES

- L'introduzione di una Custom Peripheral fornisce una accelerazione hardware.
- Il multi-threading, in alternativa alla CP, permette un miglioramento esclusivamente nel caso dell'operazione di watermarking.
- Il dewatermarking presenta significative accelerazioni solo ricorrendo alla CP.

Punti di forza del sistema

- **Parallelismo** (sia hardware che software)
- **Efficienza dell'algoritmo** utilizzato
- **Modularità ed estensibilità**
- **Sicurezza dell'algoritmo** rispetto ad attacchi a forza bruta
- **Autonomia**

Vantaggi forniti dal Nios II:

- ***Customizzabilità***
- ***Tool dedicati*** *come Qsys per lo sviluppo di SOPC complessi*
- *Semplicità di utilizzo dell'****Avalon Bus***
- ***RTOS*** *eseguibile su NIOS*
- *Ambienti di sviluppo (****Quartus II*** *e* ***Nios II SBT****) supportati da dettagliata documentazione*

La riprogrammabilità dell'hardware via software rende l'FPGA un dispositivo ideale per situazioni in cui è necessario realizzare il prototipo di un sistema, aggiornarne uno esistente o ottimizzarne la realizzazione.

Seconda Università
degli Studi di Napoli

Altera University
Program

Grazie per la cortese attenzione

1 Dicembre 2011
Villa Trivulzio – Omate di Agrate Brianza (MB)