UIC – Università Italiana Cracking

http://quequero.org

Generatore hardware retroazionato di bit casuali

written by LittleLuk

UIC New Year Pack 2 – 01/Jan/2007

Generatore hardware retroazionato di bit casuali

Ettore Majorana: **Tutto ciò che si riesce a capire è banale**.		**Io non ho particolari doti... sono solo appassionatamente curioso.** (Albert Einstein)
	Per questo pregai e mi fu elargita la prudenza; implorai e venne in me lo spirito della sapienza. La preferii a scettri e a troni, stimai un nulla la ricchezza al suo confronto; non la paragonai neppure a una gemma inestimabile, perché tutto l'oro al suo confronto è un po' di sabbia e come fango sarà valutato di fronte ad essa l'argento. **L'amai più della salute e della bellezza, preferii il suo possesso alla stessa luce, perché non tramonta lo splendore che ne promana.** **...** **Senza frode imparai e senza invidia io dono, non nascondo le sue ricchezze.** Sapienza 7, 7:13	
Inizia con il piccolo ed il semplice per conseguire il grande ed il complesso. Tutto ciò che è complesso ha inizio da ciò che è semplice. Tutto ciò che è grande si produce da ciò che è piccolo. Tao-Teh-Ching,63		**Ed ecco mi apparve un cavallo bianco e colui che lo cavalcava aveva un arco, gli fu data una corona e poi egli uscì vittorioso per vincere ancora**. Apocalisse 6,2

Indice

0 Prefazione

Gestire la riservatezza di informazioni può essere un aspetto critico e complicato da gestire; per evitare che persone non autorizzate ne entrino in possesso solitamente si ricorre all'utilizzo di algoritmi crittografici la cui forza risiede nel fatto che, utilizzando risorse e tempi limitati, la mancata conoscenza di alcuni parametri renda quasi impossibile e comunque molto arduo l' ottenimento delle informazioni.

I punti fondamentali per garantire una buona riservatezza sono la possibilità di ottenere parametri robusti, non facilmente predicibili e tutti diversi l' uno dall' altro.
Questi parametri vanno conservati con modalità tali da escluderne l' utilizzo da parte di persone non autorizzate.

L’obiettivo di questo documento è la costruzione di un generatore hardware retroazionato di numeri casuali mentre non tratterò gli aspetti relativi alla conservazione della chiavi.

1 Test statistici

Per cominciare ecco un po' di teoria, si tratta di concetti che verranno utilizzati in seguito quindi vi consiglio di perdere un po' di tempo e leggerli.

1.1 chi^2

In tutte quelle situazioni in cui le osservazioni possono ricadere dentro un numero finito di categorie è possibile utilizzare test del chi^2 o **χ^2 test** per verificare se l' osservazione segue un andamento previsto.
Supponendo che la quantità N_i sia il numero di eventi osservati nell' i° bin, cioè una porzione in cui è suddiviso il dominio utilizzato e che n_i sia il numero aspettato in accordo con una qualche distribuzione prevista è possibile definire una variabile casuale chi quadro:

$$\chi^2 = \sum (N_i - n_i)^2 / n_i$$

dove la somma è effettuata su tutti i bin.
Se il numero di bin è grande o se il numero di eventi in ciascun bin è grande la funzione di probabilità del χ^2 risulta essere una buona approssimazione della distribuzione nota a cui appartengono gli ni.
Più in generale si indica la probabilità α che non venga superato il valore critico $\chi_{\alpha,g}^2$, nel caso di una variabile casuale chi quadro con g gradi di libertà dove per **grado di libertà** si intende un concetto introdotto in statistica da Ronald Fisher negli anni 1920 per esprimere il numero di dati effettivamente disponibili per valutare la quantità d'informazione contenuta nella statistica.
Infatti quando un dato non è indipendente, l'informazione che esso fornisce è già contenuta implicitamente negli altri: è possibile quindi calcolare le statistiche utilizzando soltanto il numero di osservazioni indipendenti.

α è detto anche livello di confidenza perché se il valore del χ^2 calcolato è minore del valore critico $\chi_{\alpha,g}^2$ con α e g opportuni significa che la distribuzione degli N_i non segue quella degli n_i con una probabilità inferiore a $(1 - \alpha)$.
I valori di $\chi_{\alpha,g}^2$ sono tabellati.

1.2 Nist Statistical Test Suite e RaBiGeTe

Per analizzare le sequenze casuali si possono utilizzare anche altri test. In particolare esiste una batteria di prove elaborata dal NIST (**N**ational **I**nstitute of **S**tandards and **T**echnology) detta Nist Statistical Test Suite che permette di analizzare una sequenza in 15 differenti modi.
Per avere risultati attendibili bisogna analizzare sequenze sufficientemente lunghe: ho scelto di analizzare quelle di lunghezza compresa tra i 0,1M e 10Mbit perché lo ritengo un buon compromesso tra i tempi necessari per acquisirle, elaborarle, la potenza di calcolo a mia disposizione e l' avere i risultati dei test significativi.
Per questo tipo di sequenze non posso utilizzare tutti i test dell' STS ma soltanto approximate entropy test, frequency test, cumulative sums (forward o reverse) test, discrete fourier transform (spectral) test, block frequency test, nonperiodic templates test, rank test, run test, serial test.

È possibile utilizzare anche **RaBiGeTe - Ra**ndom **Bi**t **Ge**nerators **Te**ster, un tool studiato appositamente per analizzare i generatori di bit casuali, scaricabile gratuitamente da http://www.webalice.it/cristiano.pi/RaBiGeTe/ che implementa numerose versioni di test parametrizzati e adattati per lavorare con i bit anziché su byte o word di lunghezza maggiore a 6.

2 Generatore di numeri casuali

CASCINE (Cascinata Applicata alla **Sicurezza** e **Cifratura** di **Informazioni Non Elementare)**, il generatore di numeri casuali che ho realizzato è un dispositivo hardware che utilizza il rumore bianco per produrre numeri casuali; si tratta di un circuito formato da due parti mostrate in figura 2.0. La prima è quella che forma la catena diretta e si occupa di produrre una sequenza di numeri casuali partendo dalla tensione di rumore termico prelevata da un ponte resistivo e da un firmware caricato su un microcontrollore PIC16F877A che ha il compito di analizzare la bontà dei numeri casuali e gestire l' interfaccia del generatore per permetterne il collegamento con qualsiasi dispositivo che supporti lo standard RS232. Inoltre il firmware pilota anche la seconda parte: una retroazione che ha il compito di massimizzare la velocità di generazione dei numeri casuali e di migliorarne le proprietà statistiche.

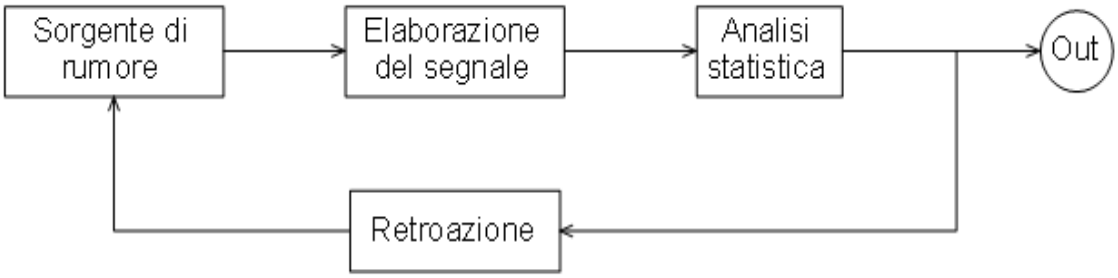

Figura 2.0: Schema generale di CASCINE

2.1 Il generatore di numeri casuali

La figura 2.1 è un esempio di CASCINE in funzione mentre trasmette la sequenza random generata al portatile tramite interfaccia seriale, è possibile vedere anche la demoboard della Microchip utilizzata per interfacciare il pc con il microcontrollore e per programmarlo.

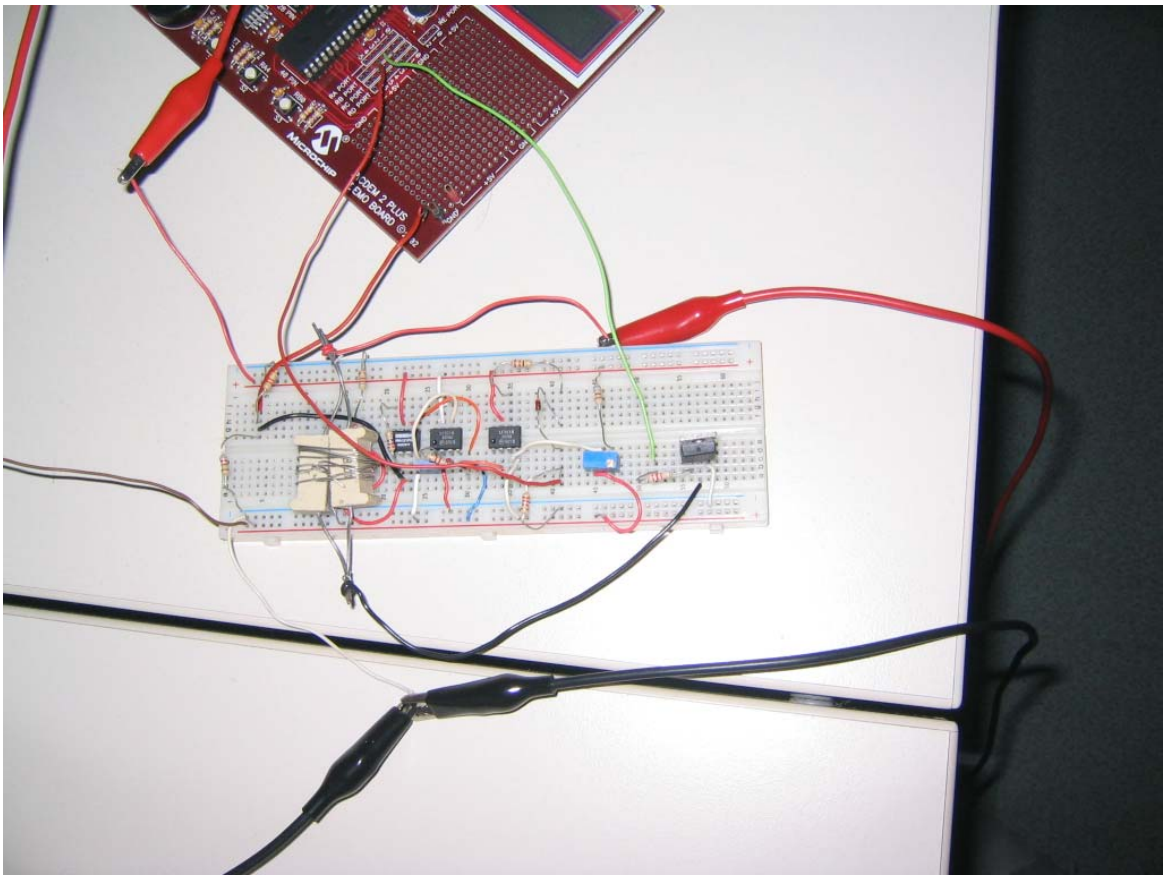

Figura 2.1: CASCINE in funzione

2.2 Suddivisione funzionale del dispositivo

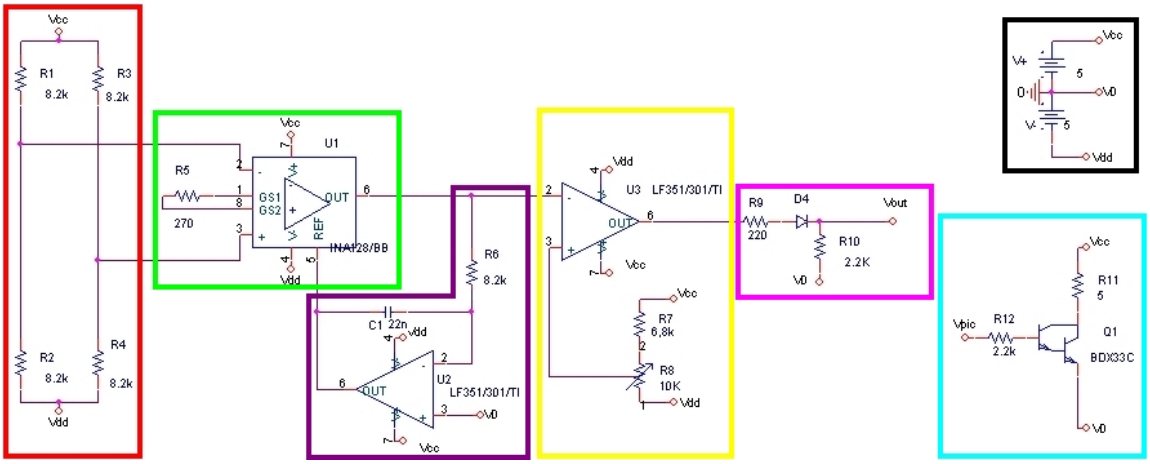

Figura 2.2: Schema elettrico dell'hardware del generatore.

2.2.1 Alimentazione

Il circuito è alimentato da due alimentatori stabilizzati collegati come mostrato nel blocco nero in grado di fornire una tensione duale V_{cc}= 5V, V_{dd}= -5V, il riferimento V_0 e una corrente di circa 2A. Non ho previsto nessun altro tipo di filtraggio e stabilizzazione delle tensioni di alimentazioni.
Ho scelto questa soluzione perché così posso avere un circuito duale evitando i problemi legati alla asimmetria, in particolare degli amplificatori e i segnali possono variare in un range di ampiezza=
5 - (-5)= 10V permettendo così di apprezzare un numero maggiore di valori dei segnali.
In realtà a causa della non perfetta linearità degli operazionali il range è di 4 - (-4)= 8V circa.
L' elevata corrente che i generatori devono essere in grado di fornire viene assorbita quasi esclusivamente dal circuito di retroazione.

2.2.2 Sorgente di rumore bianco

La sorgente di rumore bianco è il blocco rosso mostrato in figura 2.2 ed è formata da quattro resistori di 8,2 kΩ ± 5% da ¼ W collegati in una configurazione a ponte alimentato in modo duale. Anche in questo caso la dualità permette di avere un range di lavoro più ampio e una maggiore simmetria.
La struttura a ponte, a differenza di una sorgente formata da una sola resistenza, permette di utilizzare amplificatori con ingressi **rail-to-rail**.

Il dispositivo utilizza la tensione differenziale prelevata dai morsetti centrali del ponte: nel caso ideale tale tensione vale 0 in quanto R1*R4 = R2*R3[1].

1 Facilmente verificabile con Thevenin o sovrapponendo gli effetti

In realtà il resistore reale è formato da un resistore ideale e da un generatore di tensione che si fa carico della tensione di rumore V_n collegato in serie dove V_n soddisfa la relazione

$$V_n^2 = 4KT\,\mathrm{B}\,R$$

e T è la temperatura assoluta all' equilibrio termico, R è i il valore della resistenza ideale, B è la larghezza di banda in cui vengono effettuate le misure e K è la costante di Boltzmann $K \approx 1{,}380 \times 10^{-23} J/s$ ed è la conseguenza dell'agitazione termica dei portatori di carica in grado di muoversi liberamente. Il loro movimento è tale da creare ai capi del resistore una differenza di tensione il cui valore medio calcolato in un tempo sufficientemente lungo, è pari a 0 Volt ma che istantaneamente è altamente variabile e come tutti i rumori è descrivibile solo in termini statistici.

La tolleranza del 5% sul valore delle resistenze non è un grande problema perché ciò si traduce in una tensione differenziale che è funzione della tensione di rumore delle resistenze e della tensione di alimentazione. Quest' ultima è una componente costante dovuta allo sbilanciamento del ponte ed è facilmente eliminabile inserendo a valle un filtro passa alto.

Per i quattro resistori del ponte ho scelto il valore di 8,2 kΩ perché è un buon compromesso tra l' avere una tensione di rumore sufficientemente elevata ($Vn \approx 10^{-5} V$ con B= 1 MHz e T= 300 °K), avere consumi ridotti e evitare problemi di autoriscaldamento per effetto Joule.
Esistono sorgenti di rumore bianco migliori del rumore termico come ad esempio dispositivi che coinvolgono il decadimento radioattivo o fenomeni ottici ma ho scelto il ponte resistivo per i motivi spiegati nel precedente capitolo dedicato alla generazione dei numeri casuali; inoltre ciò mi permette di creare un **anello di retroazione semplicissimo ma efficace** in grado di bilanciare il ponte e spostare il punto di lavoro senza ricorrere a grandezze elettriche.

La tensione di rumore dei resistori V_n prelevata tramite il ponte è funzione della temperatura, quindi per un corretto funzionamento è necessario mantenere la temperatura del dispositivo costante e schermarlo il più possibile dai disturbi esterni come sorgenti radio e cellulari.

2.2.3 Amplificatore per strumentazione

Il blocco verde è l' amplificatore del rumore: ha il compito di amplificare la tensione di rumore dall' ordine di qualche microvolt a valori più facilmente misurabili. Si tratta di un blocco di particolare importanza in quanto se l' uscita non ricalca fedelmente l' ingresso si perdono tutte le caratteristiche di randomness associate al segnale prelevato dal ponte resistivo.
Per avere un amplificatore il più possibile lineare ho utilizzato un amplificatore per strumentazione.

Un amplificatore per strumentazione differisce da un classico amplificatore operazionale per via della sua realizzazione schematizzata in figura 2.3:

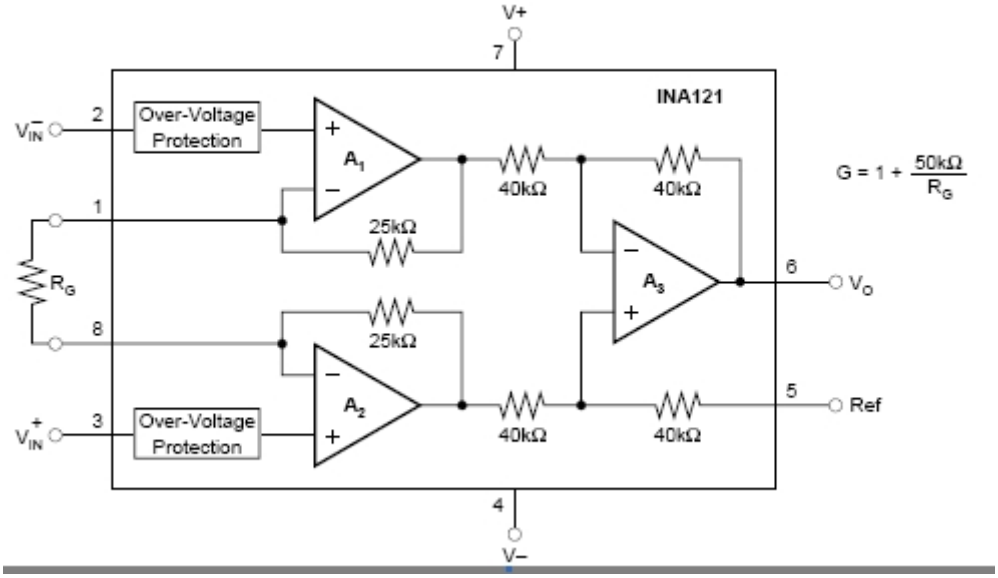

Figura 2.3: Schema generale di un amplificatore per strumentazione

Quando Ref, il pin di riferimento, è collegato alla tensione di riferimento è possibile dimostrare[2] che

$$Vout = (1 + 2 * 25\mathrm{K} / Rg)(V3 - V2)$$

Vout assomiglia alla tensione di uscita di un classico amplificatore differenziale ma a differenza di quest' ultimo con l' amplificatore per strumentazione è possibile lavorare con ingressi differenziali ciascuno dei quali vede la **stessa impedenza di ingresso**[3] **molto elevata** perché le resistenze di ingresso di A1 e A2 tendono all' infinito. La resistenza di uscita invece tende a zero per merito della retroazione negativa che abbassa ulteriormente la già piccola Zout dell' operazionale A2.
Il rapporto di reiezione di modo comune **CMRR tende all' infinito** perché il guadagno rispetto al valore medio delle tensioni V3 e V2 è circa nullo e dai datasheet si vede che è di almeno 100 dB.
Per via della doppia alimentazione è possibile avere in uscita un segnale di ampiezza massima compresa tra i ±10V circa e la **struttura rail-to-rail** permette alla tensione di modo comune in ingresso di assumere qualsiasi valore compreso tra le tensioni di alimentazione senza mandare in crisi gli stadi di ingresso.
Tutto ciò garantisce che i due morsetti del ponte resistivo da cui prelevo la tensione di rumore vedano un amplificatore che si comporta in modo sufficientemente lineare ottenendo così in uscita un segnale con le stesse caratteristiche di randomness del segnale in ingresso.
Non è però possibile ottenere in uscita una tensione che sfrutta l' intero range perché per via della piccola tensione di rumore in ingresso servirebbe un guadagno elevatissimo che per via del **GBW** limitato limiterebbe troppo la banda di lavoro dell' operazionale. Come compromesso ho fissato Rg= 270 Ω a cui corrisponde un guadagno

$$G = (1 + 2 * 25\mathrm{K} / Rg) = 190$$

Per tale guadagno la frequenza massima di lavoro[4] misurata sperimentalmente è di circa 0,1 MHz e Vout è dell' ordine dei mV sufficientemente elevata da poter essere trattata

2 Vedi http://www.etec.polimi.it/studenti/matdid/OpAmp/sld051.htm e seguenti
3 Deve essere A1=A2=A3
4 È la frequenza per cui il guadagno dell' amplificatore è pari al valore massimo diminuito di 3 dB

senza troppi problemi dagli stadi successivi ed è anche abbastanza robusta da resistere ad altri eventuali altri disturbi esterni.

Come amplificatore per strumentazione ne ho scelto uno integrato e non ho optato per una realizzazione homebrew per avere gli operazionali A1, A2, A3 e le resistenze il più possibili uguali tra loro e ho scelto l' INA121PA il cui datasheet è prelevabile dal sito www.alldatasheet.com .

2.2.4 Filtro

Per realizzare il mio dispositivo ho deciso di non collegare il pin Ref alla tensione di riferimento ma lo ho collegato ad una rete RC attiva evidenziata dal blocco viola realizzando così un amplificatore passa alto in grado di attenuare le basse amplificare e amplificare quelle alte.

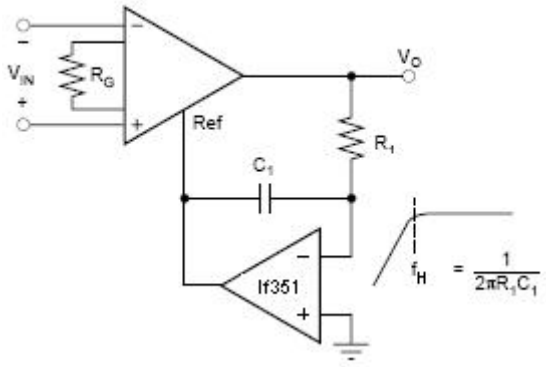

Figura 2.4: Amplificatore passa alto

Tale amplificatore, il cui schema elettrico è riportato in figura 2.4, ha la particolarità di comportarsi globalmente come un filtro passa alto ma è realizzato utilizzando R1 e C1 come filtri passa basso.
Il guadagno totale vale

$$Vo/Vin = G(f) = (1 + 2 * 25K/Rg) * jf/(fH + jf)$$

avendo scelto R1= R6= 8,2kΩ e C1=22 nF si ha che f_H= 820 Hz quindi complessivamente la rete si comporta come un filtro passa alto con frequenza di taglio inferiore pari a f_H .

La frequenza di rete ha un valore di 50 Hz è quindi logico pensare che il generatore di numeri random risenta di un disturbo a tale frequenza nonostante l' utilizzo di alimentatori stabilizzati ma la distanza da f_H è superiore ad una decade quindi tale disturbo viene attenuato almeno di 20 dB.
Il filtro ha anche il compito di eliminare la componente continua dovuta allo sbilanciamento del ponte a causa della elevata tolleranza delle resistenze e attenua il rumore 1/f importante solo alle basse frequenze.
Tutti i segnali con f > 820 Hz, invece, vengono amplificati di un valore praticamente costante pari a circa 190. Il limite della frequenza di lavoro superiore è imposto dal **GBW** e dallo **slew rate** dell' operazionale con prestazioni peggiori.

2.2.5 Comparatore

A questo punto il rumore amplificato e opportunamente filtrato viene inviato al blocco giallo che è un semplice convertitore A/D ad 1 bit con soglia regolabile.
L' operazionale è un classico LF351 che lavora ad anello aperto e la mancanza della retroazione negativa fa si che che l' uscita si porti sempre alla tensione di saturazione positiva o negativa quindi quando la tensione V_- è maggiore di V_+ l' uscita si pone circa a V_{dd} altrimenti è a V_{cc}, **si ha quindi un segnale digitale alternato a 2 livelli**.
La tensione V_- rappresenta la soglia del comparatore e può essere tarata agendo sul partitore resistivo che va regolata in modo da avere V_- a circa 0V perché così nel caso ideale si ha $V_6 = V_{cc}$ con una probabilità pari al 50% e $V_6 = V_{dd}$ sempre col 50% di probabilità.

2.2.6 Raddrizzatore segnale

Il segnale digitale a 2 livelli ottenuto dallo stadio precedente viene reso unidirezionale dal blocco fucsia. Questo passaggio è indispensabile per avere un segnale digitale compatibile con le specifiche TTL / HCMOS e quindi poterlo interfacciare con il microcontrollore non mostrato nello schema elettrico ma visibile in figura 2.1. Il collegamento è un semplice filo che collega V_{out} al pin RB5 del micro che è programmato come ingresso.
Questo blocco è formato da un diodo che quando la V_6 dello stadio precedente è a V_{dd} conduce portando la tensione V_{out} a[5]

$$V6 * R10 / (R9 + R10) \approx 0{,}9 * V6 \approx 4{,}5\,V$$

che viene riconosciuta dal PIC16F877A come un '1' logico.
In caso contrario il diodo non conduce perché è polarizzato inversamente e trascurando sempre la corrente assorbita dal pic si ha $Vout \approx 0$ che viene interpretata come uno '0' logico.

2.2.7 Retroazione

La Vout digitale e unidirezionale ottenuta dal rumore termico a questo punto viene inviata al microcontrollore PIC16F877A programmato con un firmware in grado di leggere la sequenza digitale e eseguire test statistici sui bit acquisiti per verificarne la randomness, se danno esito positivo la sequenza viene trasmessa al dispositivo che ne ha fatto richiesta tramite un' interfaccia RS232 altrimenti attiva il circuito di retroazione rappresentato del blocco azzurro che agisce sui parametri della sorgente di rumore.
Avendo scelto come sorgente casuale la tensione di rumore termico che è proporzionale alla temperatura assoluta a cui si trova il resistore **è possibile modificarne l' ampiezza senza modificare grandezze elettriche ma agendo esclusivamente sulla temperatura** mantenendo quindi costante il punto di lavoro elettrico.
Questa particolarità ha avuto un ruolo decisivo nella scelta della sorgente infatti in tutti gli altri casi avrei dovuto utilizzare come retroazioni dei circuiti derivati dallo schema del *controllo automatico di guadagno* ACG complicando inutilmente lo schema elettrico e utilizzando dispositivi FET che avrebbero introdotto un forte rumore shot. Inoltre si avrebbe una variazione significativa della corrente assorbita dalla sorgente di rumore rendendo probabilmente necessario dover ritarare la soglia del comparatore.

5 È possibile trascurare la piccola corrente assorbita dal microcontrollore e la caduta di tensione introdotta dal diodo.

Nel caso di retroazione termica, invece, l' unica variazione della corrente che fluisce nel dispositivo è dovuto al fatto che la resistenza varia modificando la temperatura ma la dipendenza è trascurabile infatti

$$R(t) = R0 * (1 + \alpha \Delta t)$$

dove R0 è il valore di resistenza assunto dal resistore per $\Delta t = 0$ e α è il coefficiente di temperatura che tipicamente è di circa 25 ppm/°C.

I dettagli sul funzionamento della retroazione sono riportati nell' apposita sezione: "*tecniche per migliorare la randomness"*.

2.2.8 RS232

Per interfacciare il dispositivo CASCINE con il mondo esterno ho scelto di utilizzare lo standard RS232 perché è molto noto ed è facile reperire informazioni sul funzionamento e ho potuto collegarlo al pc durante le fasi di debug e test senza problemi utilizzando la demoboard fornita dalla Microchip. Inoltre l' RS232 è molto diffuso quindi è possibile utilizzarlo con un numero molto vasto di apparecchiature.
L' intera gestione della comunicazione seriale, sia per spedire valori che per leggere i comandi, è demandata al microcontrollore PIC16F877A il cui firmware oltre a gestire la retroazione implementa un protocollo di comunicazione (non scritto da me) anche se risulta sovradimensionato per i miei scopi in quanto utilizzo solo i comandi per iniziare e fermare l' acquisizione di una sequenza random.

3 Il firmware

3.1 Il Pic

Il microcontrollore usato appartiene alla famiglia dei **PIC** (Peripheral Interface Controller); si tratta di componenti che integrano in un unico dispositivo tutti i circuiti necessari a realizzare sistemi digitali programmabili in quanto dispongono di:

- una **CPU** (Central Processor Unit), unità centrale di elaborazione il cui scopo è interpretare le istruzioni di programma.
- una **memoria FLASH** in cui vengono memorizzate in maniera permanente le istruzioni del programma da eseguire.
- una memoria **RAM** (Random Access Memory) a supporto delle variabili utilizzate dal programma.
- una serie di **linee di input/output**, linee di ingresso e uscita per pilotare dispositivi esterni o ricevere impulsi da sensori, pulsanti (per esempio).
- una serie di **dispositivi ausiliari** al funzionamento quali generatori di clock, bus, contatori

Il PIC utilizzato che ho utilizzato è un PIC16F877A. Ho scelto questo modello perché in rete è presente una ampia documentazione, ha caratteristiche che ben si adattano alle specifiche del progetto ed è possibile farselo inviare gratuitamente come sample dalla Microchip. Alternativamente si può utilizzare il PIC16F877 che è perfettamente compatibile col 16F877A.

3.2 Flowchart

In questa sezione riporto il diagramma di flusso che ha il compito di spiegare ad alto livello con un linguaggio like-C le operazioni effettuate dal firmware. L' intero source-code codato direttamente in assembler per pic è riportato nei file allegati.
Questo è la routine generale del programma che si occupa di eseguire tutte le operazioni richieste.

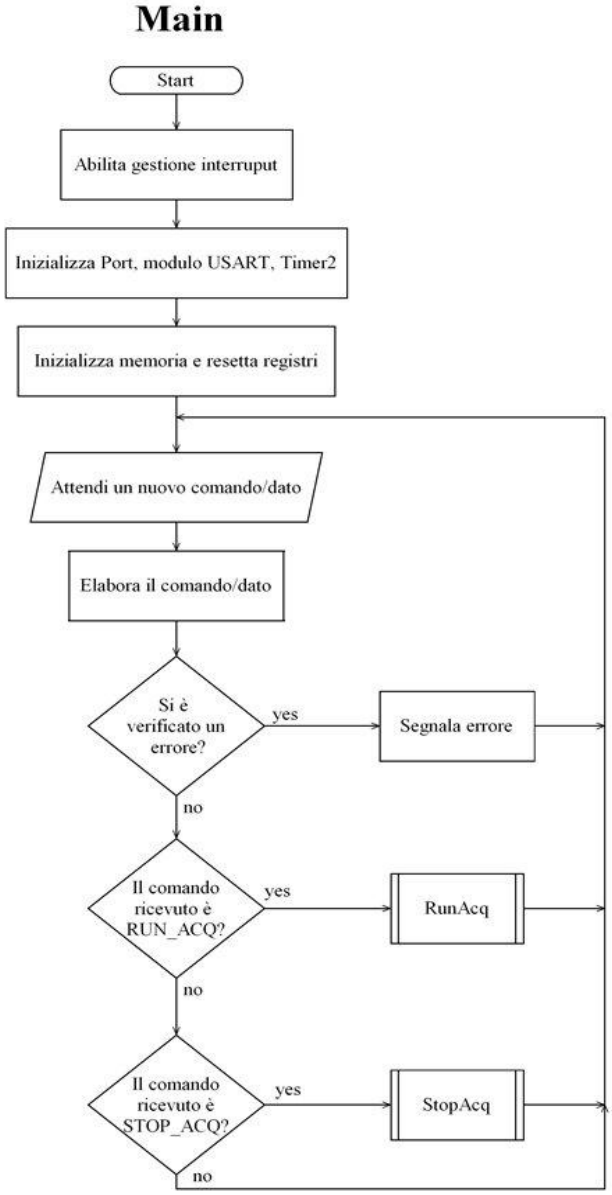

Grafico 3.1: Blocco principale del programma

Ecco la routine che si occupa di attivare l' acquisizione dei valori:

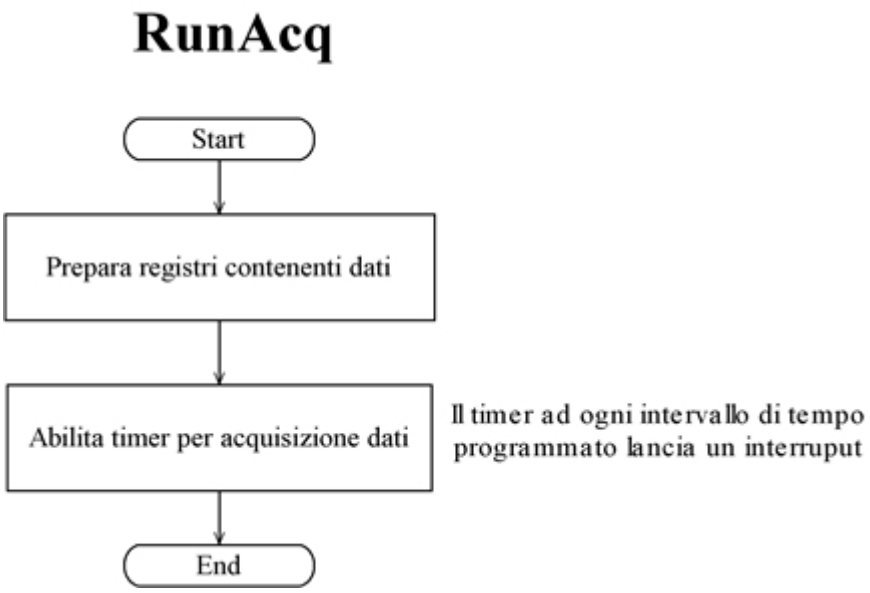

Grafico 3.2: Routine attivata alla ricezione del comando RUN_ACQ

e quella che la disabilita:

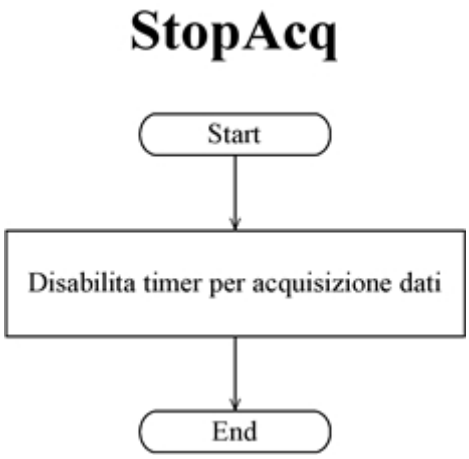

Grafico 3.3: Routine attivata alla ricezione del comando STOP_ACQ

Il firmware si occupa anche di gestire le interruput generate dal Pic e relativi moduli:

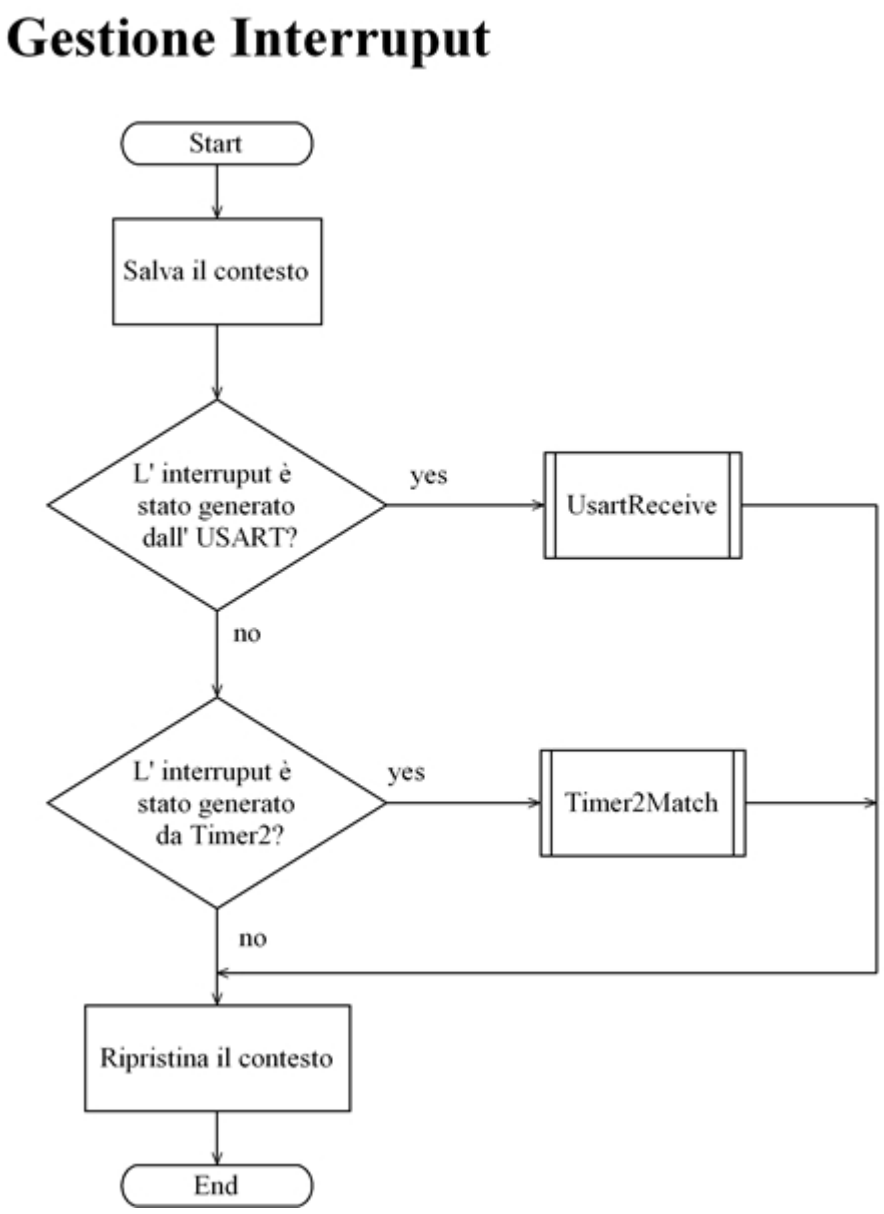

Grafico 3.4: Routine principale per la gestione delle interruput generate

La gestione delle interruput generate dall' USART avviene come indicato di seguito:

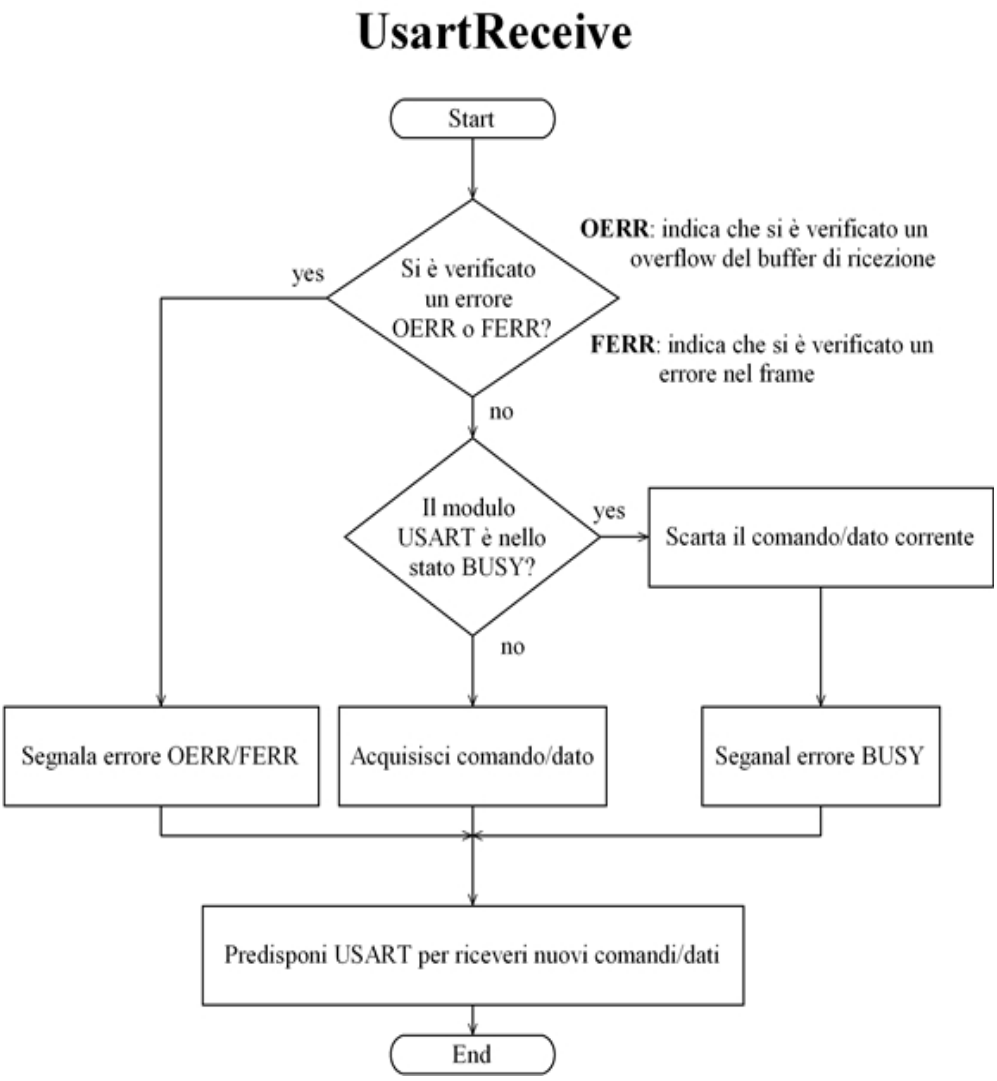

Grafico 3.5: Routine che gestisce l' interruput generata dal modulo USART

Ecco ora il core dell' applicazione:

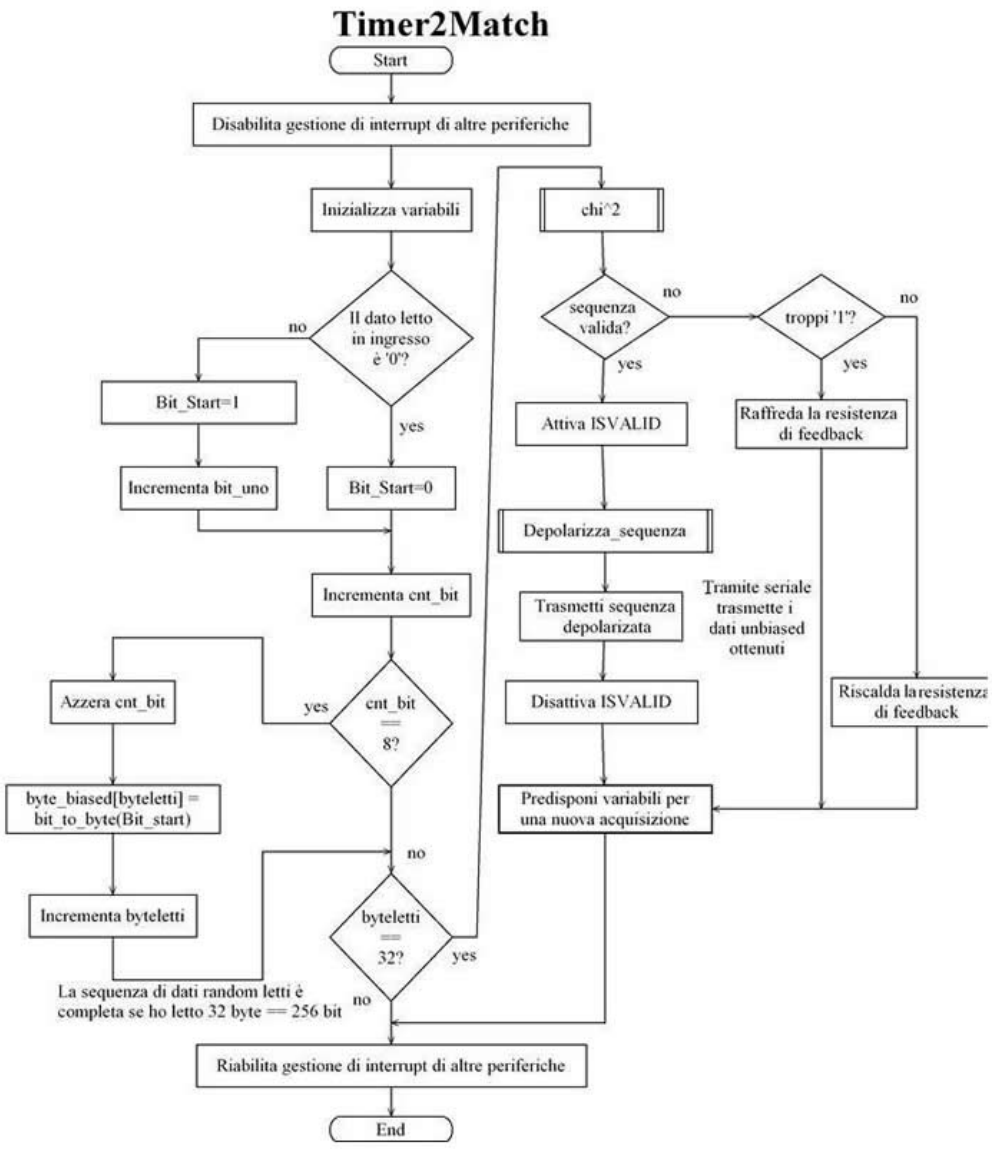

Grafico 3.6: Routine che gestisce l' interruput generata da Timer2

Le funzioni **chi^2** e **Depolarizza_sequenze** sono tecniche utilizzate per migliorare la randomness e sono affrontate nell' apposito capitolo.

4 Tecniche per migliorare la randomness

4.1 Gestione della retroazione

Il funzionamento della retroazione è semplice e sfrutta l' effetto Joule: il firmware del pic pone a '1' o '0' logico la tensione Vpic che va a pilotare in modo on-off il BJT che deve essere di tipo darlington data la piccola corrente che il pic è in grado di fornire e dell' elevata corrente che invece deve attraversare R11 che è il resistore di retroazione.
Non è necessario utilizzare un dissipatore per il bjt in quanto lavorando on-off si ha che o la Vce o la Ic sono nulle e quindi consuma potenza solo durante la commutazione.
R11 è formata in realtà da 2 resistori di potenza da 10Ω e 5W R11a e R11b collegati in parallelo con al centro R4 realizzando così la struttura a sandwich riportata in figura 4.1 in grado di scaldare R4.

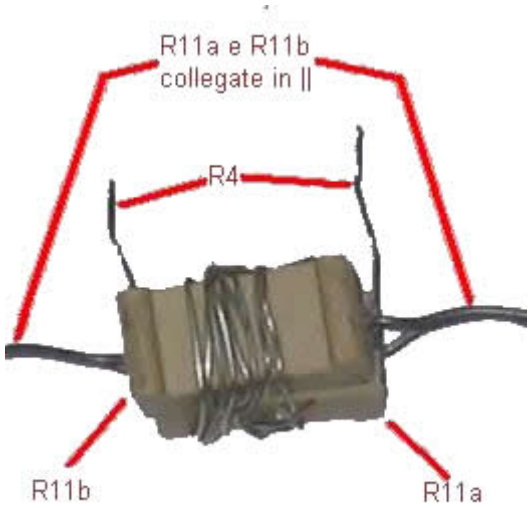

Figura 4.1: Struttura della resistenza di retroazione

Se dai test statistici il firmware si accorge che la sequenza acquisita contiene troppi '0' setta Vpic, il darlington va in conduzione e una corrente di circa 1A attraversa R11 che comincia a scaldare R4, di conseguenza il valore efficace della sua tensione di rumore V_n^2 aumenta. La conseguenza di ciò è che mediamente, pure il modulo della tensione istantanea di rumore aumenta e il comparatore A/D ad 1 bit avrà l' uscita sbilanciata verso Vcc aumentando così il numero di bit a '1' nella nuova sequenza acquisita.
Nel caso contrario invece il microcontrollore resetta Vpic, il bjt va in interdizione, R11 non è più attraversata da corrente, R4 si raffredda aumentando così la quantità di '0' nella nuova acquisizione.
Questo fatto si dimostra sperimentalmente collegando un oscilloscopio a valle del diodo D4 e misurando la tensione rispetto a massa al variare della temperatura di R3.
Figura 4.2 mostra i dati sperimentali di CASCINE ottenuti osservando la tensione di uscita, una volta regolata la soglia del convertitore, quando tutte le 4 resistenze del ponte sono alla stessa temperatura ambiente.
Non si nota la transizione tra i livelli logici '1' e '0' perché avviene casualmente e non in modo periodico: il trigger dell' oscilloscopio non riesce ad agganciare il passaggio per lo

zero e non è neppure possibile fornirlo dall' esterno per lo stesso motivo.
Figura 4.3 mostra la tensione di uscita ottenuta lasciando invariati tutti i parametri del caso precedente tranne la temperatura di R4 che è maggiore degli altri 3 resistori. Si nota un deciso sbilanciamento dell' uscita verso il livello logico alto come previsto dalla teoria.

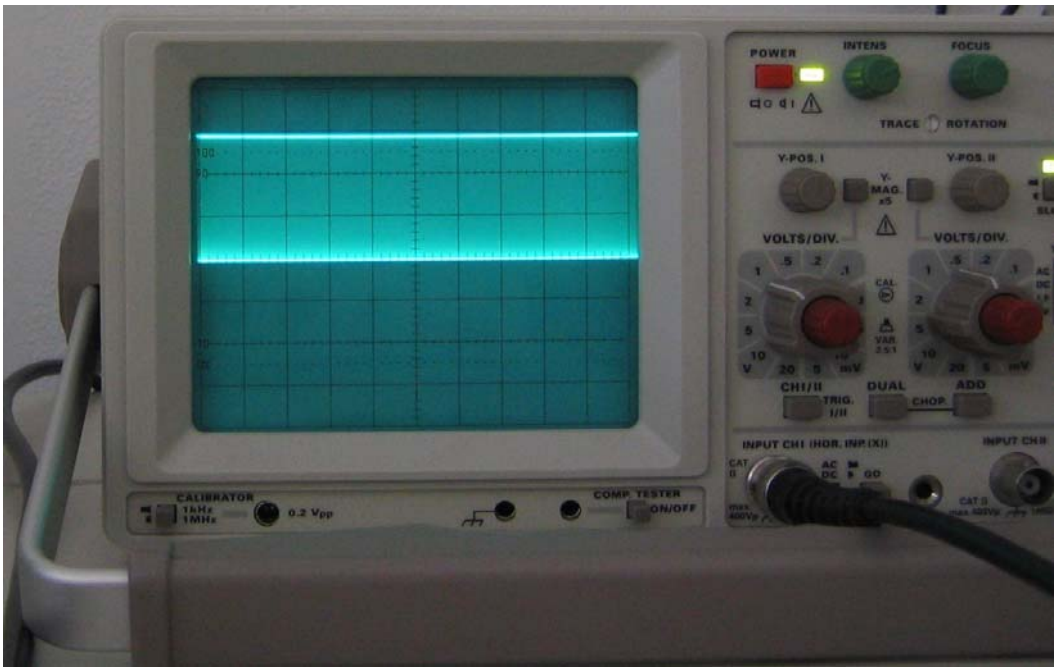

Figura 4.2: Vout misurata con il ponte resistivo a temperatura ambiente

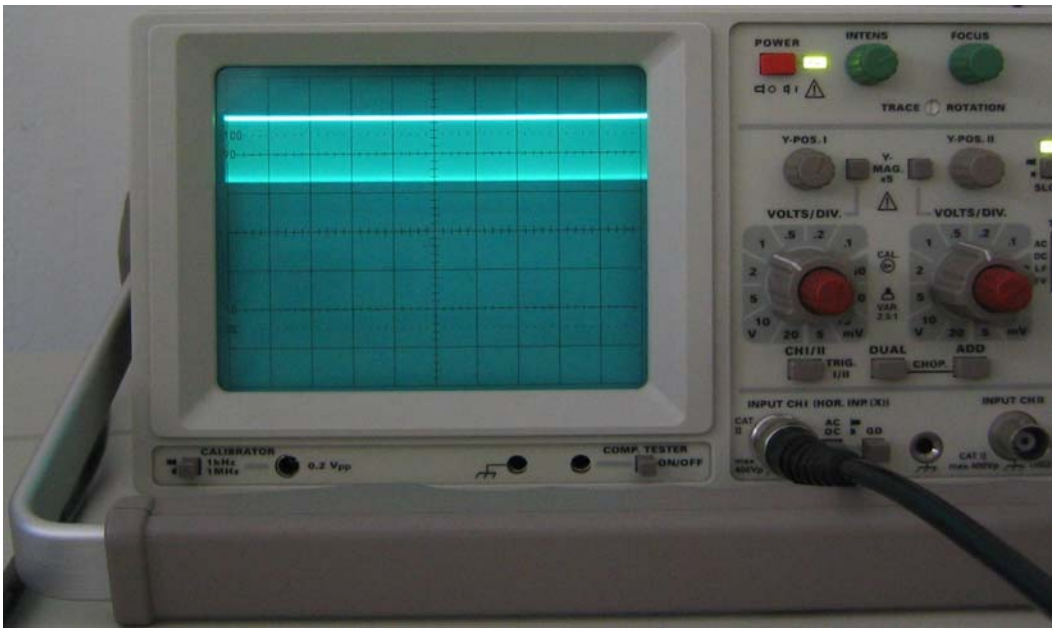

Figura 4.3: Vout misurata con R4 a temperatura maggiore di quella degli altri resistore del ponte resistivo.

4.2 Test statistici

Per migliorare le qualità di randomness delle sequenze generate il microcontrollore effettua dei test su di esse prima di trasmetterle in uscita in modo da eliminare tutte quelle che per svariati motivi hanno caratteristiche pessime.
Tipicamente alcune sequenze vengono eliminate o per via della retroazione termica che, a causa delle dimensioni e della massa delle resistenze di potenza, ha una grande inerzia termica e quindi il suo effetto si fa sentire dopo un po' di tempo o che la soglia del comparatore non è regolata bene e produce uno sbilanciamento nel numero di '0' e '1', e a volte, anche un insieme di pattern che si ripete con sufficiente periodicità. Queste sequenze potrebbero presentarsi anche con un generatore di numeri random perfetto ma con una probabilità talmente bassa che ho preferito eliminarle e non considerarle minimamente.
Decido se scartare o no la sequenza in base al test chi^2.
Una sequenza veramente casuale è formata mediamente per metà da uno e per metà da zero mentre una sequenza binaria qualsiasi lunga N ha N1 uno e N0 zeri con

$$N1+N0=N \qquad (4.1)$$

Si tratta di una sequenza con un solo grado di libertà in quanto data la lunghezza e il numero di uno si ottiene il numero di zero che la compongono.
I bin sono 2 in quanto le categorie che formano il dominio sono solo i bit zero e uno perciò ho modificato le operazioni eseguite normalmente per calcolare χ^2 procedendo come indicato dalle (4.2) con l' obbiettivo di ridurre il numero di operazioni necessarie e quindi velocizzare l' esecuzione del test.
Scegliendo un livello di confidenza 0.95 si vede dalle apposite tabelle che se la sequenza ha un valore della variabile χ^2 minore di 3,84 può essere considerata casuale[6] : la sequenza è una buona sequenza e quindi viene trasmessa in uscita.
Il PIC16F877A lavora con aritmetica intera perciò ho scelto, come si vede negli allegati, N= 256 e un valore di soglia di $\chi^2 = 3{,}75$ che è una condizione di lavoro ancora più stringente del valore teorico $\chi^2 = 3{,}84$ perché a ciò corrisponde $|N1-N0|\le 31$. Inoltre 31 è un numero intero.
Risolvendo il sistema (4.3) si vede che le sequenze che superano questo test hanno $112\le N1\le 144$; nel caso contrario la sequenza non supera il test: invece di trasmetterla controlla se è troppo elevato il numero di 0 o di 1 e setta o resetta Vpic rispettivamente per pilotare la retroazione termica.

$$\chi 2=((N0-N/2)^2)/(N/2)+((N1-N/2)^2)/(N/2) \qquad (4.2a)$$

$$\chi 2=((N0-N0/2-N1/2)^2+(N1-N0/2-N1/2)^2)/(N/2) \qquad (4.2b)$$

$$\chi 2=(N0-N1)^2/N \qquad (4.2c)$$

Applicando la (4.1) si ottiene:

$$\chi 2=(N0-N1)^2/(N0+N1) \qquad (4.2d)$$

$$\begin{aligned} |N1-N0|&\le 31 \\ N1+N0&=256 \end{aligned} \qquad (4.3)$$

6 Per questo test e per questi parametri.

Ho implementato a livello di firmware la funzione **chi^2** secondo il seguente flowchart:

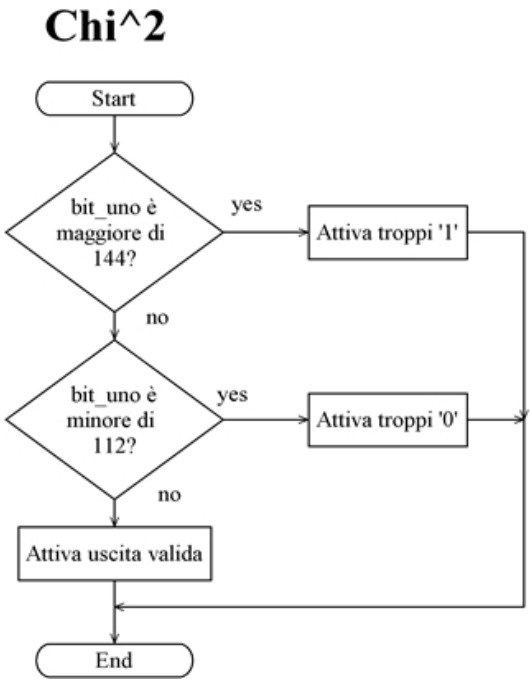

Grafico 4.1: Routine chi^2: stabilisce se la sequenza analizzata ha le stesse proprietà della sequenza random ideale

4.3 Depolarizzazione dei bit

Per migliorare ulteriormente le sequenze ho utilizzato, oltre alla retroazione termica e al test χ^2 , anche una funzione per ridurre e possibilmente eliminare la polarizzazione residua dei bit.
Se la probabilità di avere uno 0[7] nella sequenza originale è pari a $0.5+C$ con C costante opportuna (chiaramente C < ½) nella sequenza trasmessa è possibile ridurre gli effetti della polarizzazione usando la depolarizzazione di Von Neumann che consiste nell' eliminare tutte quelle coppie che hanno i 2 bit uguali altrimenti si trasmette solo il primo bit della coppia. Un esempio che spiega questo metodo è riportato in tabella 4.1.
Con questo procedimento la sequenza è depolarizzata anche se il bitrate può diminuire di molto: nel caso migliore, cioè quando C = 0, dimezza e all' aumentare di C diminuisce fino a diventare nullo nel caso teorico di C = ½.

sequenza	10	11	00	10	10	01	00	01	10	10	01	01	11	00	10	00	10	10	01	01
Von Neumann	1			1	1	0		0	1	1	0	0			1		1	1	0	0

Tabella 4.1: Depolarizzazione eseguita con il metodo di Von Neumann

7 Il discorso per 1 è analogo

Il firmware del microcontrollore PIC16F877A depolarizza la sequenza utilizzando la funzione **Depolarizza_sequenza** il cui flowchart è riportato nei grafici 4.2 e 4.2.

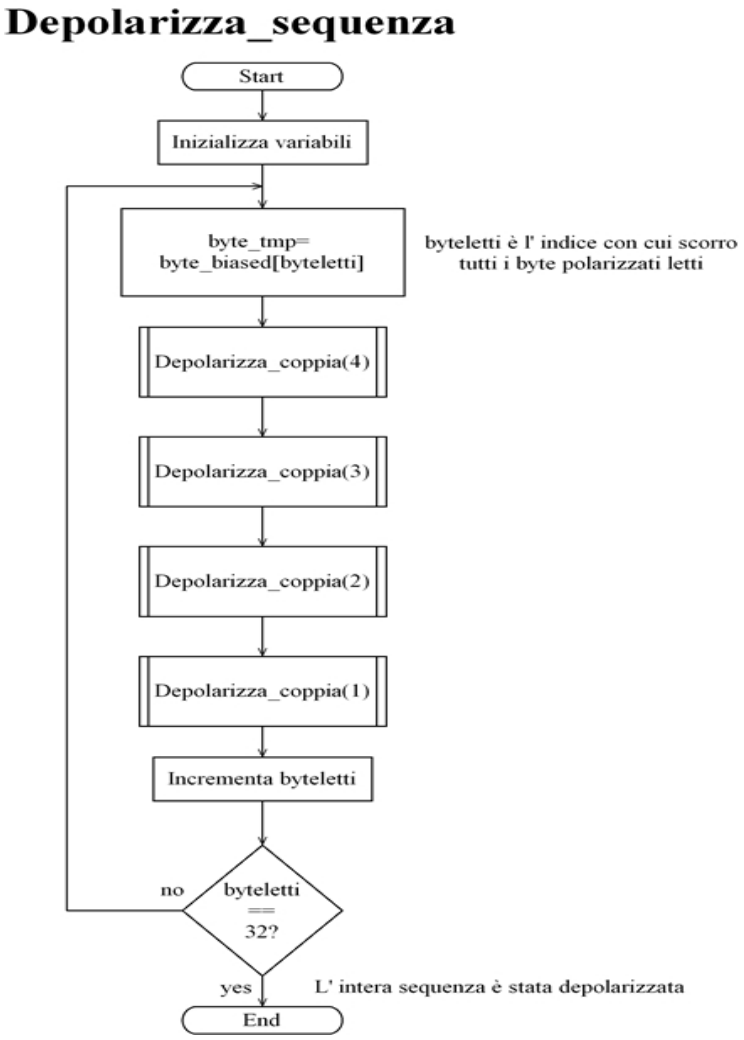

Grafico 4.2: Depolarizzazione della sequenza. Notare che Depolarizza_coppia è implementata tramite macro.

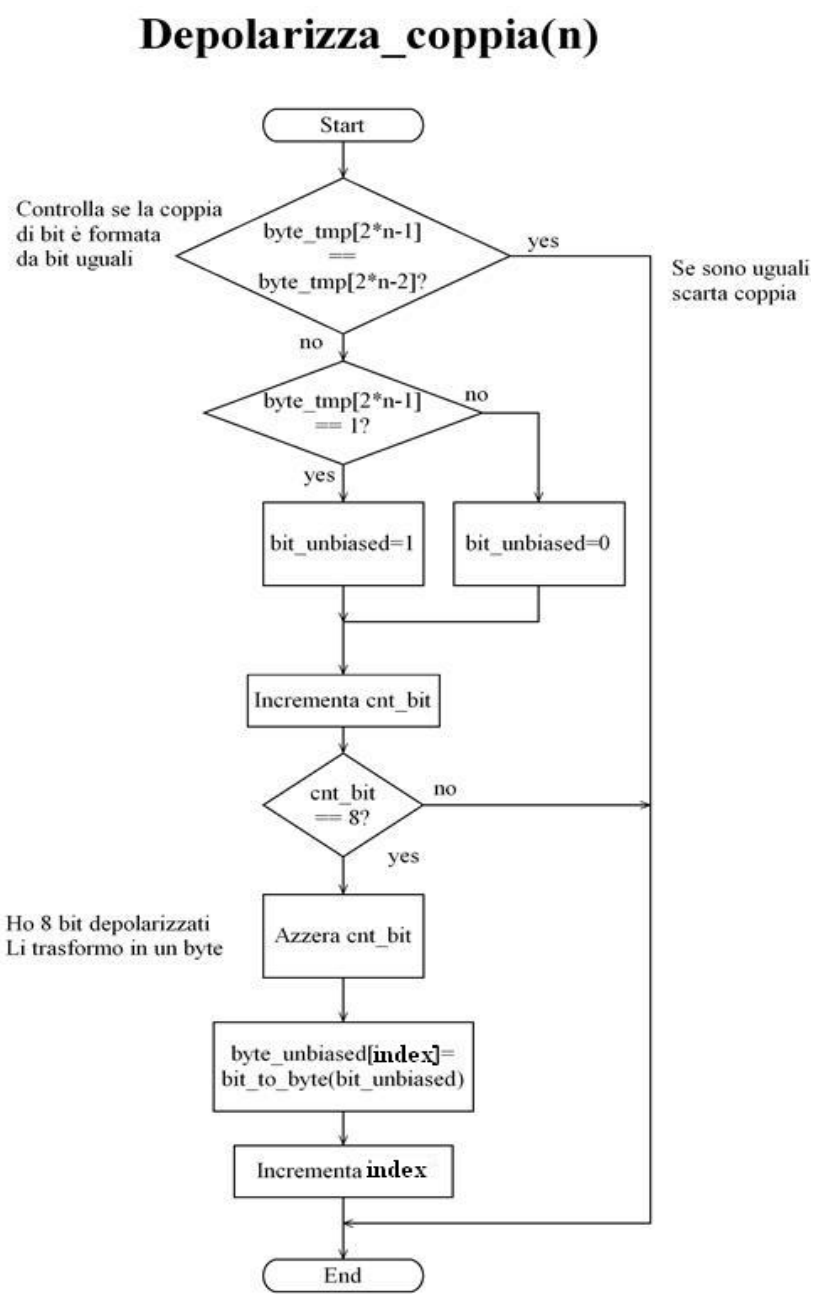

Grafico 4.3: Applicazione dell' algoritmo di Von Neumann per depolarizzare le sequenze di dati

5 Risultati sperimentali

5.1 Collaudo

Per collaudare CASCINE dopo aver controllato le tensioni di alimentazioni e il corretto funzionamento degli amplificatori e filtri ho dovuto regolare la tensione di soglia del convertitore A/D che è un passo di importanza cruciale.
Se la tensione di soglia è troppo bassa la sequenza avrà molti più '0' che '1' e la tensione di uscita di questo stadio avrà un andamento simile a figura 5.2, nel caso contrario ci saranno troppi '1' come si nota in figura 5.3, in quest' ultimo caso l' andamento della tensione è simile a quello di figura 4.3 ma la causa è completamente differente in quanto in queste fasi di misurazione con il pic scollegato il darlington è interdetto e le resistenze di retroazione sono fredde.
Per tarare la tensione di soglia è sufficiente ruotare il trimmer in senso orario o antiorario fino ad ottenere un andamento simile a quello di figura 5.1.
Poi ho collegato il pic e relative schede al pc come indicato nella documentazione della Microchip e utilizzando un programma in grado di comunicare con la porta seriale, ho usato *hyperterminal private edition* che a differenza dell' *hyperterminal* standard fornito con windows permette di salvare l' intera comunicazione su file, è possibile testare il funzionamento del pic.
Innanzitutto è necessario creare una connessione settando 9600 bit per secondo, 8 bit per simbolo, uno per lo stop e nessuno per la parità e nessun controllo di flusso. A questo punto mandando al micro dei comandi senza senso[8] ad esempio la stringa **LTK** si deve ottenere una serie di **e10** come risposta che corrisponde a "*L'header inviato non è stato riconosciuto, cioè non è un header dato o comando. Il modulo di ricezione viene predisposto per accettare un nuovo comando/dato"*.
Il protocollo di comunicazione è largamente inutilizzato non avendo previsto l' utilizzo da parte di CASCINE di nessun *header dato* e utilizzando solo 2 *header comando*:
RUN_ACQ e **STOP_ACQ** utilizzati per iniziare una nuova acquisizione di sequenze casuali o per stopparla.
Per come è stato definito il protocollo al comando RUN_ACQ corrisponde la stringa esadecimale 43046153 a cui sono associati i valori ASCII C?aS mentre a STOP_ACQ corrisponde la stringa esadecimale 43044153 a cui sono associati i valori ASCII C?AS.
Il carattere '?' indica che il simbolo non è rappresentabile graficamente.

Per mettere in funzione il dispositivo è sufficiente lanciare da hyperterminal il comando RUN_ACQ e salvare su file i dati trasmessi da CASCINE al pc. Se la velocità di acquisizione dei simboli fosse bassa o addirittura nulla significa che la taratura ad occhio della soglia è troppo grossolana e va ritarata più finemente agendo sul trimmer ruotandolo lentamente fino a massimizzare la velocità di produzione dei numeri casuali.
Ora dovrebbe funzionare tutto correttamente con l' hyperterminal che cattura e registra le sequenze generate mentre Vout vista all' oscilloscopio è simile a quanto riportato in figura 5.3.
Da notare che anche con la base dei tempi a 10 μS il trigger non riesce ad agganciare il passaggio per lo zero come spiegato in precedenza: significa che anche per questi intervalli temporali molto piccoli 2 campioni successivi della sequenza sono tra loro scorrelati.

8 Secondo le specifiche del protocollo di comunicazione non scritto da me

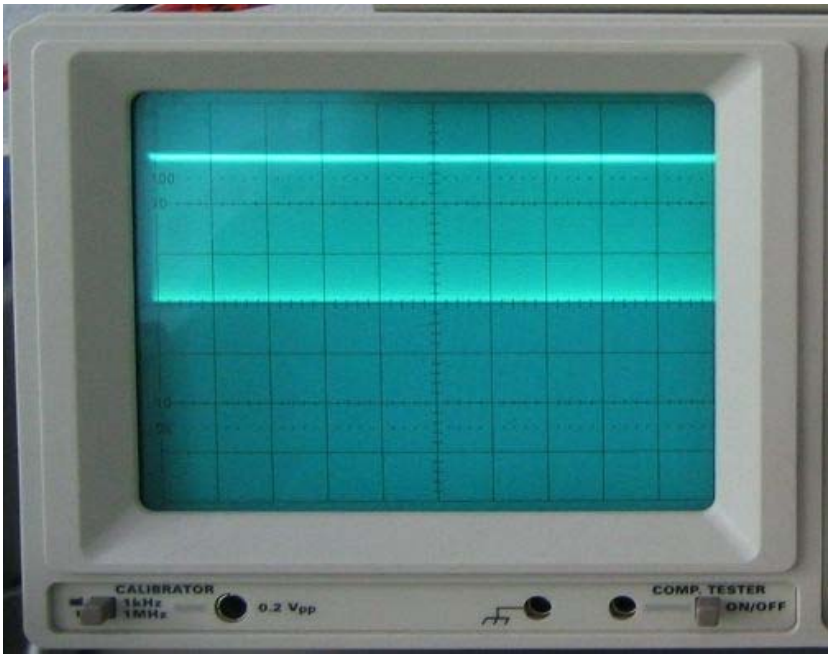

Figura 5.1: Esempio di soglia tarata correttamente.

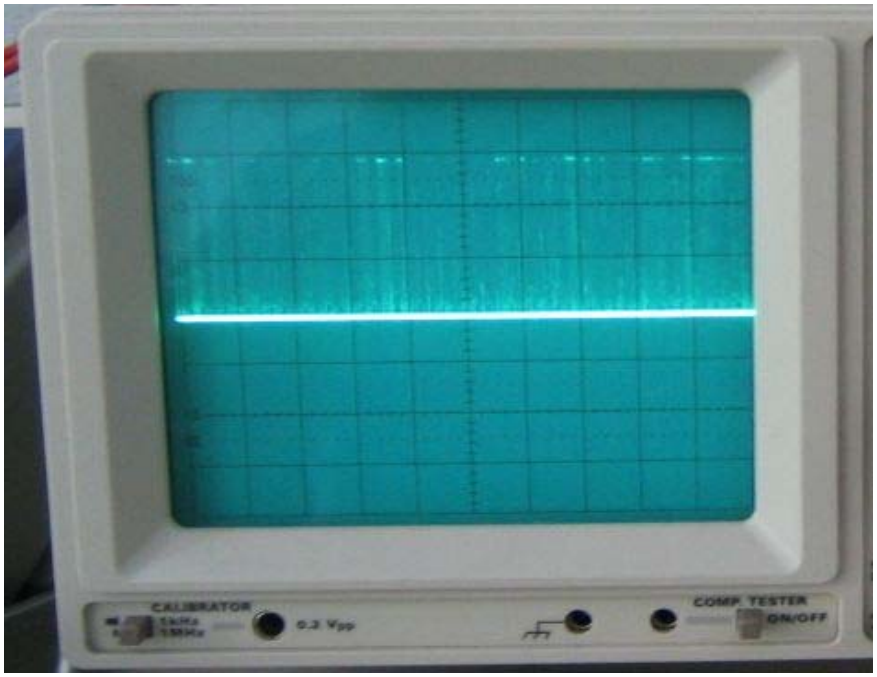

Figura 5.2: Tensione di soglia troppo bassa.

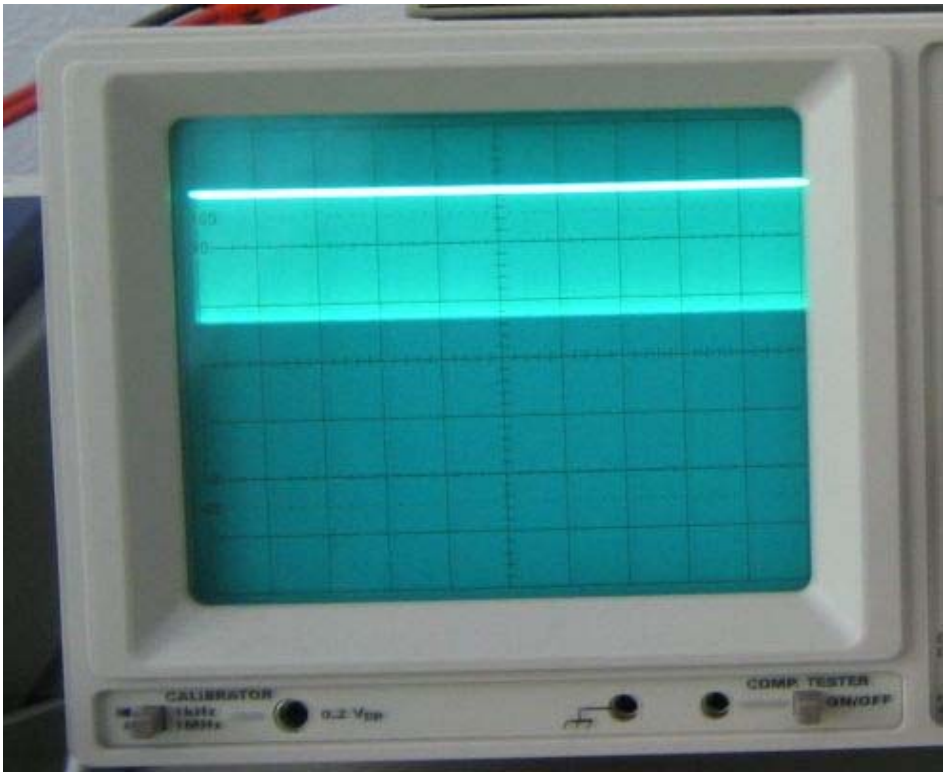

Figura 5.3: Tensione di soglia troppo alta.

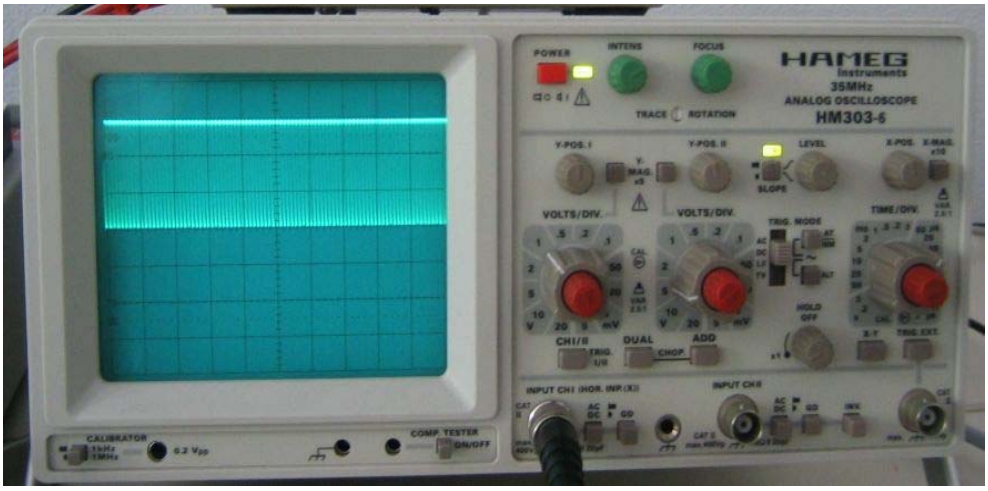

Figura 5.4: Andamento di Vout quando CASCINE è in funzione.

5.2 Influenza parametri

Dopo aver collaudato e verificato che CASCINE funge è possibile agire su alcuni parametri del firmware per migliorarne le prestazioni.
Per come ho strutturato il programma è possibile controllare due soli valori: il baud rate del modulo USART tramite il parametro *VELOCITA* e il valore di matching del timer2 tramite *MATCHING*. Entrambi si trovano nel file const.inc sezione *PARAMETRI CASCINE*.

La breadboard utilizzata fornisce una frequenza di lavoro di 4 MHz e avendo scelto di porre BRGH = 1, il baud rate BR si calcola utilizzando $BR = f/(16(x+1))$ dove x è il valore del parametro *VELOCITA*.

Per quanto riguarda il timer2 ho impostato il prescaler a 1:1 e il postscaler a 1:2 per ottenere un segnale alla frequenza di $Fosc/4*1/1*1/2 = 500\,KHz$ che posso dividere ulteriormente di un valore pari a MATCHING per effettuare il numero voluto di acquisizioni di Vout in un secondo.
Con CASCINE perfettamente tarato e con la retroazione termica, il test chi^2 e la depolarizzazione attivi ho effettuato differenti prove allo scopo di trovare i parametri migliori e ho scelto come coppia ottima VELOCITA = 25 e MATCHING = 250.

6 Analisi risultati

6.1 Risultati delle analisi

Tutte le prove, se non diversamente specificato, sono state eseguite su un campione di circa 8Mbit di dati casuali acquisiti con le modalità indicate e diviso in 50 subsequenze.
Come prima cosa ho analizzato i dati ottenuti disabilitando il test del chi^2, la depolarizzazione e la gestione della retroazione **in modo da studiare i dati grezzi forniti dalla sorgente di rumore termico** e acquisiti direttamente senza nessuna elaborazione.
Dai risultati dell' analisi si capisce che la sorgente **non è una buona sorgente** perchè approximate entropy test, cumulative sums test sia forward che reverse, fft test, frequency test, rank test e serial test falliscono nel 100% dei casi.
Questo significa che la sorgente di rumore impiegata ha il vantaggio di potere utilizzare una retroazione basata sulla temperatura e non su una grandezza elettrica ma non è affatto buona perché dall' esito del nonperiodic-templates test si capisce che ci sono molti pattern che si ripetono con una frequenza troppo superiore o inferiore rispetto ad una sequenza random ideale. Gli altri test indicano anche che la sequenza è polarizzata e che la distribuzione degli '0' e degli '1' non è uniforme.

Come seconda prova ho abilitato il test del chi^2 ottenendo:

Test	**2kHz[numero di fallimenti]**	**8kHz[numero di fallimenti]**
cumulative-sums	67 su 100 prove	89 su 100 prove
fft	46 su 50 prove	39 su 50 prove
frequency	9 su 50 prove	12 su 50 prove
nonperiodic-templates	702 su 7400 template	657 su 7400 template
runs	48 su 50 prove	46 su 50 prove

Con la depolarizzazione di Von Neumann le caratteristiche migliorano sensibilmente perché oltre ad **eliminare le sequenze troppo sbilanciate viene eliminata pure la polarizzazione residua** ma la velocità di generazione dei numeri casuali diminuisce moltissimo: scartando le coppie di bit uguali nel caso migliore la velocità diventa i 1/2 rispetto al caso precedente con il solo test del chi^2 attivo ottenendo:

Test	Risultati [numero di fallimenti]
approximate entropy	49 su 50 prove
block-frequency	1 su 50 prove
cumulative-sums	89 su 100 prove
fft	1 su 50 prove
frequency	46 su 50 prove
nonperiodic-templates	231 su 7400 template
runs	2 su 50 prove
serial	2 su 50 prove

Con tutte le tecniche di ottimizzazione attive invece la situazione cambia decisamente infatti è possibile notare immediatamente **un aumento della velocità di produzione** dei dati una volta che la retroazione termica va a regime.
Per ottenere le sequenze da analizzare con queste prove ho tarato CASCINE, la tensione di soglia è il valore più critico, e lo ho lasciato funzionare a vuoto per una decina di minuti per essere sicuro che i vari transitori, in particolare quello termico che è il più lungo, si esaurissero dopo di che ho iniziato l' acquisizione delle sequenze.
I risultati ottenuti analizzando la sequenza col NIST sono riportati in tabella.

Test	Risultati [numero di fallimenti]
approximate entropy	29 su 50 prove
block-frequency	2 su 50 prove
cumulative-sums	2 su 100 prove
fft	4 su 50 prove
frequency	1 su 50 prove
nonperiodic-templates	917 su 7400 template
rank	0 su 50 prove
runs	43 su 50 prove
serial	7 su 50 prove

Per calcolare la velocità di produzione dei numeri casuali ho acquisito 32 sequenze di differente durata temporale, rispettivamente di 45 - 60 - 120 - 180 - 240 - 300 - 360 e 600 secondi ottenendo un totale di 4038512 bit in un tempo di 7555 secondi e una **velocità di generazione media di bit pari a 534 bit / sec** mentre per i dati grezzi la velocità media è di 5241 bit / sec: questo vuol dire che la sorgente produce sequenze con circa il 90% dei bit da scartare.

Bibliografia

[1] Girgio Brida, “Appunti del corso: processi di rumore”, 2005, http://www.ien.it/~brida/SegnaleRumore/meccanismi%20di%20rumore.pdf
[2] Franco Carvelli, Ruben Zocco, “Sistema elettronico basato su microcontrollore per la generazione di numeri casuali”, 2004, http://minosse.dibe.unige.it/CdS/Dati/Tesi/T2562672.PDF
[3] Revised NIST Special Publication 800-22, "A Statistical Test Suite for the Validation of Random Number Generators and Pseudo Random Number Generators for Cryptographic Applications" , 2001, http://csrc.nist.gov/rng/SP800-22b.pdf
[4] Cristiano, “RaBiGeTe Random Bit Generators Tester”, MMIV, http://www.webalice.it/cristiano.pi/rabigete/
[5] Lorenzo Pareschi, “Numeri Casuali”, 2006, http://www.utenti.unife.it/lorenzo.pareschi/talks/numeri%20casuali.pdf
[6] Autori vari, http://it.wikipedia.org/
[7] Autori vari, http://en.wikipedia.org/
[8] Evilcry, “A study about radomness, security tests and weakness analisys”, http://evilcry.altervista.org/tuts/RndNumbers.zip
[9] De Vita Laura, Savignano Tiziana, “Numeri Casuali”, 2000, http://www.dia.unisa.it/~ads/corso-security/www/CORSO-9900/randomness/master.html
[10] Beneduce Leopoldo,Napolitano Gavino,Palma Domenico,Sannino Ilaria, “Crittografia Classica”, 2000, http://www.dia.unisa.it/~ads/corso-security/www/CORSO-9900/crittografiaclassica/index.htm
[11] Bernd Ulmann, “A True Random Number Generator”, http://vaxman.de/projects/rng/rng.html
[12] John Walker, “Genuine random numbers, generated by radioactive decay”, 2006, http://www.fourmilab.ch/hotbits/
[13] Cecconi Alessia, Gallileo Giovanni, “Cryptography”, 1997, http://telemat.det.unifi.it/book/1997/cryptography/Welcome.html
[14] Alessandro Ferrero, “L’amplificatore operazionale”, 2000, http://www.etec.polimi.it/studenti/matdid/OpAmp/
[15] Charles Wright, “Deterministic Random Number Generators”, 2004, http://islab.oregonstate.edu/koc/ece399/f04/final/Wright.pdf
[16] Gary McGraw, John Viega, “Make your software behave: Beating the bias”, 2000, http://www-126.ibm.com/developerworks/library/s-beating.html
[17] VincenzoV, “Idee per chi lavora con l'elettronica (... o ci gioca)”, http://www.vincenzov.net/
[18] “Amplificatori operazionali”, http://www.scuolaelettrica.it/elettrotecnica/volume2.html
[19] Claudio Fin, “PIC - Appunti di utilizzo”, 2005, http://stor.altervista.org/pic/pic.htm
[20] Sergio Tanzilli, “Pic by example”, 1991, http://www.acmesystems.it/?id=207
[21] Io, “Comunicazioni private e pensieri personali con me stesso medesimo”, 2006, http://125.0.0.1[9]

9 Ma non entrate dalla back door. Tnx, Giulia