I SEGRETI DEI ROLLING-CODE

Entriamo nel regno dei messaggi cifrati per scoprire il principio di funzionamento dei nuovissimi telecomandi per antifurto basati su questo metodo. Si tratta di una vera anteprima di cui sentiremo parlare per molto tempo...

Andrea Sbrana

Quando, molti anni fa, nacque il famosissimo integrato della National siglato MM53200N, si pensava che nel campo della sicurezza fosse il chip che doveva risolvere ogni problema di codifica e decodifica nel settore dei telecomandi e delle chiavi per allarme.

Ben presto però si scoprì che le sue 4.094 combinazioni non erano più sufficienti a garantire un buon margine di sicurezza, poiché bastava costruire un "generatore di codici" sequenziale per, nel giro di alcuni minuti, aprire qualsiasi serratura impiegante l'MM53200N.

Il principio era molto semplice: se si voleva spegnere un antifurto di una certa marca, bastava acquistarne uno simile, leggere la frequenza di uscita del trasmettitore in dotazione e realizzare, modificando lo stesso, un generatore di codice con pochi componenti esterni.

In Figura 1a è possibile vedere il diagramma a blocchi di un simile circuito.

Capite quindi che, con il passare del tempo, nacquero altri encoder-decoder per chiavi di sicurezza, come ad esempio il conosciutissimo MC145026 con i relativi MC145027 e MC145028 della Motorola.

A differenza dei primi, questi chip potevano vantare di un numero maggiore di combinazioni (19.683) e, di conseguenza, anche il tempo per trovare la combinazione giusta cresceva.

In Figura 1b vediamo il diagramma a blocchi di un generatore di codice necessario per implementare il terzo stato.

Apriamo una piccola parentesi per specificare che il primo chip aveva il codice componibile con 12 ingressi digitali, quindi le massime combinazioni possibili erano date dal numero di livelli per ingresso elevato al numero di ingressi, quindi 2 elevato a 12.

Nel caso dell'MC145026, i livelli per ogni ingresso potevano essere tre (0, 1 e terzo stato, cioè pin lasciato scollegato) mentre il numero di ingressi era fissato in 9. Allora le combinazioni venivano calcolate come 3 elevato alla 9.

Anche per questo chip, fu adottata la soluzione simile alla precedente, con la differenza di dover, mediamente, attendere un tempo quattro volte maggiore.

Con l'avvento dei microcontroller poi, la sicurezza offerta da questi due tipi di integrati e dai loro simili, è naufragata nel giro di pochi mesi: sono stati messi in commercio, telecomandi che riescono a memorizzare un codice ricevuto, di qualunque tipo esso sia, e poi ritrasmetterlo sulla stessa frequenza.

I costruttori di antifurti allora hanno richiesto dei prodotti che potessero rendere la vita più difficile anche ai ladri più esperti e, dopo aver relegato i due encoder-decoder prima citati ai soli telecomandi per apricancelli e simili, hanno iniziato ad impiegare degli integrati realizzati appositamente, oppure gli stessi microcontroller per crearsi dei codici particolari, non ripetibili, molto lunghi e quindi difficilmente copiabili come i precedenti ed è così nato il cosiddetto "Rolling-Code", ovvero tradotto letteralmente codice rollante.

Il principio del Rolling-Code

In teoria, codice rollante significa che ad ogni trasmissione viene generato un nuovo codice, che deve sincronizzarsi con quello inserito nel ricevitore.

In pratica, ci sono delle restrizioni che, per non far perdere la sincronizzazione tra i due elementi che dialogano, impongono altre regole più flessibili.

Vedremo ora come viene implementato un Rolling-Code generico, anche se

quelli in commercio sono molto simili per numero di bit e tempistiche.

Poiché avevamo verificato che i 12 bit dell'MM53200N erano pochi, il nostro ideale Rolling-Code si compone di 64 bit, quindi già con un numero di combinazioni di partenza pari a

184.470.000.000.000.000.000.

In pratica, se volessimo impiegare un generatore come nel caso di Figura 1, impiegheremmo in media circa 146.240.000.000 anni!

Ma, come abbiamo già visto, con i sistemi che riescono a leggere e riprodurre qualsiasi codice su qualsiasi frequenza, l'indice di sicurezza di un simile protocollo, sarebbe identico a quello dell'MM53200N.

Ci rendiamo conto allora che aumentare il numero di bit si è reso necessario per aumentare il numero di combinazioni, ma non sufficiente a risolvere del tutto il problema dell'inespugnabilità.

Il passo successivo è stato quindi quello di far sì che il codice, ad ogni trasmissione, venga modificato sia sul trasmettitore che sul ricevitore. Come?

Nella maggior parte dei Rolling-Code esistenti oggi sul mercato vengono posizionati 32 bit fissi e gli altri 32 vengono modificati ad ogni trasmissione.

In questo modo si garantiscono oltre 4 miliardi di combinazioni "fisse" da abbinare ad esempio a più case costruttrici, e altri 4 miliardi di combinazioni variabili ad ogni trasmissione. Per coloro che non si intendono di calcolo delle probabilità ricordiamo che la risultante delle combinazioni totali è di ben 4 x 4 miliardi e non, come si potrebbe pensare erroneamente, di 4 + 4. Vediamo però le restrizioni cui avevamo precedentemente accennato: per prima cosa, se il trasmettitore viene attivato continuamente, il codice non varia durante la trasmissione ma rimane costante.

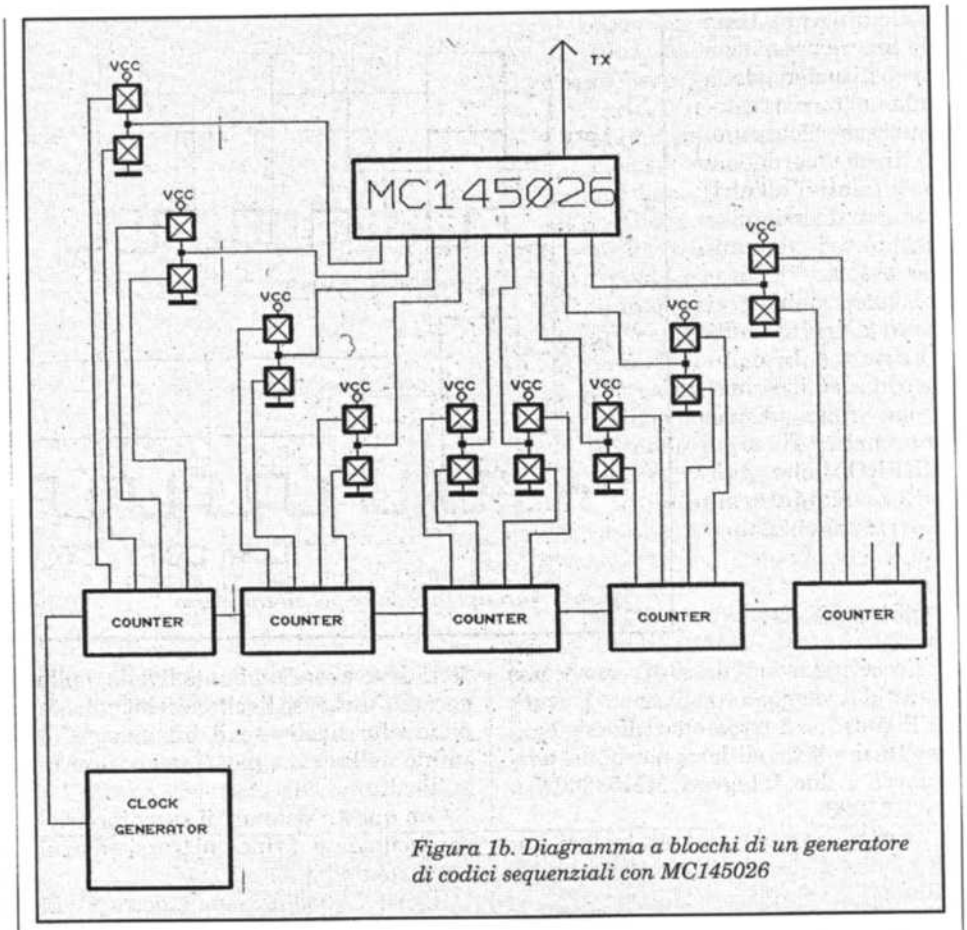


Figura 1b. Diagramma a blocchi di un generatore di codici sequenziali con MC145026

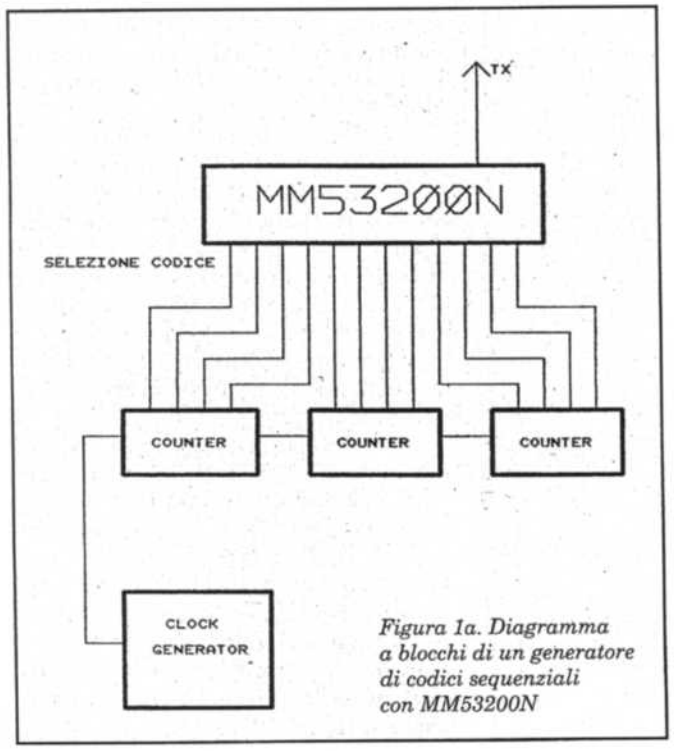


Figura 1a. Diagramma a blocchi di un generatore di codici sequenziali con MM53200N

Lo stesso vale se il trasmettitore viene attivato consecutivamente a distanza di un certo tempo impostabile da ogni costruttore e variabile in genere da 1 a 10 secondi.

Questo è necessario se si pensa ad una situazione in cui l'utente, avvicinandosi all'automobile, attiva il telecomando, ma non è sufficientemente vicino, e quindi il ricevitore non "sente" la trasmissione.

In questo modo, il codice del trasmettitore avanzerebbe di una combinazione mentre quello del ricevitore resterebbe sul precedente, creando una desincronizzazione.

Nella pratica ciò è difficile che avvenga, poiché è prevista una "finestra" di 256 combinazioni consecutive da parte del ricevitore: quando l'apparecchio riceve un codice, lo confronta con il suo attuale.

Se si verifica l'abbinamento (detto anche matching) il chip ricevente dà il consenso, altrimenti tenta di comparare le successive 255 combinazioni derivate dal suo codice attuale.

Se pensiamo infatti che, per errore, per distrazione, oppure per semplice divertimento l'utente attivi una o più volte il trasmettitore quando non è nel raggio del ricevitore, per forza di cose il codice del trasmettitore passerà alle combinazioni successive lasciando indietro il codice del ricevitore.

La finestra di 256 combinazioni consecutive dovrebbe essere sufficiente per prevenire simili situazioni.

Un altro problema da tenere presente è la sostituzione della pila nel trasmettitore: ci sono alcuni modelli che perdono i dati relativi all'ultima trasmissione e che, quindi, dovranno essere risincronizzati, mentre ci sono altri modelli che non subiscono perdite dell'informazione grazie ad una memoria di tipo EEPROM che può essere sia interna che esterna al chip.

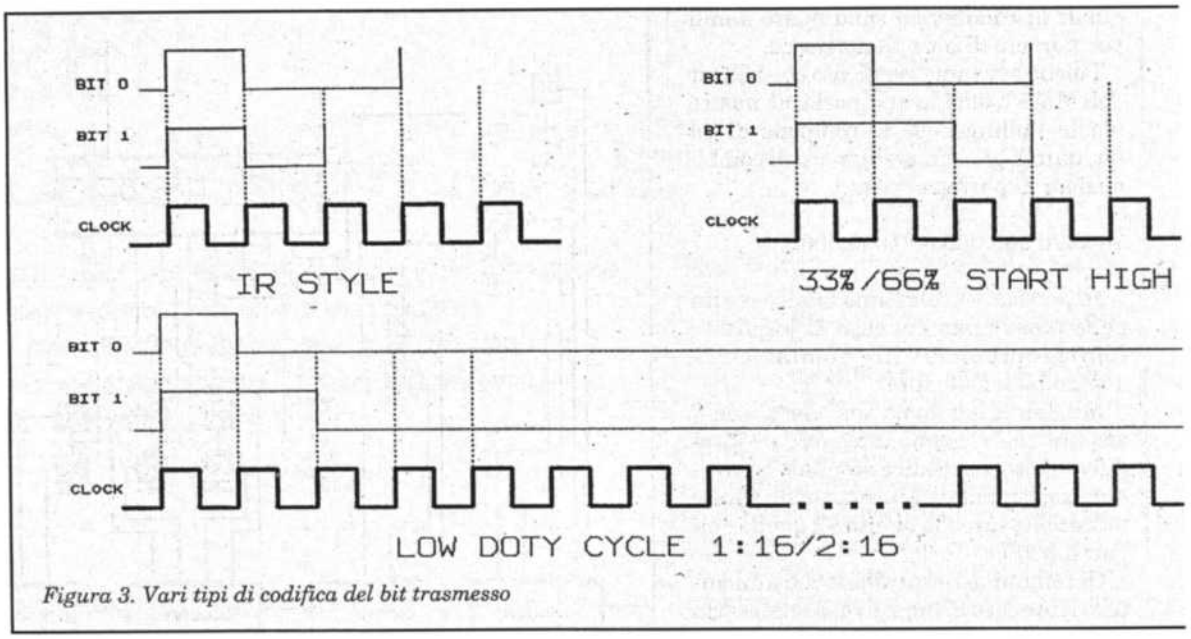


Figura 3. Vari tipi di codifica del bit trasmesso

I bit trasmessi

Preoccupiamoci adesso di vedere come questi bit vengono trasmessi e ricevuti: in Figura 2a e 2-b vediamo i due sistemi, molto simili tra di loro, per quanto riguarda i due integrati MM53200N e MC145026.

Il primo prevede uno start bit, poi una sequenza di 12 bit propri del codice e lunghi tre volte lo start bit.

Nella prima parte obbligatoriamente si deve avere un basso livello, nella seconda un basso livello se il bit vale "1" o un alto livello se il bit vale "0" e infine nella terza parte sempre un livello alto.

Con questo sistema, il ricevitore riesce facilmente a sincronizzarsi ad ogni arrivo di un bit.

Il tipo di modulazione è detto PWM ovvero modulazione a larghezza (durata) di impulso, mentre il protocollo è conosciuto con il nome di 33%-66% End High.

Il secondo non prevede uno start bit, ma si sincronizza anche questo all'arrivo di ogni bit (Figura 2b).

Sebbene questi due tipi di protocolli siano i più diffusi, ultimamente sono stati realizzati chip che possono supportare, oltre a questi, anche altri protocolli, come per esempio quelli riportati in Figura 3.

Recentemente, inoltre, con l'ausilio di controller sempre più veloci, i tempi di durata di un bit si sono notevolmente ridotti, permettendo l'invio di 64 bit in poche centinaia di microsecondi, contro i millisecondi richiesti dai modelli più anziani.

Il Rolling-Code, però, non è minimamente influenzato dal genere di trasmissione impiegata, in quanto l'importante è mantenere la corretta sequenzialità, senza preoccuparsi se un bit viene trasmesso con codice Manchester oppure con codice 25%-50% Start High.

E così pure non influisce il mezzo di transito: vanno bene sia la radiofrequenza che i raggi infrarossi che il collegamento diretto.

Cerchiamo adesso di capire, con un esempio pratico, come funziona uno dei tanti metodi detti Rolling-Code.

Supponiamo di avere il nostro codice fisso di 32 bit uguale alla stringa 11000111000011010101111001111100 che in esadecimale è uguale a C70D5E7C e supponiamo sempre di averlo come primi 32 bit (in pratica poi si può anche decidere se "mischiare" byte del codice fisso con byte del codice random per

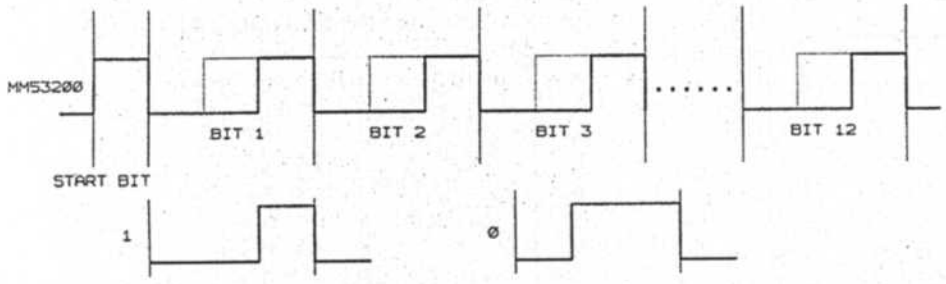


Figura 2a. Esempio di codice trasmesso da un MM53200N

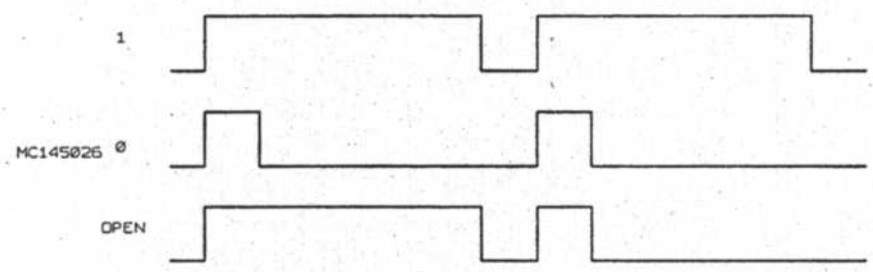
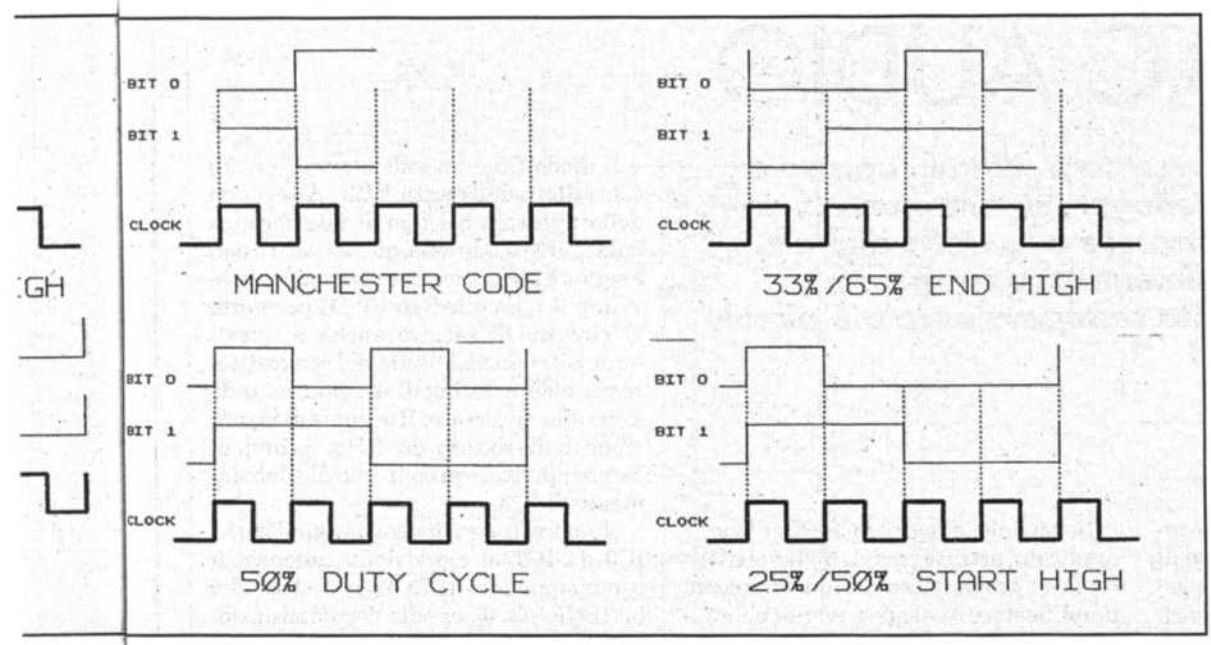


Figura 2b. Esempio di codice trasmesso da un MC145026



creare un po' di confusione a chi tenti una decodifica).

Supponiamo poi di avere la stringa random di soli 16 bit (per sveltire le operazioni) uguale a 1110001011010100 cioè a E2D4. Quando il trasmettitore è sincronizzato con il ricevitore, l'unione della prima stringa con la seconda rappresenta il codice di partenza.

Tutti i codici successivi dovranno poi essere calcolati sia dal trasmettitore che dal ricevitore con lo stesso medesimo algoritmo.

Nel caso più banale, supponiamo che questo algoritmo sia la somma di 1 alla stringa random.

Allora tale stringa, dopo la prima trasmissione, diventerà uguale a 1110001011010101

In questo modo, se qualcuno ha "captato" la vostra prima trasmissione, non sapendo quale sia l'algoritmo da voi impostato, non potrà fare assolutamente niente per "violare" la vostra serratura.

Chiaramente, gli algoritmi applicabili sono infiniti, poiché visti come formule matematiche: la stringa random può, di fatti, potrebbe anche essere condizionata da un algoritmo che invece di fare una semplice somma, prima moltiplica per X, poi somma Y, poi divide per Z ed infine uno XOR con un'altra stringa programmata.