IBB Ingenieur-Büro Bergmann, Sonnenweg 3, 15537 Grünheide, info@ibbergmann.org

BENUTZERHANDBUCH

des Physikalischen Zufallsgenerators PRG260

Version 4.0

Autor: **Frank Bergmann**

Letzte Änderung: **01.08.2017 11:28**

1 Inhaltsverzeichnis

2 Copyright

IBB Ingenieurbüro Bergmann

Sonnenweg 3

D-15537 Grünheide

Ausgabe vom 20.11.2014

Schutzrechte

Für wesentliche Schaltungsdetails des Physikalischen Zufallsgenerators sind Schutzrechte eingetragen.

Patentnummern:

- Deutsches Patent DE 102 23 252 vom 18.06 2003
- Europäisches Patent EP 150 98 38 vom 15.03.2006

Haftung

Bei der Erarbeitung dieser Dokumentation wurde größter Wert auf die Vollständigkeit und Richtigkeit des Inhalts gelegt. Es kann dennoch keine Garantie für die Vollständigkeit und Richtigkeit übernommen werden.

Für Hinweise zu dieser Dokumentation sind wir dankbar.

Hotline

Die Hotline des Herstellers erreichen Sie unter +49(0)172 308 6554.

Warenzeichen

MS Windows ist eingetragenes Warenzeichen der Microsoft Corp.

3 Bedeutung von Zufallszahlen

Gute Zufallszahlen sind das Fundament vieler kryptographischer Verfahren und Protokolle. Es ist wichtig, dass die verwendeten Zufallszahlen nicht vorhersagbar sind. Solche Zufallszahlen zu erzeugen, fällt Computern naturgemäß schwer. Zahlreiche Meldungen kritisieren Lücken, Schwächen und Manipulationen bei der Erzeugung von Zufallszahlen für kryptografische Verfahren.

Bei allen Meldungen geht es nicht um spezielle, bedeutungslose Applikationen, sondern um millionenfach installierte Standardprogramme in professionellen Anwendungen. Vor allem dort, wo kontinuierlich viele Zufallszahlen benötigt werden (Netzwerke, Kommunikationssysteme), sind statistische Angriffe auf schwache Zufallsgeneratoren am erfolgreichsten.

Nach dem Kerckhoff-Prinzip (die Sicherheit soll nur auf der Geheimhaltung des Schlüssels beruhen, nicht auf der Geheimhaltung des kryptographischen Algorithmus) benötigt jede Art von Verschlüsselung eine geheime Komponente, die unter keinen Umständen vorhersagbar oder rekonstruierbar sein darf: der aus Zufallszahlen gebildete Schlüssel. Diese Zufallszahlen werden in den bekannten IT-Sicherheitsapplikationen aus Pseudozufallszahlen gebildet. Quelle der Generierung von Pseudozufall ist ein so genannter Seed (ein Startwert, bestehend aus Passwort, Timer-Register, Tastaturanschlägen, Mausbewegungen usw.), mit dem ein mathematisch-kryptografischer Algorithmus eine statistisch gut verteilte Zufallsfolge erzeugt. Aber die gesamte Sicherheit der per Pseudozufall erzeugten geheimen Schlüssel hängt *ausschließlich* von dieser Anfangsinitialisierung ab. Die Anfangsinitialisierung ist bei richtiger Wahl der Quelle der einzige wirklich zufällige Parameter, alles Weitere ist *deterministisch* und somit berechenbar. Eine schwache Anfangsinitialisierung (trivialer Seed) ist im statistischen Ergebnis nicht erkennbar, aber ein effizienter Angriffspunkt der Kryptoanalysis.

Auch professionelle Entwickler nutzen als Seed für Pseudozufall oftmals das Timerregister in der Annahme: wer will denn schon wissen, in welcher Sekunde das Register ausgelesen wurde. Für einen Angreifer kein Problem, denn ein Jahr hat ca. 32 Millionen Sekunden. Und um diese mit der totalen Probiermethode (brute force) durchzutesten, benötigt man nur eine durchschnittliche Rechenleistung. Wird der gleiche Seed mehrfach verwendet, so entstehen schlüsselgleiche Geheimtexte. Ein sicherer Erfolg für die Kryptoanalysis.

Der Kryptoanalysis stehen heute wesentlich leistungsfähigere Werkzeuge zur Verfügung, so dass immer häufiger Meldungen zu kompromittierten schwachen Zufallsgeneratoren veröffentlicht werden. Dagegen haben die in IT-Sicherheitslösungen implementierten Pseudozufallsgeneratoren einen Stand erreicht, der eine neue Qualität der Zufallserzeugung erfordert.

Pseudozufall basierende Zufallsquellen sammeln in diversen Entropiequellen in der Hoffnung, davon ausreichende Mengen zu sammeln. Zwar werden in diversen Chip-Sätzen (VIA, Transmeta, Renesas) und Security-Chipkarten derartige Zufallsgeneratoren angeboten, aber über Entropie und Qualität der Zufallsrohdaten werden keine oder nur unzureichende Angaben gemacht. Aus deren Chips konnten Sicherheitsforscher um Bernstein über 80 eindeutige RSA-Schlüssel auslesen, die gemeinsame Primfaktoren haben. Grund dafür war ein fehlerhafter Random Number Generator im AE45C1-Chip von Renesas, der nicht genügend Entropie erzeugt. Die veröffentlichten Daten zur Entropie des VIA C3 PadLock (7,64 Bit/Byte) zeigen für hohe Sicherheitsansprüche nicht ausreichende Werte. Zur Bit-Unabhängigkeit werden keine Informationen aufgeführt. Aber: nichts sagt mehr über die Eigenschaften eines physikalischen Zufallsgenerators aus, wie seine Rohdaten. Jede weitere Verarbeitung der Rohdaten verschleiert nur die wirklich statistischen Basiseigenschaften und kann vor allem die Entropie nicht weiter erhöhen.

4 PRG260

Bei dem PRG260 handelt es sich um einen physikalischen Zufallsgenerator mit einer UART-Schnittstelle mit TTL-Pegel. Der PRG260 unterstützt die professionelle Generierung von kryptografisch sicheren Zufallszahlen der Klassen PTG.2 und PTG.3. Vorzugsweise ist der PRG260 für den Einsatz auf den Raspberry-Pi-Boards vorgesehen, kann aber auch auf jeder anderen Pin-kompatiblen Plattform eingesetzt werden.

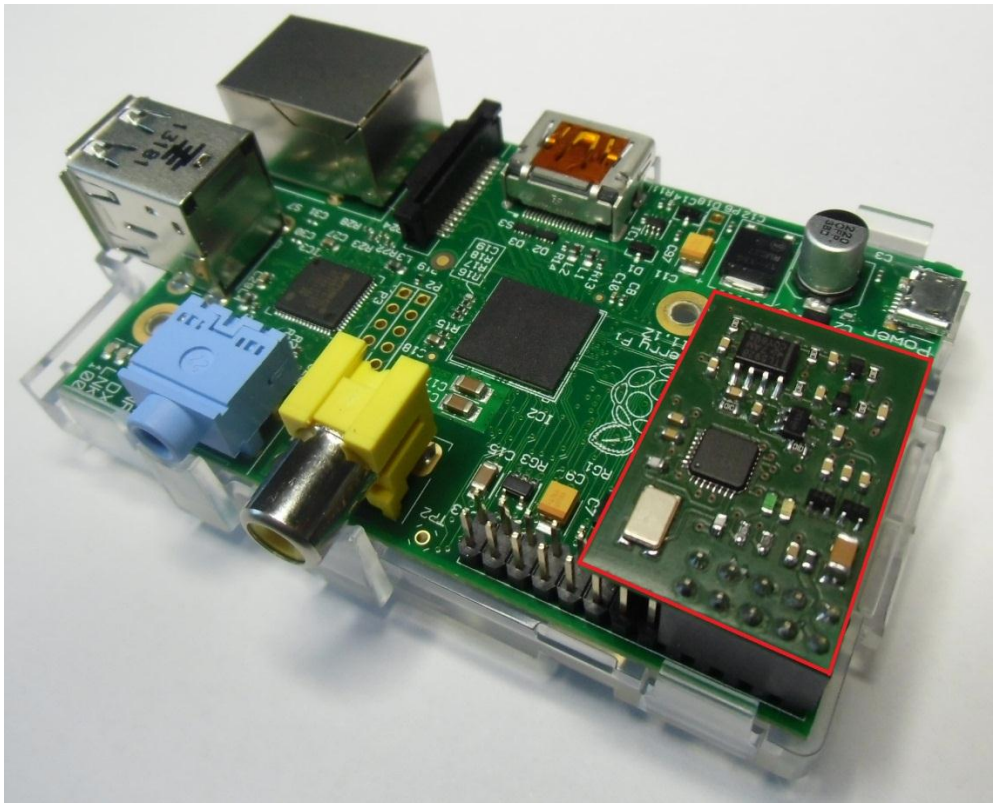


Abbildung: PRG260 auf einem Raspberry-Pi-Board

Kern des PRG260 ist ein patentierter physikalischer Zufallsgenerator des IBB:

- Deutsches Patent DE 102 23 252 vom 18.06 2003
- Europäisches Patent EP 150 98 38 vom 15.03.2006

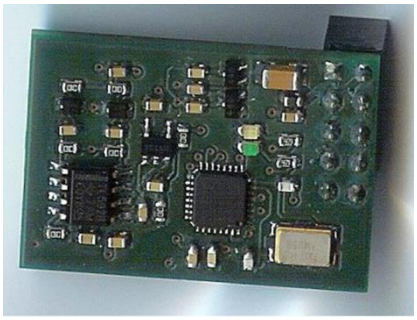


Abbildung: PRG260-Modul

Grundlage der Entwicklung des im Folgenden vorgestellten PRG260 ist ein stochastisches Modell, mit dem die Leistungsfähigkeit zur Generierung kryptografisch sicherer Zufallszahlen begründet wird. Dieses Modell erklärt:

- die robuste und hohe Entropie der Rauschquelle
- das Prinzip der Abtastung des analogen Rauschsignals,
- die permanente Überwachung der Rauschquelle durch Frequenzmessung
- die kryptografische Nachbearbeitung durch Mayer-Einwegfunktionen
- den permanenten statistischen Online-Test

In Deutschland hat die Regulierungsbehörde für IT-Sicherheit (Bundesnetzagentur BNetzA) folgende Verbindlichkeiten im „Algorithmenkatalog 2014" festgelegt:

„Für Zertifizierungsdienstanbieter wird die Verwendung von Zufallsgeneratoren der Funktionalitätsklassen *PTG.3* und DRG.4 im Grundsatz *ab 2015 verpflichtend* werden, sowohl allgemein bei der Erzeugung von Langzeitschlüsseln als auch bei der Erzeugung von Ephemeralschlüsseln."

Bemerkungen:

- Hybride Zufallszahlengeneratoren vereinen Sicherheitseigenschaften von deterministischen und physikalischen Zufallszahlengeneratoren.
- Hybride physikalische Zufallszahlengeneratoren der Klasse PTG.3 besitzen neben einer starken Rauschquelle eine starke kryptographische Nachbearbeitung mit Gedächtnis.
- PTG.3 stellt die stärkste Funktionalitätsklasse dar.

Alle aktuellen Lösungen des IBB entsprechen den Forderungen der Klassen PTG.2 (physikalische Zufallsgeneratoren) und PTG.3 aus dem Algorithmenkatalog 2014.

5 Technische Daten

Abmessungen:	45*19*10 mm (mit Pfostensteckverbinder)
Versorgungsspannung:	5V (+/- 10%)
Stromaufnahme:	max. 35mA
Temperaturbereich:	funktionell und statistisch stabil von -20°C..+70°C
Schnittstellen:	UART-Interface mit TTL-Pegel, 115.200 bps, Protokoll 8,N,1
Qualitätssicherung:	automatischer Selbstabgleich von Verstärkung und Digitalisierung
	Tot-Test zur Überwachung der Rauschquellen
	Abschaltung der Zufallsausgabe bei Ausfall einer Rauschquelle
	Online-Test zur statistischen Überwachung des Zufallssignals
	Intensiver Test zur erweiterten statistischen Kontrolle des Zufalls- Rohsignals
Entropie:	>7,997 Bits/Byte (ermittelt aus Zufalls-Rohdaten nach Shannon)
0/1-Verhältnis:	garantiert im Bereich 0,49..0,51 (Rohdaten > 8.000 Bit)

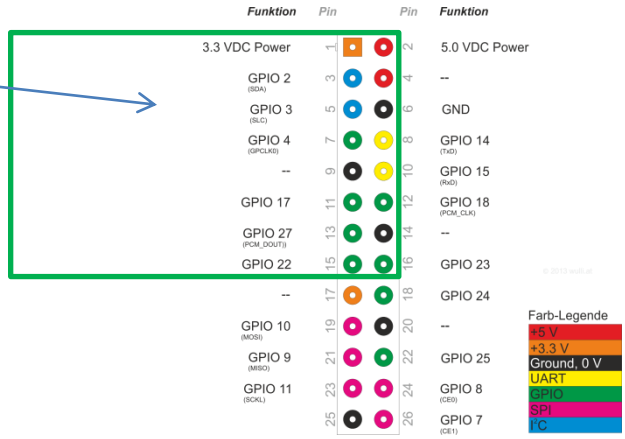
6 Genutzte GPIO des Raspberry-Pi

Der PRG260 nutzt folgende Pins des Raspberry-Boards:

- Pin 2: 5.0V
- Pin 6: GND
- Pin 8: TXD
- Pin 10: RXD

PRG260-Modul

(Draufsicht)



7 Prinzip der Rauscherzeugung des PRG260

Rauschen ist ein physikalisches Phänomen und stellt eine Störgröße mit breitem unspezifischem Frequenzspektrum dar. Dieses Frequenzspektrum besteht aus der Überlagerung mehrerer Schwingungen oder Wellen mit unterschiedlicher Amplitude und Frequenz beziehungsweise Wellenlänge. Diese Eigenschaften wurden erstmalig 1918 durch Walter Schottky beschrieben. Später wurde das thermische Rauschen experimentell durch John Bertrand Johnson verifiziert. Eine Modellvorstellung der spektralen Leistungsdichte des thermischen Rauschens erfolgte durch Harry Nyquist

Das in diesem Zufallsgenerator verwendete 1/f-Rauschen bezeichnet ein Rauschen, dass mit steigender Frequenz abnimmt, die Amplitudenverteilung ist umgekehrt proportional zur Frequenz (~ 1/f). Die verwendeten Rauschquellen sind Z-Dioden, die in Sperrrichtung betrieben werden. Rauschen entsteht hier durch den Lawineneffekt (Avalancheeffekt) in der pn-Sperrschicht des Halbleiterbauelements (Dioden und Transistoren). Dioden und Transistoren lassen sich durch ein kontrolliertes Avalanche-Verhalten vor Zerstörung durch Überspannungen schützen. Rauschen ist Zufall. Die nächstfolgenden Rauschwerte können nicht vorausgesagt werden.

Beispielsweise können Z-Dioden höherer Durchbruchspannung Rauschspannungen von 50mV in einem breiten Rauschspektrum erzeugen. Technologisch bedingt haben alle Z-Dioden beliebiger Hersteller diese Eigenschaft bei einer definierten Durchbruchspannung. Da Störspannungen umgebender Schaltungen diese Rauschspannungen überlagern, wurden zwei Z-Dioden verwendet und an einen Differenzverstärker geschaltet. Dieser hat die angenehme Eigenschaft, Gleichtakte (Störsignale) sehr wirksam zu unterdrücken. Dadurch sind beste Voraussetzungen gegeben, eine sehr hohe Entropie zu generieren. Das informationstheoretische Verständnis des Begriffes Entropie geht auf Claude E. Shannon zurück und existiert seit etwa 1948. In diesem Jahr veröffentlichte Shannon seine fundamentale Arbeit

A Mathematical Theory of Communication und prägte damit die moderne Informationstheorie. Der Informationsbegriff von Shannon bewertet nicht den semantischen Inhalt oder die Bedeutung einer Nachricht, sondern dessen Unvorhersagbarkeit.

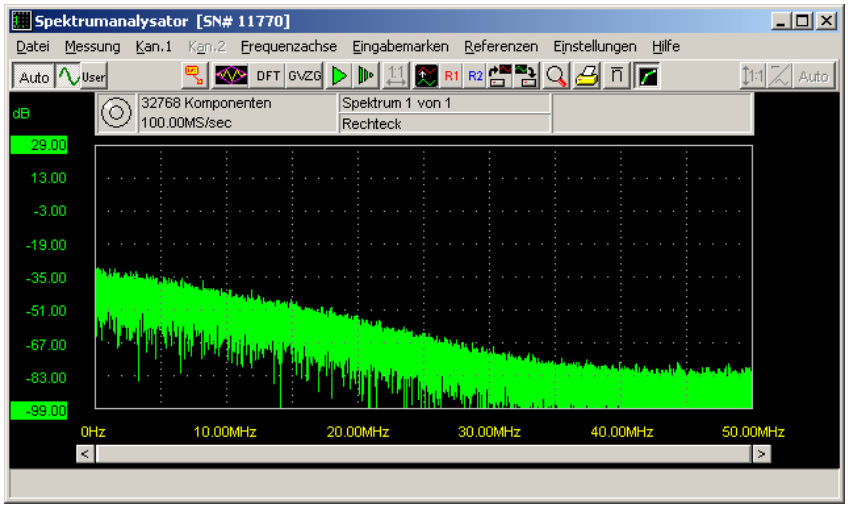


Abbildung: Typisches 1/f-Rauschen nach der Verstärkung

8 Generierung des Zufallssignals

Die Rauschsignale zweier Z-Dioden werden einem Differenzverstärker zugeführt, der eine ca. 300-fache Verstärkung realisiert und durch die hohe Gleichtaktunterdrückung Störsignale sehr wirksam eliminiert. Ein schneller Schmitt-Trigger mit einer Hysteresis von ca. 0,5V digitalisiert das verstärkte Rauschsignal und führt es einem Porteingang des Mikrocontrollers zu.

Das Blockschaltbild und oszillografische Aufnahmen zeigen wesentliche Komponente des PRG260:

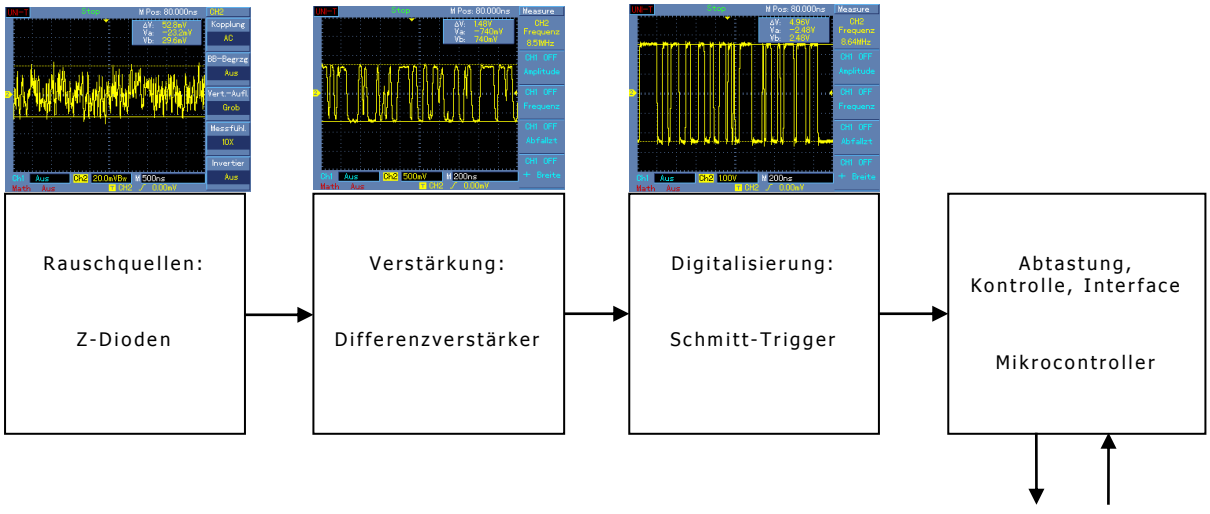


Abbildung: Rauschsignale und Blockschaltbild des PRG260

9 Entropie

Die Entropie der Zufallsrohdaten (keine Verarbeitung der abgetasteten digitalisierten Rauschsignale) ist die entscheidende Eigenschaft eines echten Zufallsgenerators und sollte so hoch als möglich sein. Die reproduzierbare Generierung von Zufallsdaten mit hoher Entropie der PRG260-Applikationen zeigen beispiellose Werte.

Folgende Entropiewerte (nach Shannon) der Rohdaten wurden für verschiedene Applikationen ermittelt:

PRG260	Mittelwert 0/1	Entropie der Rohdaten
Modul 1	0.50342629	7.99972901
Modul 2	0.50048198	7.99999464
Modul 3	0.50165468	7.99993680
Modul 4	0.49894816	7.99997446

10 Generierung von Klassen

Auf Grund der sehr hohen Entropie des physikalischen Zufallsgenerators wurden die Funktionen für die Zufallsgenerierung für die

- Klasse PTG.2 (echter physikalischer Zufallsgenerator)
- Klasse PTG.3 (hybrider Zufallsgenerator)

ausgelegt. Für beide Klassen können beliebig lange Zufallsfolgen generiert werden.

In der Klasse PTG.2 sind folgende Ausgaben möglich:

- Zufallsfolgen mit Rohdaten (keine Nachbearbeitung)
- Zufallsfolge mit XOR2-Verknüpfung von aufeinanderfolgenden Rohdaten
- Zufallsfolge mit XOR3-Verknüpfung von aufeinanderfolgenden Rohdaten
- Zufallsfolge mit vonNeumann-Verknüpfung von aufeinanderfolgenden Rohdaten

In der Klasse PTG.3 erfolgt die Nachbearbeitung der generierten Rohdaten mit Mayer-Einwegfunktionen, realisiert mit dem AES128-Algorithmus in folgendem Schema (rechtes Bild):

Diese Nachbearbeitung nutzt 2 Prinzipien des Entropiesammelns: das Aufxorieren der Rohdaten für z1 und die Glättung durch Aufxorieren des mit zufälligem Schlüssel verschlüsselten Zustands für z2. Beide Komponenten nutzen die Mayer-Einwegfunktion AES(k,s) xor k, um den vorangegangenen inneren Zustand zu schützen. Durch die XOR-Summe der Zwischenwerte kann nicht auf den inneren Zustand geschlossen werden.

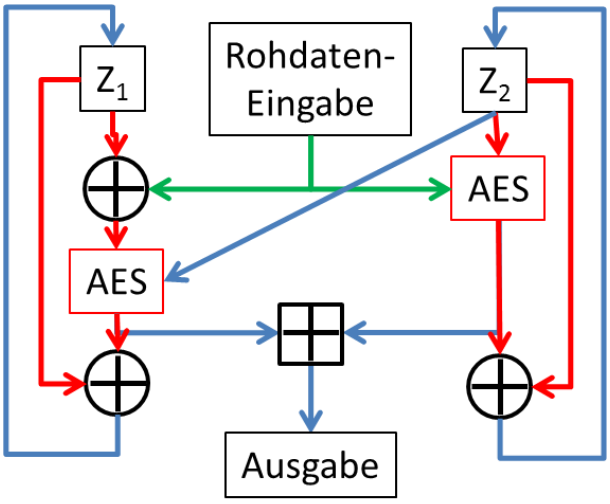


Abbildung: Schema der Nachbearbeitung

Dadurch werden auch bei einem kurzzeitigen Ausfall oder Manipulation der Rauschquellen statistisch gleichverteilte Zufallsdaten ausgegeben. 16 Byte Rohdaten erzeugen immer 16 Byte nachbearbeitete Zufallsdaten in der Ausgabe.

Die Generierungsgeschwindigkeiten der Zufallsdaten erreichen folgende Werte:

- PTG.2-Klasse
  - Rohdaten 320 Kbit/s
  - XOR2 166 Kbit/s
  - XOR3 116 Kbit/s
  - vonNeumann 134 Kbit/s
- PTG.3-Klasse
  - AES128 142 Kbit/s

Die reale Ausgabegeschwindigkeit ist durch das UART-Interface begrenzt.

11 Sicherheitsfunktionen

11.1 Tot-Test

Es ist ein Kontrollsystem installiert, welches das digitalisierte Rauschsignal am Anschluss des Mikrocontrollers überwacht. Unmittelbar vor und nach jeder Byte-Generierung des digitalisierten Rauschsignals durch die verschiedenen Funktionen der Zufallsgenerierung wird eine Funktion aufgerufen, die folgende Aufgabe hat:

- Es wird die Zeit ermittelt die erforderlich ist, um vier wechselnde Flanken des digitalisierten Rauschsignals zu erfassen
- Wird nach 16µs diese Bedingung nicht erfüllt, wird eine Fehlermeldung generiert, die zur sofortigen Einstellung aller Zufallsausgaben führt
- Dieser Zustand ist nur durch einen „Intensiven Selbsttest" oder PON aufzulösen
- Typische Zeiten, um die Bedingung zu erfüllen, sind 2..8µs

Sollte eine der Rauschquellen ausfallen entsteht am Eingang des Mikrocontrollers ein Mäander von ca. 20ms und wird durch den Tot-Test eindeutig als Fehlzustand erkannt

11.2 Permanenter Online-Test

Im permanenten Halbbytetest (zyklisch im Abstand von 1 Sekunde) zur Kontrolle der statistischen Qualität der Zufallsrohdaten werden drei Kriterien erfüllt:

- Sind die Werte innerhalb der statistischen Vorgaben, wird kein Fehler generiert
- Sind die Werte außerhalb der statistischen Vorgaben, aber noch innerhalb von weitestgehend gleichverteilten Zufallsdaten, wird ein Fehlerzähler inkrementiert und beide Leuchtdioden blinken im Sekundentakt
- Ist einer der 16 Werte im Halbbytetest gleich Null, wird von einem Totalausfall mindestens einer Rauschquelle ausgegangen und jede Zufallsausgabe blockiert. Angezeigt wird dieser Zustand durch statisches Leuchten beider Leuchtdioden und einem Fehlerwert von 255 (0xff). Dieser Zustand kann nur durch PON oder Aufruf des intensiven Selbsttest mit positivem Ergebnis beendet werden.

Beispiel für ein Ergebnis eines permanenten Online-Tests (in hex-Werten, Mittelwert 20, Index links Halbbyte 00, Index rechts Halbbyte 0f):

25 1E 20 23 14 1A 22 23 19 2B 1E 23 20 25 1C 21

11.3 Intensiver Test der Zufallsrohdaten

Im Kommandoaufruf „Intensiver Selbsttest" kann mit engeren statistischen Grenzen die Zufallserzeugung mit Rohdaten geprüft werden. Dieser Aufruf ist die einzige Möglichkeit (außer PON), eine Blockierung der Zufallsausgabe zu beenden.

Dieser Test generiert 1280 Halbbytes und bewertet die Anzahl der aufgetretenen 16 möglichen Index-Werte (0x00 bis 0x0f). Die Bewertungsgrenzen sind der Schiefe der Zufallsrohdaten angepasst. Ein bestandener intensiver Selbsttest muss fünf Mal hintereinander einen Testdurchlauf bestehen. Sechs Versuche stehen zur Verfügung. Werden die Kriterien nicht erfüllt, wird ein Hardwarefehler generiert und jede Zufallsausgabe blockiert. Die Bewertungsgrenzen wurden so gewählt, dass der statistische Fehler bei 1% aller Tests liegt.

Dieser Zustand kann nur durch einen erneuten intensiven Selbsttest oder durch PON aufgelöst werden.

Bei der Rückmeldung der Funktion „Intensiver Selbsttest" durch den PRG260 wird als 2. Byte die Anzahl der benötigten Versuche ausgegeben. Damit kann eingeschätzt werden, welche Qualität die Zufallsrohdaten zum Zeitpunkt des Tests haben.

Beispiel für ein Ergebnis eines intensiven Tests (in hex-Werten, Mittelwert 0x50, Index links Halbbyte 0x00, Index rechts Halbbyte 0x0f):

55 4B 56 49 60 4D 54 4C 4F 56 3F 5A 51 57 4C 42

12 Statistische Qualität

Dieser physikalische Zufallsgenerator generiert kontinuierlich Zufallsbits in herausragender statistischer Qualität. Eine Qualitätsaussage zu einem solchen Produkt ist aber nicht durch einen einzelnen statistischen Test möglich. In Zusammenarbeit mit Mathematikern und Kryptologen hat sich die Summe aus folgenden statistischen Tests für eine sichere Qualitätsaussage eines physikalischen Zufallsgenerators bewährt:

- Statistische Forderungen der AIS31-Dokumente für Rohdaten (keine digitale Nachbearbeitung, denn nichts beschreibt besser die Eigenschaften eines physikalischen Zufallsgenerators, als seine Rohdaten!)
- NIST-Test-Suite (Summe verschiedener Test der USA-Sicherheitsbehörde)
- Diehard-Test-Suite nach George Marsaglia
- Proprietärer statistischer Basistest

Zur Evaluierung wurden umfangreiche statistische Tests durchgeführt. So wurden die ausgewählten Tests auf mehrere erzeugte Bitfolgen angewendet. Keine dieser Testfolgen konnte Unterschiede zu einem idealen Zufallszahlengenerator aufzeigen. Darüber hinaus wurden Testergebnisse mit Untersuchungen von kryptographisch starken Pseudo-Zufallszahlengeneratoren, wie dem DES-, AES- und SH1-Generator verglichen. Die Vergleiche zeigten keine signifikanten Unterschiede zu den Testergebnissen dieser deterministischen Generatoren.

13 Sicherheitshinweise

Um eine sichere Generierung von Zufallszahlen zu gewährleisten, sind folgende Sicherheitshinweise zu beachten:

- Zugriff und Nutzung in sicherheitsrelevanten Bereichen ist nur autorisierten Personen zu gestatten
- Vor Nutzung und in zyklischen Abständen (1 x pro Stunde) ist der „Intensive Selbsttest" aufzurufen und das Ergebnis auszuwerten
- Kontinuierlich arbeitende Applikationen sollten in kurzen Intervallen (1 x pro Minute) den Fehlerzähler des permanenten Online-Test aufrufen und auswerten

14 Anwendungen

Auf Grund der sehr hohen Entropie des physikalischen Zufallsgenerators wurden die Funktionen für die Klassen PTG.2 (echter physikalischer Zufallsgenerator) und PTG.3 (hybrider Zufallsgenerator) ausgelegt. Bezugnehmend auf die AIS31-Dokumente (Funktionalitätsklassen und Evaluationsmethodologie für physikalische Zufallszahlengeneratoren: www.bsi.bund.de/zertifiz/zert/interpr/ais31e.pdf).ist ein permanenter Online-Test und ein Tot-Test (Ausfall der Rauschquelle) der Zufallsrohdaten integriert. Werden statistische Grenzen überschritten, wird ein Fehlerzähler aktiviert, der vom Anwender jederzeit abgefragt werden kann. Fällt die Rauschquelle aus, wird die Ausgabe von Zufallsdaten blockiert.

Der stetig steigende Bedarf an Zufallszahlen in vielen Bereichen von Wissenschaft und Technik erfordert eine zuverlässige und stabile Generierung von Zufallszahlen mit physikalischem Zufall und sicherer Gewährleistung aller erforderlichen statistischen und funktionellen Normen. Damit sind vor allem Entwickler von Sicherheitsapplikationen in der Lage, Wirksamkeit und Widerstand gegen Angriffe besser zu kalkulieren. Besonders hohe Ansprüche an die Statistik von Zufallszahlen werden für kryptografisch sichere Zufallszahlen gestellt.

Exemplarische Beispiele für den Einsatz des PRG260:

- Netzwerksicherheit
- Transaktionen beim Homebanking (SSL-Verschlüsselung)
- Dateiverschlüsselung mit Security-Applikationen (Speicher-Sticks)
- einfachere Administration und sichere Verschlüsselung bei drahtloser Datenübertragung: WLAN, Bluetooth, GSM, ZiggBee, Industriedatenfunk
- Internet-Verschlüsselung
- elektronischer Zahlungsverkehr
- Erstellung von PKI-Zertifikaten
- OneTimePad-Verfahren

15 Einsatzumgebung

Der PRG260 ist für den permanenten Einsatz in beliebigen PC-Systemen entwickelt und getestet worden. Auch bei erhöhtem Industriestandard (-20°C bis +70°C) bleiben die Entropiewerte sehr hoch und unterschreiten die Vorgaben aus den AIS31-Dokumenten nicht. Getestet wurden Applikationen in einem Temperaturbereich von -60 bis +110°C. Alle Grenzwerte der Zufallsrohdaten (Entropie, Halbbytetest) wurden nicht überschritten. Statistische Analysen, auch für diese Temperaturbereiche, befinden sich auf folgendem Link:
http://www.ibbergmann.org/1080799.htm

16 Funktionen der Leuchtdioden

Die auf der Platine befindlichen Leuchtdioden reflektieren die jeweils ablaufenden Funktionen und Zustände des PRG260. Das Blinken der Leuchtdioden erfolgt immer im Sekundentakt. Synchron zur Änderung des Blinkens (ein→aus und aus→ein) wird der permanente Online-Test gestartet.

LED grün	LED gelb	Zustand
Blinkt	Aus	Selbsttest ok, es wird keine Funktion ausgeführt
Blinkt	Ein	Selbsttest ok, eine Funktion wird ausgeführt
Blinkt	Blinkt	Statistischer Fehler im Selbsttest
Ein	Ein	Hardwarefehler im Online-Test festgestellt

17 Kommando-Interface

17.1 Versionsabfrage

Übergabeparameter

0x76

RC:

String mit Version, String-Ende mit 0x0a, 0x0d

17.2 Intensiver Selbsttest

Statistisch bewertet können 1% aller Tests negativ sein.

Übergabeparameter

0x74

RC:

1. Byte	0x55 ok
	0xaa Qualität der Zufallsrohdaten nicht ausreichend
2. Byte	Anzahl der benötigten Versuche (0x05 = ein Versuch, 0x04 = zwei Versuche,..0x01 = fünf Versuche, 0x00 = Fehler)

17.3 Abfrage des Fehlerzählers

Werden beim zyklischen Selbsttest die statistischen Grenzen über- oder unterschritten, wird der Fehlerzähler inkrementiert und kann jederzeit abgefragt werden. Wird der „Intensive Selbsttest" erfolgreich durchlaufen, wird der Fehlerzähler wieder gelöscht, ebenso nach PON.

Übergabeparameter

0x66

RC:

1 Byte Fehlerzähler (0x00..0xff)

17.4 Start der permanenten Zufallsgenerierung

Je nach eingestellter Nachbearbeitung der Zufallsrohdaten (gilt für Klasse PTG.2 und PTG.3, default PTG.3) wird mit der permanenten Zufallsgenerierung begonnen. Beendet werden kann diese Funktion mit Ausgabe des Zeichens 0x65 vom Host.

Übergabeparameter

0x62

RC:

Keinen

Permanente Zufallsgenerierung

Beenden: 0x65

17.5 PTG.2: Keine Nachbearbeitung der Zufallsrohdaten

Übergabeparameter

0x72

RC:

0x72

17.6 PTG.2: Nachbearbeitung der Zufallsrohdaten XOR2

Übergabeparameter

0x32

RC:

0x32

17.7 PTG.2: Nachbearbeitung der Zufallsrohdaten XOR3

Übergabeparameter

0x33

RC:

0x33

17.8 PTG.2: Nachbearbeitung der Zufallsrohdaten vonNeumann

Übergabeparameter

0x6e

RC:

0x6e

17.9 PTG.3: Nachbearbeitung der Zufallsrohdaten mit AES128

Die permanente Ausgabe der Zufallsdaten erfolgt durch Nachbearbeitung der bei jeder Iteration als Klartext erzeugten 16 Byte Zufallsrohdaten mit dem AES128-Algorithmus

Übergabeparameter

0x34

RC:

0x34

17.10 PTG.3: Ausgabe einer definierten Anzahl von Zufallsdaten

Mittels eines übergebenen Parameters können ein Vielfaches von 16 Byte Blöcken erzeugt und ausgegeben werden. Diese Funktion wird sofort gestartet und nach Abarbeitung der gewünschten Anzahl der 16-Byte-Blöcke automatisch beendet.

Übergabeparameter

0x73

1 Byte Anzahl der Ausgabebyte *16 (Wertebereich 0x01..0xff)

RC:

Je nach übergebenem Parameter 16..4080 Byte Zufallsdaten

17.11 Key-Management: Allgemeines

Unabhängig von der Generierung der Zufallsdaten ist ein Key-Management implementiert, dass folgende Funktionalitäten ermöglicht:

1. Eingabe und Speicherung von zwei getrennten Informationen (i.a. Schlüssel1 und Schlüssel2) mit einer Länge von 192 Byte pro Information
2. Eingabe und Speicherung einer Master- und User-Pin
3. Bei richtiger Authentisierung ist nur die Ausgabe von Schlüssel 1 möglich
4. Werden Master- oder User-Pin mehr als fünf Mal hintereinander falsch eingegeben, werden weitere Eingaben nur noch mit einer Fehlermeldung quittiert. Dieser Zustand wird nur durch ein Reset (Power-on) aufgelöst

17.12 Key-Management: Initialisierung (Funktion1)

Initialisierung in gesicherter Umgebung

Übertragen und Speichern im auslesegeschützten EEPROM des ATmega8 von

- 192 Byte Schlüssel 1
- 192 Byte Schlüssel 2
- 8 Byte User-PIN
- 8 Byte Master-PIN

Diese Funktion kann nur **einmal** erfolgreich durchgeführt werden! Für eine erneute Initialisierung ist die Funktion 6 erforderlich. Alle Parameter müssen innerhalb von 5 Sekunden übertragen werden. Die Programmierung dauert ca. 4 Sekunden.

Übergabeparameter:

0x99 (Grundinitialisierung, alle Schlüssel und Pins werden gespeichert)

192 Byte Schlüssel 1

192 Byte Schlüssel 2

8 Byte User-PIN

8 Byte Master-PIN

1 Byte Prüfsumme (XOR über alle Schlüssel-Bytes und PINs)

RC:

0x99

2.Byte:

0x55 ok

0x66 bereits erfolgreich durchgeführt

0xaa Fehler (Zeitüberschreitung bei Eingabe oder Prüfsummenfehler)

17.13 Key-Management: User-Pin ändern (Funktion2)

User-PIN ändern

Um eine User-Pin zu ändern, ist die Eingabe von Master-Pin und neuer User-Pin notwendig. Die Rückmeldung bei Fehlereingabe erfolgt mit 1 Sekunde verzögert, nach 5 Fehler erfolgt ohne Prüfung nur noch eine Fehlermeldung im RC, nach fehlerfreier Funktion bei max. 4 Fehleingaben wird der Fehlerzähler zurückgesetzt. Alternativ wird der Fehlerzähler nach jedem PON rückgesetzt.

Übergabeparameter:

0x28 (User-PIN überschreiben)

8 Byte Master-PIN

8 Byte neue User-PIN

1 Byte Prüfsumme (XOR über alle PINs)

RC:

0x28

2. Byte:

0x55 ok

0xaa Fehler (1 Sekunde verzögert)

17.14 Key-Management: Ausgabe Schlüssel 1 (Funktion3)

Ausgabe von Schlüssel 1

Die Rückmeldung bei einem Fehler erfolgt mit 1 Sekunde verzögert, nach 5 Fehler erfolgt ohne Prüfung nur noch eine Fehlermeldung im RC, nach fehlerfreier Funktion bei max. 4 Fehleingaben wird der Fehlerzähler zurückgesetzt. Alternativ wird der Fehlerzähler nach jedem PON rückgesetzt.

Übergabeparameter:

0x29

8 Byte User-PIN

1 Byte Prüfsumme (XOR über die PIN)

RC:

0x29

2. Byte:

0x55 ok

oder 0xaa Fehler (bei Fehler 1 Sekunde verzögert)

192 Byte Schlüssel 1, wenn User-PIN ok (2.Byte vom RC muss 0x55 sein)

17.15 Key-Management: Verifizierung Schlüssel 2 (Funktion4)

Übergebene 192 Byte werden mit Schlüssel 2 verglichen

Die Rückmeldung bei Fehler erfolgt mit 1 Sekunde verzögert, nach 5 Fehler erfolgt ohne Prüfung nur noch eine Fehlermeldung im RC, nach fehlerfreier Funktion bei max. 4 Fehleingaben wird der Fehlerzähler zurückgesetzt, Alternativ wird der Fehlerzähler nach jedem PON rückgesetzt.

Übergabeparameter:

0x2a

8 Byte User-PIN

192 Byte Schlüssel

1 Byte Prüfsumme (XOR über Schlüssel und PIN)

RC:

0x2a

2. Byte:

0x55 ok

0xaa Fehler (1 Sekunde verzögert)

17.16 Key-Management: Zustandsabfrage (Funktion5)

Zustandsabfrage EEPROM

Übergabeparameter

0x9a

RC:

0x9a

2.Byte:

0x55 Speicher frei

0x66 Speicher belegt

17.17 Key-Management: Parameter löschen (Funktion6)

Speicherbereich von Schlüssel 1, Schlüssel 2, User-Pin und Master-Pin im EEPROM nach Eingabe der Master-Pin vollständig löschen

dauert ca. 4 Sekunden

Die Rückmeldung bei einem Fehler erfolgt mit 1 Sekunde verzögert, nach 5 Fehler erfolgt ohne Prüfung nur noch eine Fehlermeldung im RC, nach fehlerfreier Funktion bis max. 4 Fehleingaben wird der Fehlerzähler zurückgesetzt. Alternativ wird der Fehlerzähler nach jedem PON rückgesetzt.

Übergabeparameter

0x39

8 Byte Master-PIN

1 Byte Prüfsumme (XOR über die PIN)

RC:

0x39

2. Byte

0x55 ok, EEPROM gelöscht

0xaa Fehler (1 Sekunde verzögert)

18 Literatur

[1] Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and General Model; CCMB-2012-09-001, Version 3.1, Revision 4, September 2012

[2] Common Criteria for Information Technology Security Evaluation, Part 2: Security Functional Components; CCMB-2012-09-002, Version 3.1, Revision 4, September 2012

[3] Common Criteria for Information Technology Security Evaluation, Part 3: Security Assurance Requirements; CCMB-2012-09-003, Version 3.1, Revision 4, September 2012

[4] Common Methodology for Information Technology Security Evaluation, Evaluation Methodology; CCMB-2012-09-004, Version 3.1, Revision 4, September 2012

[5] AIS20: Functionality classes and evaluation methodology for deterministic random number generators, Version 2.1, 02.12.2011, Bundesamt für Sicherheit in der Informationstechnik

[6] AIS31: Functionality classes and evaluation methodology for true (physical) random number generators, Version 2.1, 02.12.2011, Bundesamt für Sicherheit in der Informationstechnik

[7] Killmann, W. Schindler, „A proposal for: Functionality classes for random number generators“, Version 2.0, September 18, 2011

[8] Evaluation of random number generators, Version 0.8, Bundesamt für Sicherheit in der Informationstechnik