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

# BENUTZERHANDBUCH

des Physikalischen Zufallsgenerators PRG220

Version 1.0

**Autor:** **Frank Bergmann**

**Letzte Änderung:** **11.11.2017 10:41**

# 1 Inhaltsverzeichnis

1 Inhaltsverzeichnis ..... 2
2 Copyright ..... 3
3 Bedeutung von Zufallszahlen ..... 4
4 Starke Entropie für kryptografische Applikationen ..... 4
5 PRG220 ..... 5
6 Technische Daten ..... 7
7 Genutzte GPIO des Raspberry-Pi ..... 7
8 Prinzip der Rauscherzeugung des PRG220 ..... 8
9 Generierung des Zufallssignals ..... 9
10 Entropie ..... 9
11 Generierung von Klassen ..... 10
12 Sicherheitsfunktionen ..... 11
12.1 Tot-Test ..... 11
12.2 Permanenter Online-Test ..... 11
13 Statistische Qualität ..... 12
14 Sicherheitshinweise ..... 12
15 Anwendungen ..... 13
16 Einsatzumgebung ..... 13
17 Funktionen der Leuchtdioden ..... 13
18 Literatur ..... 14

# 2 Copyright



IBB Ingenieurbüro Bergmann

Sonnenweg 3

D-15537 Grünheide



Der rechtmäßige Erwerb des Physikalischen Zufallsgenerators PRG220 erlaubt eine Nutzung ausschließlich entsprechend Lizenzvertrag.

### 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.

# 4 Starke Entropie für kryptografische Applikationen

Mit dem PRG220 steht ein professioneller Zufallsgenerator der Klasse PTG.3 (hybrider Zufallsgenerator) für die permanente Generierung von kryptografisch sicherem Zufall zur Verfügung. Dieser Zufallsgenerator hat ein UART-Interface und **arbeitet ohne Kommando-Interface**. Nach PON werden kontinuierlich Zufallszahlen mit hoher Geschwindigkeit ausgegeben. Ein permanent im Hintergrund laufendes Sicherheitssystem garantiert, dass bei Ausfall oder Manipulation der Rauschquellen die Zufallsausgabe sofort eingestellt und solange weiter getestet wird, bis alle Qualitätskriterien wieder eingehalten werden.

Die generierten Zufallszahlen sind garantiert kryptografisch sicher und werden vorzugsweise zur Entropieerhöhung und -bereitstellung in Sicherheitsapplikationen verwendet. Unter Linux gibt es bekanntlich immer wieder Probleme mit der Bereitstellung von Zufall im Entropie-Pool. Besonders kritisch wird die Situation, wenn keine Eingabegeräte an einem Server zur Verfügung stehen.

Die Qualität der vom PRG220 erzeugten Zufallszahlen ist qualitativ unvergleichlich höher und sicherer, als jede andere Art der Zufallserzeugung. Zum Verifizieren dieser Aussage stehen diverse Analysen des Zufallsgenerators, auch unter erhöhter thermischer Belastung, zur Verfügung. Die hohe Ausgabegeschwindigkeit des

PRG220 ermöglicht eine Füllung des Entropie-Pools (4096 Bit) in 72ms. Eine kryptografische Nachbearbeitung der ausgegebenen Zufallsdaten ist prinzipiell nicht erforderlich.

## 5 PRG220

Bei dem PRG220 handelt es sich um einen physikalischen Zufallsgenerator mit einer UART-Schnittstelle mit TTL-Pegel. Der PRG220 unterstützt die professionelle Generierung von kryptografisch sicheren Zufallszahlen der Klasse PTG.3. Vorzugsweise ist der PRG220 für den Einsatz auf den Raspberry-Pi-Boards vorgesehen, kann aber auch auf jeder anderen Pin-kompatiblen Plattform eingesetzt werden.

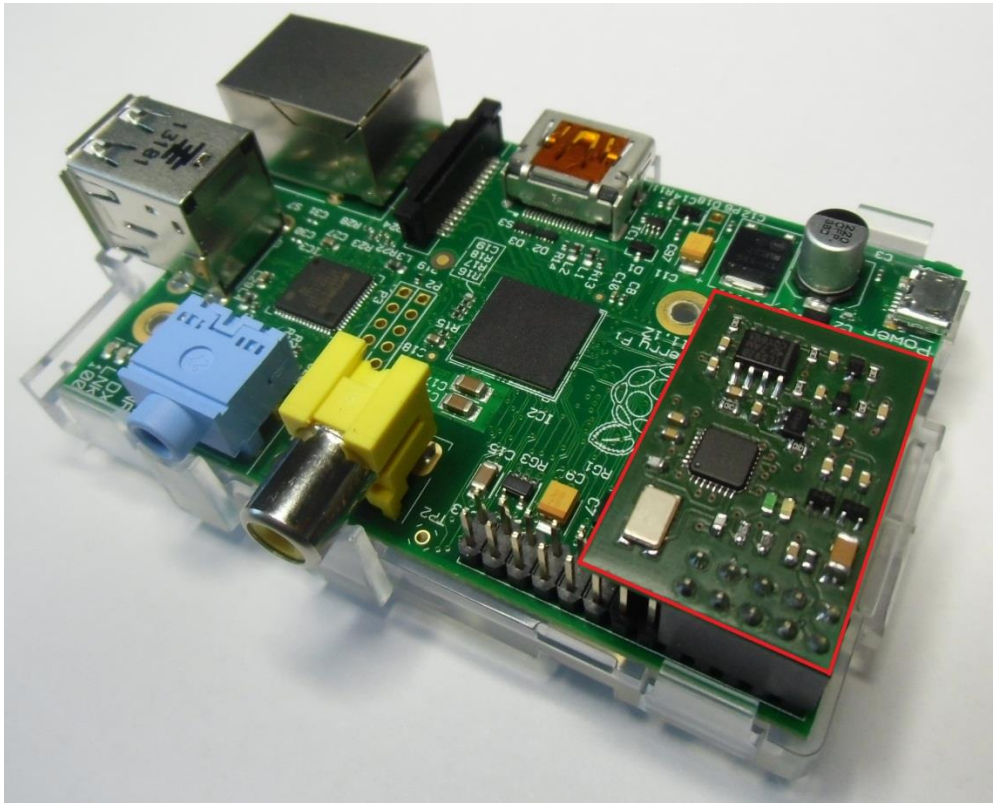


Abbildung: PRG220 auf einem Raspberry-Pi-Board

Kern des PRG220 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

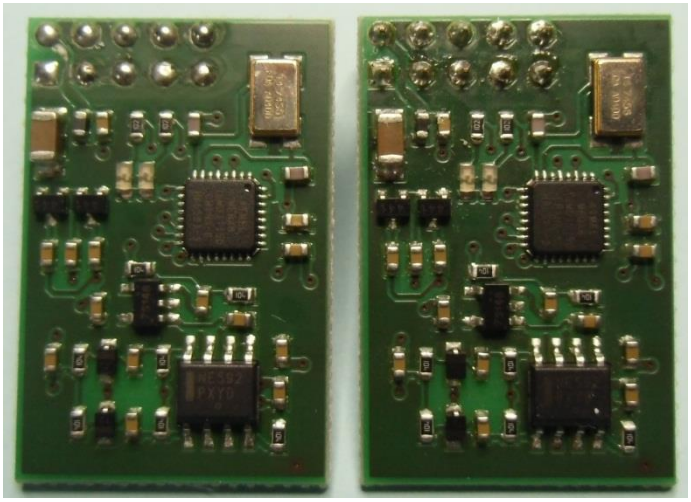


Abbildung: PRG220-Module

Grundlage der Entwicklung des im Folgenden vorgestellten PRG220 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 2016" 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 2016.

# 6 Technische Daten

| | |
|---|---|
| Abmessungen: | 45*19*10 mm (mit Pfostensteckverbinder) |
| Versorgungsspannung: | 5V (+/- 10%) |
| Stromaufnahme: | max. 35mA |
| Temperaturbereich: | funktionell und statistisch stabil von -20°C..+85°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 |
| Entropie: | >7,997 Bits/Byte (ermittelt aus Zufalls-Rohdaten nach Shannon) |
| 0/1-Verhältnis: | garantiert im Bereich 0,49..0,51 |
| Entropie-Pool füllen: | 4096 Bit in 72ms |

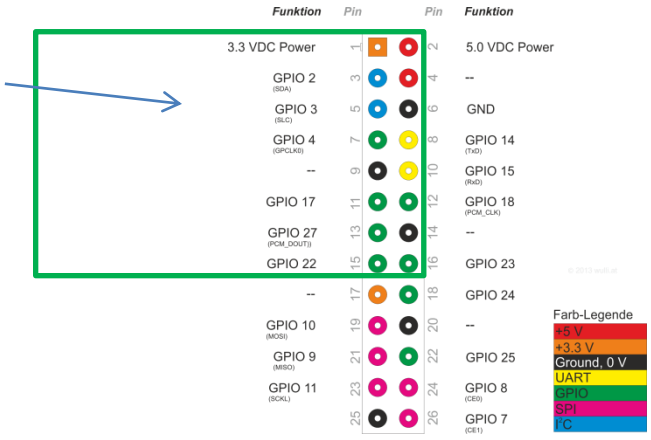
# 7 Genutzte GPIO des Raspberry-Pi

Der PRG220 nutzt folgende Pins des Raspberry-Boards:

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

PRG220-Modul

(Draufsicht)



# 8 Prinzip der Rauscherzeugung des PRG220

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.

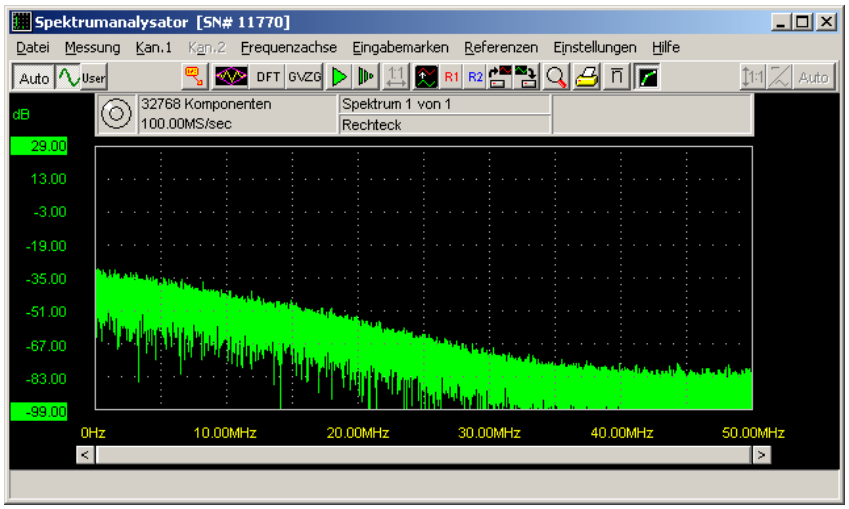



Abbildung: Typisches 1/f-Rauschen nach der Verstärkung

## 9 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 PRG220:

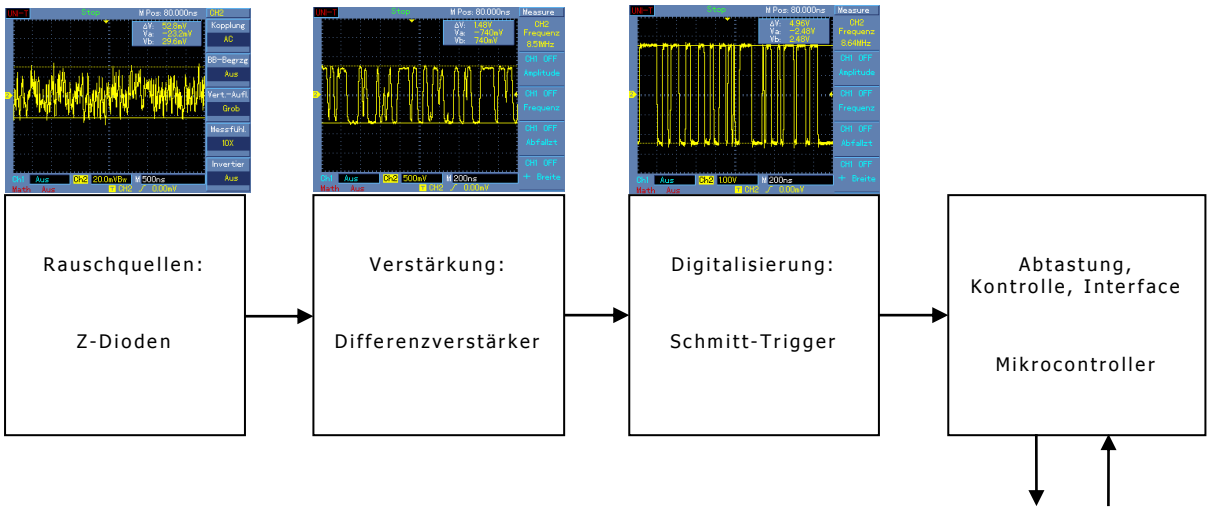



Abbildung: Rauschsignale und Blockschaltbild des PRG220

## 10 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 PRG220-Applikationen zeigen beispiellose Werte.

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

| PRG220 | 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 |

## 11 Generierung von Klassen

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

- Klasse PTG.3 (hybrider Zufallsgenerator)

ausgelegt.

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.

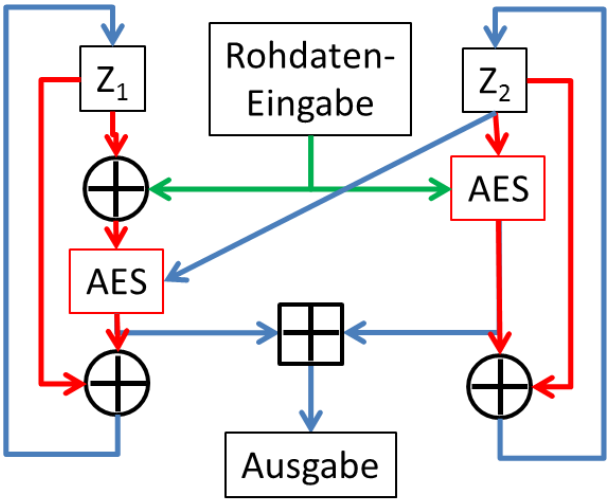



Abbildung: Schema der Nachbearbeitung

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. Da Eingabe und Ausgabe jeweils 16 Byte (=128 Bit) betragen, wird explizit kein Pseudozufall generiert.

Da alle Zufallszahlen gleichverteilt sind, ist keine Synchronisation mit dem Empfänger erforderlich.

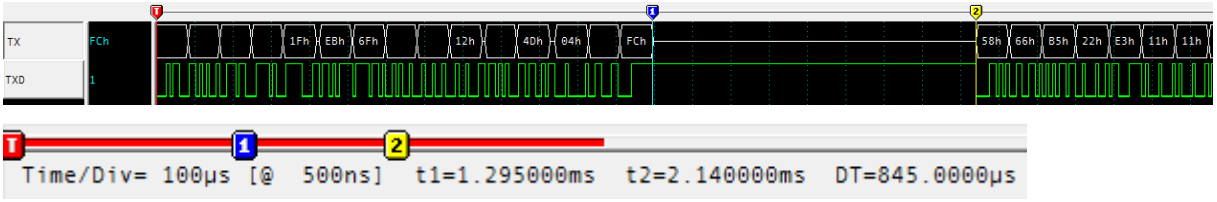



Abbildung: Bearbeitungszeit von Schema in 845µs (=128 Bit)

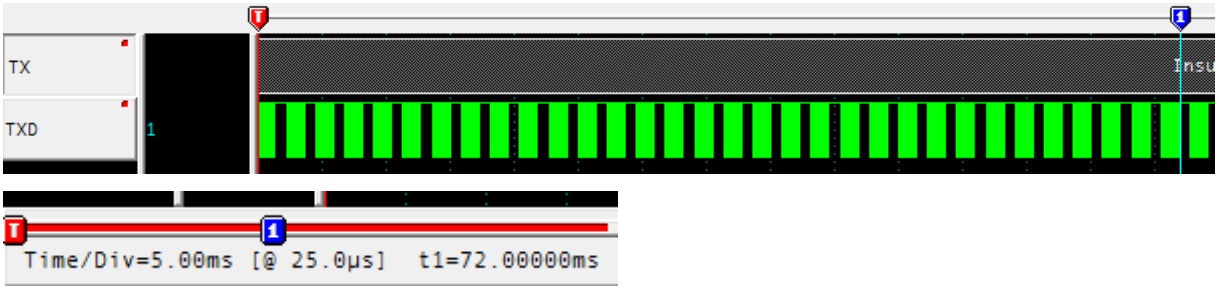



Abbildung: Ausgabe von 4096 Bit (32x 128Bit-Blöcke) in 72ms

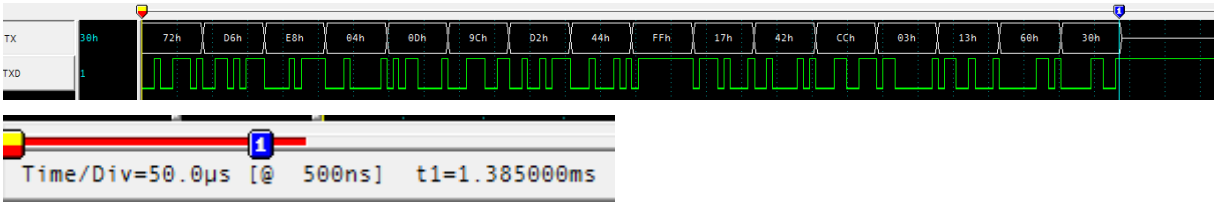



Abbildung: Ausgabe eines 128Bit-Blocks in 1,385ms

# 12 Sicherheitsfunktionen

## 12.1 Tot-Test

Es ist ein Kontrollsystem installiert, welches das digitalisierte Rauschsignal am Anschluss des Mikrocontrollers überwacht. Unmittelbar vor jeder Byte-Generierung des digitalisierten Rauschsignals 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 (entspricht 125 KHz) diese Bedingung nicht erfüllt, wird eine Fehlermeldung generiert, die zur sofortigen Einstellung aller Zufallsausgaben führt
- Dieser Zustand ist nur durch ein positives Ergebnis des weiter laufenden Tot-Test 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

## 12.2 Permanenter Online-Test

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

- Sind die Werte innerhalb der statistischen Vorgaben, wird kein Fehler generiert
- Ist einer der 16 Werte im Halbbytetest gleich Null, wird von einem Totalausfall mindestens einer Rauschquelle ausgegangen und jede Zufallsausgabe blockiert.
- Dieser Zustand ist nur durch ein positives Ergebnis des weiter laufenden Online-Test aufzulösen

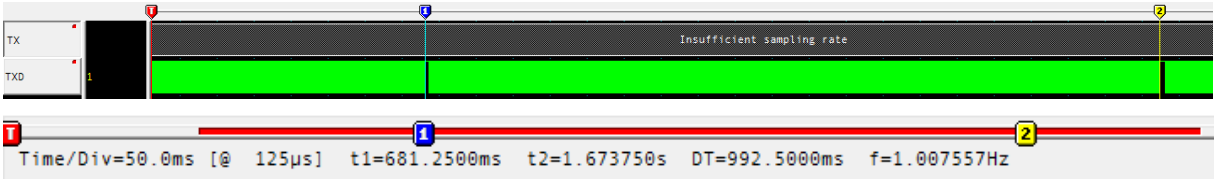



Abbildung: permanenter Online-Test pro Sekunde

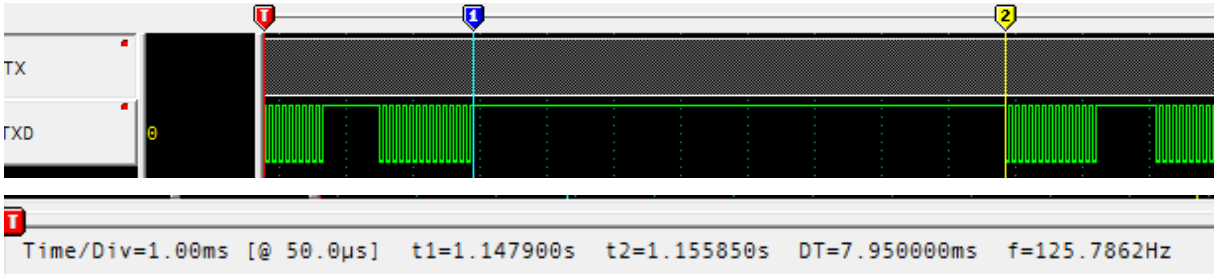



Abbildung: Bearbeitung Online-Test (7,95ms – Schema 845µs =) 7,105ms

## 13 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.

## 14 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

## 15 Anwendungen

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 PRG220:

- 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

## 16 Einsatzumgebung

Der PRG220 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

## 17 Funktionen der Leuchtdioden

Die auf der Platine befindlichen Leuchtdioden reflektieren die jeweils ablaufenden Funktionen und Zustände des PRG220. 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 | Ein | Selbsttest ok, Zufallsdaten werden ausgegeben |
| Ein | Ein | Hardwarefehler im Online-Test festgestellt |

## 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

[9] https://de.wikipedia.org/wiki//dev/random

[10] **Dokumentation und Analyse des Linux ... - BSI**