nRF51 シリーズ　リファレンスマニュアル　（翻訳）

Version 3.0

目次

1. 改版履歴
2. 本ドキュメントについて
3. システム概要
4. CPU
5. メモリー
6. 不揮発メモリーコントローラー (NVMC)
7. ファクトリーインフォメーション　設定レジスター (FUCR)
8. ユーザーインフォメーション設定レジスター (UICR)
9. メモリー保護ユニット (MPU)
10. 周辺装置インターフェイス
11. デバッガーインターフェイス (DIF)
12. パワーマネジメント (POWER)
13. クロックマネジメント (CLOCK)
14. 汎用IO (GPIO)
15. GPIO タスク・イベント (GPIOTE)
16. プログラマブル周辺装置相互接続 (PPI)
17. 2.4 GHz 無線 (RADIP)
18. タイマー／カウンター (TIMER)
19. リアルタイムカウンター (RTC)
20. ウォッチドッグタイマー (TDT)
21. 乱数発生器 (RNG)
22. 温度センサー (TEMP)
23. AES EBCモード暗号化 (ECB)
24. AES CCMモード暗号化 (CCM)
25. 促進的アドレス解決 (AAR)
26. シリアルペリフェラルインターフェイス (SPI) マスター
27. シリアルペリフェラルインターフェイス (SPI) スレーブ
28. I2C互換 ２線インターフェイス (TWI)
29. 汎用非同期送受信機 (UART)
30. クアッドレーターデコーダー (QDEC)
31. A/Dコンバーター (ADC)
32. 低出力コンパレーター (LPCOMP)
33. ソフトウェア割り込み (SWI)

# 改版履歴

# 本ドキュメントについて

# システム概要

* 1. 要約

（略）

* 1. ブロック図

　本ブロック図はシステム全体を図示しています。白い矢印は他のシグナルと物理ピンによって共有されるシグナルを示します。

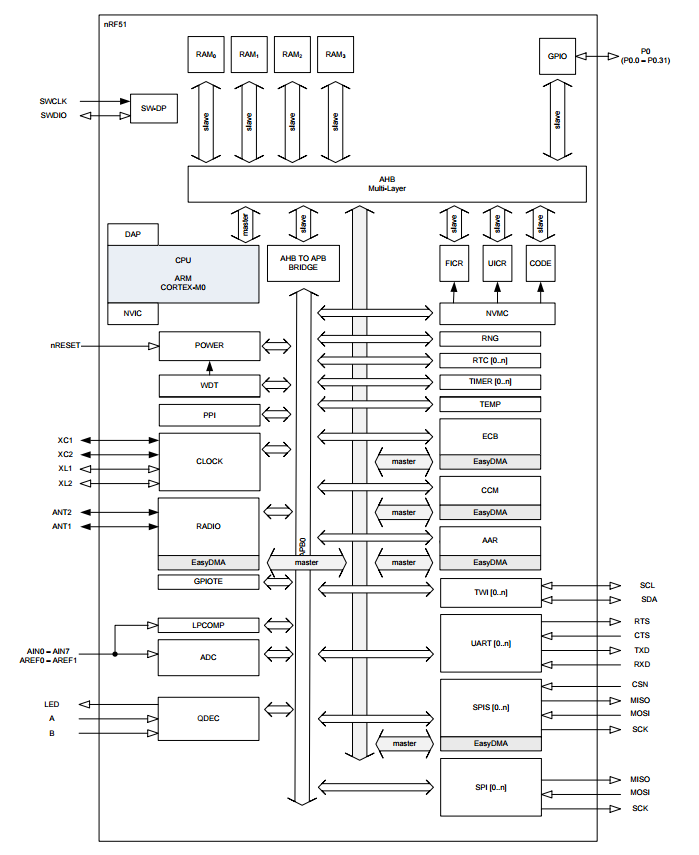


図１：ブロック図

* 1. システムブロック

　本章では、nrF51シリーズを構成する主なブロックについて解説します。

* + 1. ARM Cortex-M0

（略）

* + 1. 2.4 GHz 無線

（略）

* + 1. パワーマネジメント

　nRF51 シリーズの電源管理システムは、シンプルなON・OFFモードにより装置全体を管理する、非常に柔軟かつ独立したシステムです。システムOFFモードでは、全ての電源が落とされますが、RAM領域は保つことができます。リセットか全てのGPIOからの起動によりデバイスの状態をシステムONにすることができます。システムONモードでは、全ての機能ブロックにアクセス可能であり、それぞれの機能ブロックをIDLEモードにしておき、要求のあった場合にのみRUNモードに入ります。

* + 1. PPIシステム

　プログラマブル・ペリフェラル・インターコネクト（PPI）により、異なるペリフェラルがタスクとイベントを使用し、CPUを使用することなく、自動的に相互作用することが可能になります。

* + 1. デバッガーサポート

（略）

1. **CPU**

　低電力32bit CPU、ARM Cortex-M0 が全ての nRF51 シリーズのデバイスに組み込まれています。 ARM Cortex-M0 には 16bit の命令、32bit に拡張された(Thumb-2 technology) があり、高密度のコードかつ小メモリーフットプリントのデバイスです。1サイクルでの32bit乗算器、3ステージパイプライン、ネスト型ベクター割り込みコントローラー (NVIC) が、ARM Cortex-M0 CPUを高効率かつシンプルなプログラム実行を可能にしています。

　nRF51のデーターアライメントはリトルエンディアンです。

　ARM Cortex-M プロセッサーシリーズに実装されている、ARM Cortex マイクロコントローラー・ソフトウェア・インターフェース・スタンダード (CMSIS)ハードウェア抽象レイヤー は、M0 CPUにおいても利用可能です。コードはARM Cortex-M3ベースのデバイスにおいて前方互換性を持ちます。

　ARM Cotex-M0 CPU に関しての、より詳しい情報は、ARM Cotex M0を参照してください。

1. **メモリー**
2. **不揮発メモリーコントローラー (NVMC)**
   1. **機能概要**

　不揮発メモリーコントローラー (NVMC) は不揮発メモリー (NVM) への書き込みと消去に使用されます。

　書き込み操作が事項される前に、CONFIG.WENにより、NVMは書き込みが有効にされる必要があります。同様に消去が実行される前に、CONFIG.EENにより、NVMは消去を有効にされる必要があります。ユーザーは書き込みと消去が同時に有効にされることが無いことを確認しなくてはなりません。さもなくば、予期せぬ動作が発生する可能性があります。

* + 1. NVM への書き込み

　書き込みが有効な場合、NVMはCODEもしくはUICRに対し、アラインされたアドレスでワード単位の書き込みが行われます。NVMC はNVMが消去された領域、つまり1にセットされているビットにのみ書き込みが可能です。

　NVMにワードを書き込むのに必要な時間は製品仕様書のtWRITEに言及があります。CPUはNVMCがNVMに書き込んでいる間、停止します。

　ワード単位にアラインされた書き込みのみが許可されています。バイトや半ワードにアラインされた書き込みは、ハードフォルトを引き起こします。

* + 1. ユーザー・インフォメーション・設定レジスターへの書き込み

　UICR レジスターは通常の不揮発メモリーです。UICRが書き込まれたのち、新しいUICR設定は、リセット後にのみ有効になります。

* + 1. 全消去

　消去が有効な場合、すべての CODE と UICR は ERASEALL レジスターを使用した一度の操作で削除されることができます。ERASEALL はファクトリー・インフォメーション・設定レジスター (FICR) を消去することはありません。

* + 1. code region 1 内のページ消去

　消去が有効な場合、 NVM は ERASEPAGE レジスターもしくは ERASEPCR1 レジスターを使用することにより、ページごとに消去されることができます。消去後には NVM ページのすべてのページ内のビットは 1 にセットされます。ページを消去するために必要な時間は製品仕様書のtPAGEERASEに記述されています。NVMC が消去操作を行っている間 CPU は停止します。より詳細な情報は UICR の章を参照してください。

* + 1. code region 0 内のページ消去

　ERASEPCR0 は code region 0 のページを消去するために使用されます。 ERASEPCR0 レジスターは core region 0 を実行中のプログラムによりのみアクセスすることができます。

　code region 0 を実行しているプログラムが不揮発性記憶を有効にするため、このプログラムは全てのコードページの消去及び再書き込みが可能です。これはcode region 0 内におけるこの目的のため指定します。 ERASEPCR0 はこの目的のために使用されます。ERASEPCR 0 レジスターはその使用にMPUによって強制された、code region 0 から実行されるコードのみ書き換えが可能と言う、制限を持っています。code region 0 を実行しているプログラムはERASEPCR1 を使用することにより、code region 1 内のページを消去することができます。

　ページを消去するために必要な時間は製品仕様書のtPAGEERASEに記述されています。

* 1. **レジスター概要**

（略）

* 1. **レジスター詳細**

（略）

1. **ファクトリーインフォメーション　設定レジスター (FUCR)**
2. **ユーザーインフォメーション設定レジスター (UICR)**
   1. **機能概要**

　ユーザー・インフォメーション・設定レジスターはユーザー特有の設定のための不揮発性メモリーのレジスタです。

ｄ：　コード読み取りは全てのコードエリアに対し保護されていますが、一部のコードエリアはUICRとして、設定され有効化されます。UICRはERASEALLによってのみ、削除可能です。

　コードエリアは code region 0 (CR0) と code region 1 (CR1) の２つの領域に分割することができます。code region 0 は0x00000000 から開始され、CLENR0 レジスターとして示されるコード領域まで拡張されます。CLENR0を超えた領域core region 1として定義されます。CLENR0が定義されていない場合、つまり値が0xFFFFFFFFFの場合、全コード領域はcode region 1(CR1)として、定義されます。

　コードは code region 1 より実行され、code region 0に書き込まれることはありません。加えて、core region 0 が読み出し保護の場合、code region 0 は core region 1 により実行されているコードや、SWDインターフェイスからは読み込まれることはありません。詳細は RBPCONF の PR0 を確認してください。

　UICRを通して設定される、主読み出し保護機能は、 code region 0 と code region 1 の全コードを保護します。

　NVMC の PASSASGE コマンドは、code ragion 1 にのみ作用します。コード領域とUICRへの消去とプログラムの実施方法はNVMCの章を参照してください。

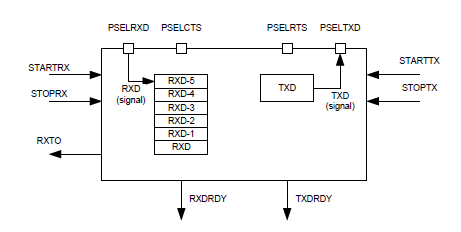
* 1. **レジスター概要**

（略）

* 1. **レジスター詳細**

（略）

1. **メモリー保護ユニット (MPU)**
2. **周辺装置インターフェイス**
3. **デバッガーインターフェイス (DIF)**
4. **パワーマネジメント (POWER)**
5. **クロックマネジメント (CLOCK)**
6. **汎用IO (GPIO)**
7. **GPIO タスク・イベント (GPIOTE)**
8. **プログラマブル周辺装置相互接続 (PPI)**
9. **2.4 GHz 無線 (RADIP)**
10. **タイマー／カウンター (TIMER)**
11. **リアルタイムカウンター (RTC)**
12. **ウォッチドッグタイマー (WDT)**
13. **乱数発生器 (RNG)**
14. **温度センサー (TEMP)**
15. **AES EBCモード暗号化 (ECB)**
16. **AES CCMモード暗号化 (CCM)**
17. **促進的アドレス解決 (AAR)**
18. **シリアルペリフェラルインターフェイス (SPI) マスター**
19. **シリアルペリフェラルインターフェイス (SPI) スレーブ**
20. **I2C互換 ２線インターフェイス (TWI)**
21. **汎用非同期送受信機 (UART)**



**Figure 67: UART configuration**

**29.1 機能概要**

UARTは以下の機能をサポートしています。

• 全２重動作

•自動フロー制御

• Parity checking and generation for the 9th data bit　9番目のデータービットへのパリティチェック及び生成

本UARTの構造は本ページFigure 67となります。 本UARTはTXDおよびRXDを直接利用し、データ送信およびデータ受信受信します。 本UARTは1つのストップビットを用います。

**29.2 ピン　コンフィグレーション**

　UARTに関連するピンは、それぞれ信号、RXD、CTS(Clear To Send, アクティブLOW)、RTS（Request To Send, アクティブLOW）、TXDが、PSELRXD、PSELCTS、PSELRTS及びPSELTXDのレジスタ設定によって、物理ピンにマッピングされます。

If a value of 0xFFFFFFFF is specified in any of these registers, the associated UART signal will not be connected to any physical pin. The PSELRXD, PSELCTS, PSELRTS, and PSELTXD registers and their configurations are only used as long as the UART is enabled, and retained only for the duration the device is in ON mode. PSELRXD, PSELRTS, PSELTRTS and PSELTXD must only be configured when the UART is disabled.

To secure correct signal levels on the pins by the UART when the system is in OFF mode, the pins must be configured in the GPIO peripheral as described in Table 273: GPIO configuration on page 151.

Only one peripheral can be assigned to drive a particular GPIO pin at a time. Failing to do so may result in unpredictable behavior.

**Table 273: GPIO configuration**

|  |  |  |
| --- | --- | --- |
| **UART pin** | **Direction** | **Output value** |
| **RXC** | **Input** | **Not applicable** |
| **CTS** | **Input** | **Not applicable** |
| **RTS** | **Output** | **1** |
| **TXD** | **Output** | **1** |

**29.3 Shared resources**

The UART shares registers and other resources with other peripherals that have the same ID as the UART. Therefore, you must disable all peripherals that have the same ID as the UART before the UART can be 29 Universal Asynchronous Receiver/Transmitter (UART) Page 152

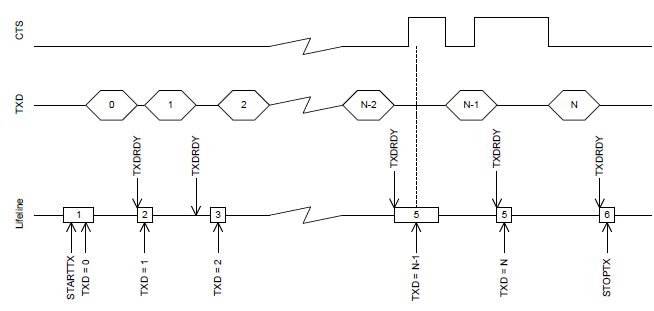
configured and used. Disabling a peripheral that has the same ID as the UART will not reset any of the registers that are shared with the UART. It is therefore important to configure all relevant UART registers explicitly to ensure that it operates correctly.

See the Instantiation table in Instantiation on page 17 for details on peripherals and their IDs.

**29.4 送信**

単体のUART 送信シーケンスはSTARTTXタスクを要因して開始されます。 複数バイトデータはTXDレジスタへの書き込みによって転送されます。1バイト送信成功時、 UARTは次の1バイトを TXD レジスタに書き込み可能状態となった後、TXDRDYイベントを出力します。 UART 送信シーケンスはSTOPTXタスクを要因として速やかに停止されます。

フロー制御が有効な場合、CTSが非アクティブになったとき、および再開されてCTSが再びアクティブになったとき、送信は自動的にサスペンドされます。（Figure 68） CTSが非アクティブにされたとき、送信中の1バイトデータは、転送がサスペンドに入る前に完全に送信されます。



**Figure 68: UART送信**

**29.5 受信**

A UART 受信シーケンスはSTARTRX タスクを要因として開始されます。 The UART 受信 chain implements a FIFO capable of storing six incoming RXD bytes before data is overwritten. Bytes are extracted from this FIFO by reading the RXD register. When a byte is extracted from the FIFO a new byte pending in the FIFO will be moved to the RXD register. The UART will generate an RXDRDY event every time a new byte is moved to the RXD register.

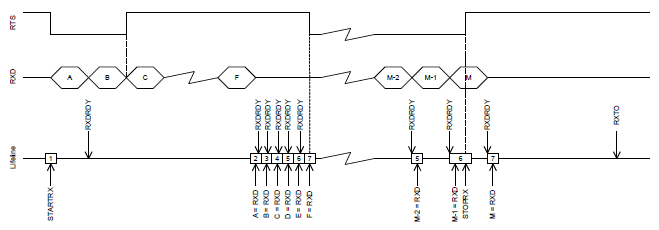
When flow control is enabled, the UART will deactivate the RTS signal when there is only space for four more bytes in the receiver FIFO. The counterpart transmitter is therefore able to send up to four bytes afterthe RTS signal is deactivated before data is being overwritten. To prevent overwriting data in the FIFO, the counterpart UART transmitter must therefore make sure to stop transmitting data within four bytes after the RTS line is deactivated.

The RTS signal will first be activated again when the FIFO has been emptied, that is, when all bytes in the FIFO have been read by the CPU, see Figure 69: UART reception on page 153.

The RTS signal will also be deactivated when the receiver is stopped through the STOPRX task as illustrated in Figure 69: UART reception on page 153. The UART will be able to receive up to four bytes if they are sent in succession immediately after the RTS signal has been deactivated. This is possible because the UART is, even after the STOPRX task is triggered, able to receive bytes for an extended period equal to the 29 Universal Asynchronous Receiver/Transmitter (UART) Page 153 time it takes to send four bytes on the configured baud rate. The UART will generate a receiver timeout event (RXTO) when this period has elapsed.

To prevent loss of incoming data the RXD register must only be read one time following every RXDRDY event.

To secure that the CPU can detect all incoming RXDRDY events through the RXDRDY event register, the RXDRDY event register must be cleared before the RXD register is read. The reason for this is that the UART is allowed to write a new byte to the RXD register, and therefore can also generate a new event, immediately after the RXD register is read (emptied) by the CPU.



**Figure 69: UART reception**

As indicated in occurrence 2 in Figure 69: UART reception on page 153, the RXDRDY event associated with byte B is generated first after byte A has been extracted from RXD.

**29.6 Suspending the UART**

The UART can be suspended by triggering the SUSPEND task. SUSPEND will affect both the UART receiver and the UART transmitter, i.e. the transmitter will stop transmitting and the receiver will stop receiving. UART transmission and reception can be resumed, after being suspended, by triggering STARTTX and STARTRX respectively.

Following a SUSPEND task, an ongoing TXD byte transmission will be completed before the UART is suspended.

When the SUSPEND task is triggered, the UART receiver will behave in the same way as it does when the STOPRX task is triggered.

**29.7 Error conditions**

有効なストップビットがフレームで検出されていない場合、エラーイベントは、フレーミングエラーの形で、生成されます。 RXDラインは、データフレームの長さよりも長いため、アクティブローに保持されている場合は別のエラーイベントは、ブレーク条件の形で、生成されます。ブレーク条件が発生する前に、フレーミングエラーが常に発生します。

**29.8 Using the UART without flow control**

If flow control is not enabled the interface will behave as if the CTS and RTS lines are kept active all the time.

**29.9 Parity configuration**

When parity is enabled, the parity will be generated automatically from the even parity of TXD and RXD for transmission and reception respectively.

**29.10 Register Overview**

**Table 274: Instances**

**Table 275: Register Overview**

|  |  |  |
| --- | --- | --- |
| **レジスタ** | **Offset** | **説明** |
| **Tasks** |  |  |
| **STARTRX** | **0x000** | **UART受信機の開始** |
| **STOPRX** | **0x004** | **UART受信機の停止** |
| **STARTTX** | **0x008** | **UART送信機の開始** |
| **STOPTX** | **0x00C** | **UART送信機の停止** |
| **SUSPEND** | **0x01C** | **UARTの一時中断** |
| **Events** |  |  |
| **CTS** | **0x100** | **CTS is activated (set low). Clear To Send.** |
| **NCTS** | **0x104** | **CTS is deactivated (set high). Not Clear To Send.** |
| **RXDRDY** | **0x108** | **Data received in RXD** |
| **TXDRDY** | **0x11C** | **Data sent from TXD** |
| **ERROR** | **0x11C** | **Error detected** |
| **RXTO** | **0x144** | **Receiver timeout** |
| **Registers** |  |  |
| **INTEN** | **0x300** | **割り込みの利用可／不可** |
| **INTENSET** | **0x304** | **割り込みの有効化** |
| **INTENCLR** | **0x308** | **割り込みの無効化** |
| **ERRORSRC** | **0x480** | **エラー原因** |
| **ENABLE** | **0x500** | **UARTの有効化** |
| **PSELRTS** | **0x508** | **Pin select for RTS** |
| **PSELTXD** | **0x50C** | **Pin select for TXD** |
| **PSELCTS** | **0x510** | **Pin select for CTS** |
| **PSELRXD** | **0x514** | **Pin select for RXD** |
| **RXD** | **0x518** | **RXD register** |
| **TXD** | **0x51C** | **TXD register** |
| **BAUDRATE** | **0x524** | **ボーレート** |
| **CONFIG** | **0x56C** | **パリティ、ハードウェアフロー制御の設定** |

**29.11 Register Details**

**Table 276: INTEN**

**Table 277: INTENSET**

**Table 278: INTENCLR**

**Table 279: ERRORSRC**

**Table 280: ENABLE**

**Table 281: PSELRTS**

**Table 282: PSELTXD**

**Table 283: PSELCTS**

**Table 284: PSELRXD**

**Table 285: RXD**

**Table 286: TXD**

**Table 287: BAUDRATE**

**Table 288: CONFIG**

1. **クアッドレーターデコーダー (QDEC)**
2. **A/Dコンバーター (ADC)**

ｎ

図71: A/D コンバーター

* 1. A/Dコンバーター (ADC)
     1. 入力電圧レンジの設定

　ADCの設定を行い、入力電圧レンジとADC電圧レンジを一致させることは非常に重要です。

　もし、入力電圧レンジが ADC 電圧レンジより低い場合、解像度が十分に活用されなくなります。

　もし入力電圧レンジがADC電圧レンジより高い場合、ADC最大値を超える値はすべて、最大値となってしまいます。これはサチレーションポイント（飽和点）と呼ばれます。

　入力電圧レンジとサチレーションポイントはADCリファレンス電圧と選択されたプリスケールに因ります。もし、1.2 V VBG内部リファレンス電圧を使用する場合、ADCレンジは0～1.2 V（サチレーションポイント1.2 V）となります。これは、AINシグナルにおいて1/1プリスケールを使用している場合、適切な変換結果を得るためには、入力レンジが0～1.2 Vである必要があることを意味しています。1.2 Vを超える入力はADCの最大値に変換されます。

　しかし、もし例えば1/3プリスケールを使用した場合、AINの入力は1/3に縮小します。これによりAINの入力電圧範囲は3.6 V入力が3.6 / 3 =1.2 V に縮小されるため、0～3.6 Vになります。表１に参照電圧とプリスケール設定と対応するサチレーションポイントのADC AIN入力を示します。

表311：サチレーションポイント例

|  |  |  |
| --- | --- | --- |
| **リファレンス** | **プリスケール** | **AIN最大電圧** |
| 1.2 V VBG | 1/1 | 1.2 V |
| 1.2 V VBG | 2/3 | 1.8 V |
| 1.2 V VBG | 1/3 | 3.6 V |
| 1.0 V AREF | 1/1 | 1.0 V |
| 1.0 V AREF | 2/3 | 1.5 V |
| 1.0 V AREF | 1/3 | 3.0 V |
| VDD 3.0V VDD 1/2 | 1/1 | 1.5 V |

**分圧器**

　下記2つのルールがAINピンへの最大入力許可電圧を規定するルールです。

1. ADCにおいてAINピンは、プリスケール後に2.4 Vを超える電圧を印加してはいけません。

入力電圧　×　プリスケール = 最大 2.4 V

1. GPIO ピンにはVDD + 0.3 V より高い電圧を印加してはいけません。nRF51X22 製品仕様の最大定格を参照してください。

　例えば、2/3プリスケールを使用している場合、AIN ピンには、2.4 V / (2/3) = 3.6 V まで印加することができます。そして、ルール２に違反しないためには、VDDは3.3 V 以上である必要があります。

　表２に供給電圧と、プリスケール設定に応じたADCのAINピンに印加可能な最大電圧の例を示します。

表312　AIN最大電圧例

|  |  |  |  |
| --- | --- | --- | --- |
| **供給電圧** | **プリスケール** | **AIN 最大電圧** | **制限ルール** |
| 3.6 V | 1/1 | 2.4 V | ルール１ |
| 3.6 V | 2/3 | 3.6 V | ルール１ |
| 3.6 V | 1/3 | 3.9 V | ルール１ |
| 3.3 V | 1/1 | 2.4 V | ルール１ |
| 3.3 V | 2/3 | 3.6 V | ルール１＆ルール２ |
| 3.3 V | 1/3 | 3.3 V | ルール２ |
| 1.8 V | 1/1 | 2.1 V | ルール２ |
| 1.8 V | 2/3 | 2.1 V | ルール２ |
| 1.8 V | 1/3 | 2.1 V | ルール２ |

　もし、AINに印加可能な最大電圧以上を測定したい場合は、分圧器を使用する必要があります。ページ６、セクション2.4の 『分圧器を使用した電圧降下』を参照してください。

* + 1. 分圧器を使用した電圧降下

　センサー、バッテリーの出力電圧がADCの電圧範囲を超えている場合、ADCの入力ピンに電圧を印加する前に、電圧を下げる必要があります。分圧器を使用してこれを行うことができます。Li-Ionバッテリーから電圧を下げる例を図３に示します。

　ADCの内部インピーダンスのため、大きな抵抗値を持つ分圧器はADCの出力に対して、誤差を生じます。分圧器のインピーダンスが1k#以下の場合、誤差は無視できるほど小さくなります。分圧器のインピーダンスを大きくするほど、誤差も大きくなります。

　しかし、分圧器を流れる電流を小さくするため、分圧器に高い抵抗値を使用したいという望みもあります。高い抵抗値においても誤差を減らすためにはAINとGNDの間にコンデンサーを追加することによって実現できます。大きな値のコンデンサーはADCの出力誤差を減らしますが、代わりにサンプリング周期も減らせることとなります。

　UAINの電圧を下げるため、RAINが120k～400k#の分圧器が接続されている状態でサンプリングする場合。またAIN、GND間にキャパシタが接続されており、AIN電圧をサンプリングの間、十分な時間前回のレベルに保ちR2の高い抵抗値の影響を最小限にすることができます。キャパシタ容量は次回のサンプル時までに満たされるほど小さくなくてはなりません。すなわち、キャパシタが接続されている場合、そのサイズは精確さおサンプリング周期のトレードオフになる。サンプリングを行っていない間は、RAINは高い値であり、それを接続されていないとみなすことができます。

　入力電圧レンジがADC電圧レンジを超えており、高い正確性と高いサンプリング周期を求められる場合、電圧バッファーが必要となります。

　他の可能性としてはFETトランスミッターを電源と分圧器間に接続し、サンプリング前に、一時的にオープンにし、分圧器に電流を流します。分圧器は（1k#以下の）低い抵抗値にすることが可能で、キャパシタを必要としません。分圧器はサンプリング時に比較的大きな電流を消費し、サンプリングをしていないときは全く電流を消費しません。

* + 1. 入力インピーダンス

nRF51822製品仕様に記載されているADC誤差仕様を実現するために、電源に接続の出力インピーダンスは1k#以下でなくてはなりません。出力インピーダンスが1k#以下である利点は、ADC入力において異なるプリスケールの設定においてもADCの正確性に特に影響がないことです。

　もしハイインピーダンスの電源が選択された場合、追加ゲインとオフセット誤差が加わり、これはまた、プリスケール設定の差により変わります。

　図４はADCのサンプリングを行っている、nRF51 ADCの入力モデルで、表５は異なるプリスケールの設定によるRAINの値を示します。内部VBG参照電圧が 1.2 V であれば、ADC内部電源は VBG/2 = 0.6 V になります。

　ADCがサンプリングを行っていないとき、AINピンはとてもハイインピーダンスであり、接続されていないとみなすことができます。表５は異なるプリスケール設定による内部インピーダンスの統計値です。99.7%のデバイス（±3σ）が6.3%以内であると期待されます。たとえば、1/1プリスケールの場合121.5～137.9k#です。

表313：RAINの入力インピーダンス統計値

|  |  |  |
| --- | --- | --- |
| **プリスケール** | **平均インピーダンス** | **標準偏差** |
| 1/1 | 129.7 kΩ | 2.74 kΩ |
| 2/3 | 194.6 kΩ | 4.1 kΩ |
| 1/3 | 389.2 kΩ | 8.2 kΩ |

* + 1. 設定

　入力の選択、参照の選択、分解能、プリスケール、等の全てのパラメーターは CONFIG レジスターを使用し、設定します。

注：ADC変換の実施中(ADC BUSY中)に、ADCの設定を行ってはなりません。

* + 1. 使用方法

　ADC変換はSTARTタスクを使用することにより開始します。STARTタスクはCPUからタスクレジスタに直接書き込む方法、PPIを使用しタスクの実施を開始する方法、どちらでも可能です。

　サンプリングの実施中、ADCはBUSY状態になります。ADCがBUSY状態、もしくはREADY状態なのかは、BUSYレジスタをモニターすることにより判別可能です。

　ADC変換が完了した場合、ENDイベントが生成され、変換結果はRESULTレジスターから読み取ることができます。

　省電力のためADCのアナログ回路は、ADC変換の完了後にパワーダウンします。

* + 1. 1ショット／連続動作

　ADCそれ自身は、１ショット操作のみサポートします。それは、各々の変換はSTARTタスクを使用し、明示的に開始させる必要があることを意味しています。

　しかし、連続的なADC操作は、例えばPPIをタイマーを通し、連続的にSTARTタスクをトリガーすることにより、実施されます。

* + 1. ピン設定

　ユーザーはAIN0からAIN7までのアナログ入力ピンをのうち一つを選択するため、PSELレジスターを使用することができます。デバイスに応じ、どのアナログピンが使用可能かの情報に関する詳細はデバイス製品仕様を参照してください。選択されたアナログピンはENABLEレジスタにより有効化された際にADCに取得されます。アナログピンの選択方法はGPIOの章を参照してください。

* + 1. 共有リソース

ADCは、ADCと同じIDを持つレジスタとペリフェラルのリソースを共有しています。そのため、ユーザーはADCが設定及び使用可能になる前に、ADCと同じIDを持つすべてのペリフェラルを無効化しなくてはなりません。ADCはLPCMPと同じアナログピンを使用しています。そのためADCを有効化する前にLPCOMPを無効化してください。ADCが正しく動作することを保障するために、明示的に関連する全てのADCレジスタを設定することが重要です。

　ペリフェラル及びそのIDの詳細に関してはページ17のInstantation内にある、インスタシエインションテーブルを参照してください。

* 1. レジスター概要

表314：インスタンス

|  |  |  |  |
| --- | --- | --- | --- |
| **ベースアドレス** | **ペリフェラル** | **インスタンス** | **概要** |
| 0x40007000 | ADC | ADC | アナログ・デジタルコンバーター |

表315：レジスター概要

|  |  |  |
| --- | --- | --- |
| **レジスター** | **オフセット** | **概要** |
| ***Tasks*** |  |  |
| START | 0x000 | ADC変換開始 |
| STOP | 0x004 | ADC変換停止 |
| ***Events*** |  |  |
| END | 0x100 | ADC変換完了 |
| ***Registers*** |  |  |
| INTEN | 0x300 | 割り込みの有効化／無効化 |
| INTENSET | 0x304 | 割り込み有効化 |
| INTENCLR | 0x308 | 割り込み無効化 |
| BUSY | 0x400 | ADCビジー（変換中） |
| ENABLE | 0x500 | ADC有効化。ADCが有効化された際に、ADCはCONFIGレジスターにより指定されたアナログ入力ピンへのアクセスを得ます。 |
| CONFIG | 0x504 | ADC設定 |
| RESULT | 0x508 | 前回のADC変換結果 |

* 1. レジスター詳細

（略）

1. **低出力コンパレーター (LPCOMP)**
2. **ソフトウェア割り込み (SWI)**

**訳：プログレス・テクノロジーズ**

ソフト１０　黒田<[e-kuroda@progresstech.jp](mailto:e-kuroda@progresstech.jp)>

追訳、追記、検証、修正歓迎