megaAVRマイクロ コントローラ

AT13041 : IoT用AVR – ATWINC1500 802.11 b/g/n WiFi網制御器単位部とのmegaAVRの使用

応用記述

説明

この応用記述はAndroid応用経由でLEDを制御するのにATWINC1500 Xplained Pro WiFi®ネットワーク制御器とでAtmel® ATmega328Pを使う方法を記述します。

Rev. 42552A-09/2015, 42552AJ1-01/2021

目次

1. 序説

この応用記述と関連する例ﾌﾟﾛｼﾞｪｸﾄはATWINC1500 Xplained Proと共にATmega328P Xplained Miniを用いてWiFi1制御されたLEDのｿﾌﾄｳｪｱとﾊｰﾄﾞｳｪｱの実装を示します。これは温度測定を送信するのに応用がどう拡張されるかも記述されます。

関連する例ﾌﾟﾛｼﾞｪｸﾄはatmel.comから.zipﾌｧｲﾙとして入手可能で、Atmel StudioとAtmelｿﾌﾄｳｪｱ枠組み(ASF:Atmel Software Framework)を使うことによって実装されます。

1.1. 事前必要条件

この応用記述で記述される例の走行には以下を必要とします。

- Atmel Studio 6.2 ｻｰﾋﾞｽﾊﾟｯｸ2
- ATmega328P Xplained Mini
- ﾌｧｰﾑｳｪｱ19.3を持つATWINC1500 Xplained Pro
- I/O1 Xplained Pro
- Android応用を走らせるための装置
- 半田ごてと半田
- 線材
- 0Ω抵抗器
- 2つのL型100mil(2.54mm)ﾍｯﾀﾞ
- 単ﾋﾟﾝ ﾍｯﾀﾞ

2. 例概要

以下の項はATWINC1500 Xplained ProとI/O1 Xplained Pro拡張部を接続するためにATmega328P Xplained Miniをどう変更するかを示します。例ﾌﾟﾛｼﾞｪｸﾄとAndroid応用も詳述されます。

例ﾌﾟﾛｼﾞｪｸﾄはAndroid応用によって発見されるべきXplained Miniへ与えられた間隔でUDP(使用者ﾃﾞｰﾀｸﾞﾗﾑ規約,User Datagram Protocol)ﾊﾟｹｯﾄを送信します。応用はその後にTCP(伝送制御規約,Transmission Control Protocol)経由で接続してI/O1 Xplianed Pro上のLEDを交互切り替えするためのﾒｯｾｰｼﾞを送ることができます。

2.1. ﾊｰﾄﾞｳｪｱ

以下の項は各種Xplained Pro単位部の例の使い方だけでなく、互いにそれらを接続するためにXplained Mini基板をどう変更するかも説明します。

2.1.1. ATmega Xplained Mini

Atmel ATmega328P Xplained Mini評価ｷｯﾄはATmega328Pﾏｲｸﾛ ｺﾝﾄﾛｰﾗを評価するためのﾊｰﾄﾞｳｪｱ基盤です。この評価ｷｯﾄはAtmel Studio 6.2またはそれ以降版との繋ぎ目のない統合を提供する完全に統合されたﾃﾞﾊﾞｯｶﾞと共にやって来ます。他の全てのmegaAVR® Xplained Miniｷｯﾄもこの応用記述で記述される実演応用に使うことができます。

2.1.2. I/O1 Xplained

I/O1 Xplained Proは光感知器、温度感知器、ﾏｲｸﾛSDｶｰﾄﾞを提供します。これはどれかのXplained Pro基板の拡張ﾍｯﾀﾞに接続します。

2.1.3. ATWINC1500 Xplianed

Atmel ATWINC1500 XplainedはATWINC1500安価、低電力802.11 b/g/n WiFi網制御器単位部を評価するためのﾊｰﾄﾞｳｪｱ基盤です。Atmel Studio統合開発環境によって支援され、このｷｯﾄはATWINC1500の機能への容易なｱｸｾｽを提供して独自設計にﾃﾞﾊﾞｲｽを統合する方法を説明します。

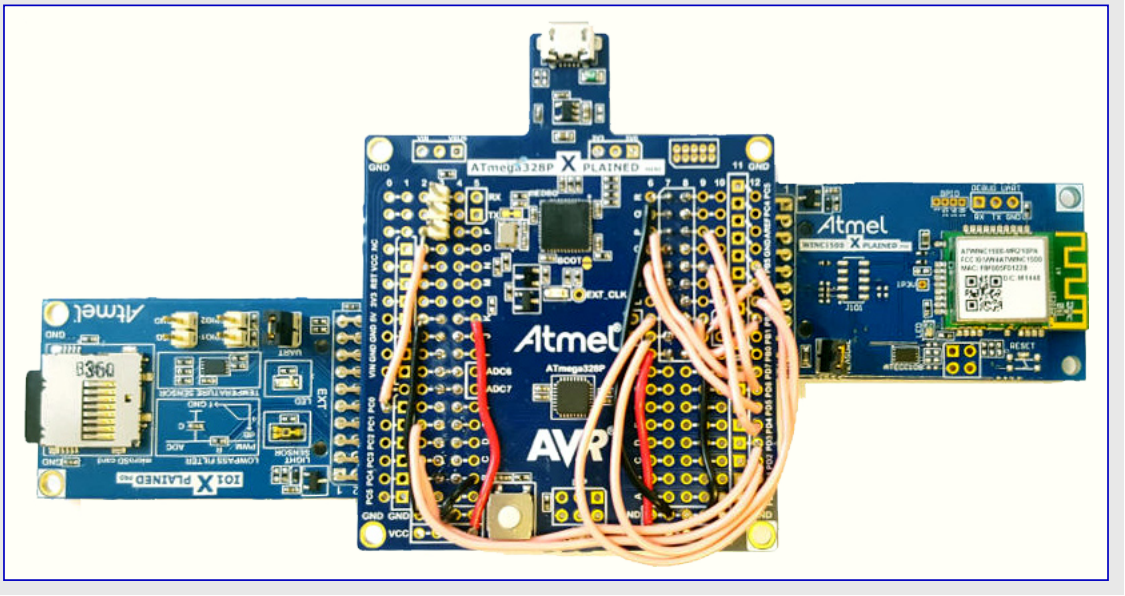
Atmel SmartConnect-WINC1500はIEEE® 802.11 b/g/n網制御器SoCです。これは既存MCU解決策にUART、I^2C、SPIとWiFiｲﾝﾀｰﾌｪｰｽを通してWiFiと網の能力をもたらす理想的な追加物です。ATWINC1500は最小資源必要条件を持つどれかのAtmel AVR®またはSMART MCUに接続します。ATWINC1500の最も高度な動作形態は最大72MHz物理層(PHY)単位処理量を提供する単一ｽﾄﾘｰﾑ 1×1 802.11n動作です。ATWINC1500は完全に統合された電力増幅器、LNA、ｽｲｯﾁ、電力管理が特徴です。ATWINC1500は内部ﾌﾗｯｼｭ ﾒﾓﾘ更にUART、SPI、I^2Cを含む多数の周辺機能ｲﾝﾀｰﾌｪｰｽを提供します。

2.2. ﾊｰﾄﾞｳｪｱ変更

ATWINC1500 Xplained ProとI/O1 Xplained ProをXplained Miniに接続するには、ﾊｰﾄﾞｳｪｱに対して行われるべきいくつかの変更が必要です。変更を行うには、Xplained Mini基板に半田付けするための2つのL型100milﾍｯﾀﾞと単ﾋﾟﾝ ﾍｯﾀﾞだけなく、線材と1つの0Ω抵抗も必要です。単ﾋﾟﾝ ﾍｯﾀﾞはﾋﾟﾝによって制御することができる何れかの外部部品に対して基板を接続するのに使うことができ、故に追加は任意選択です。

変更終了後、次図で示されるように、I/O1 Xplained Proを左側拡張ﾍｯﾀﾞにATWINC1500 Xplained Proを右側拡張ﾍｯﾀﾞに接続することができます。

図2-1. ハードウェア変更を持つXplained Mini基板

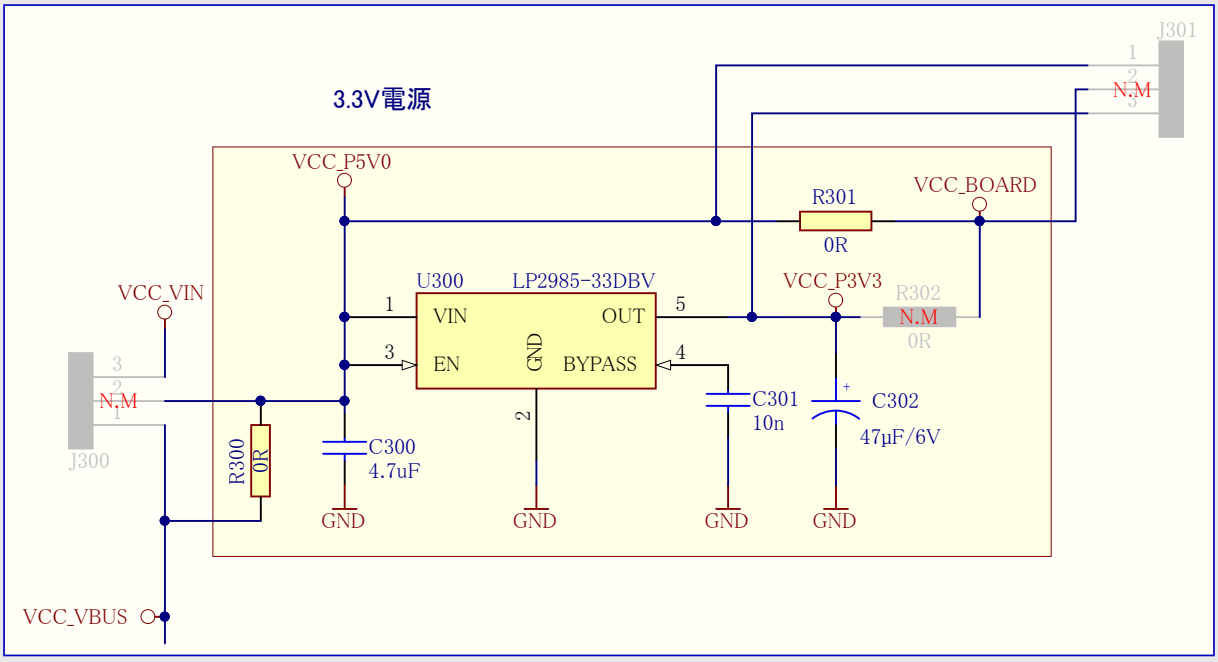


2.2.1. 電源

ATWINC1500は3.0～4.2Vの動作電圧を持ちます。Xplained Mini基板が5Vの供給電圧を持つため、ATWINC1500 Xplained Proを基板に接続する前に電圧がより適切な3.3Vに変更されることが重要です。

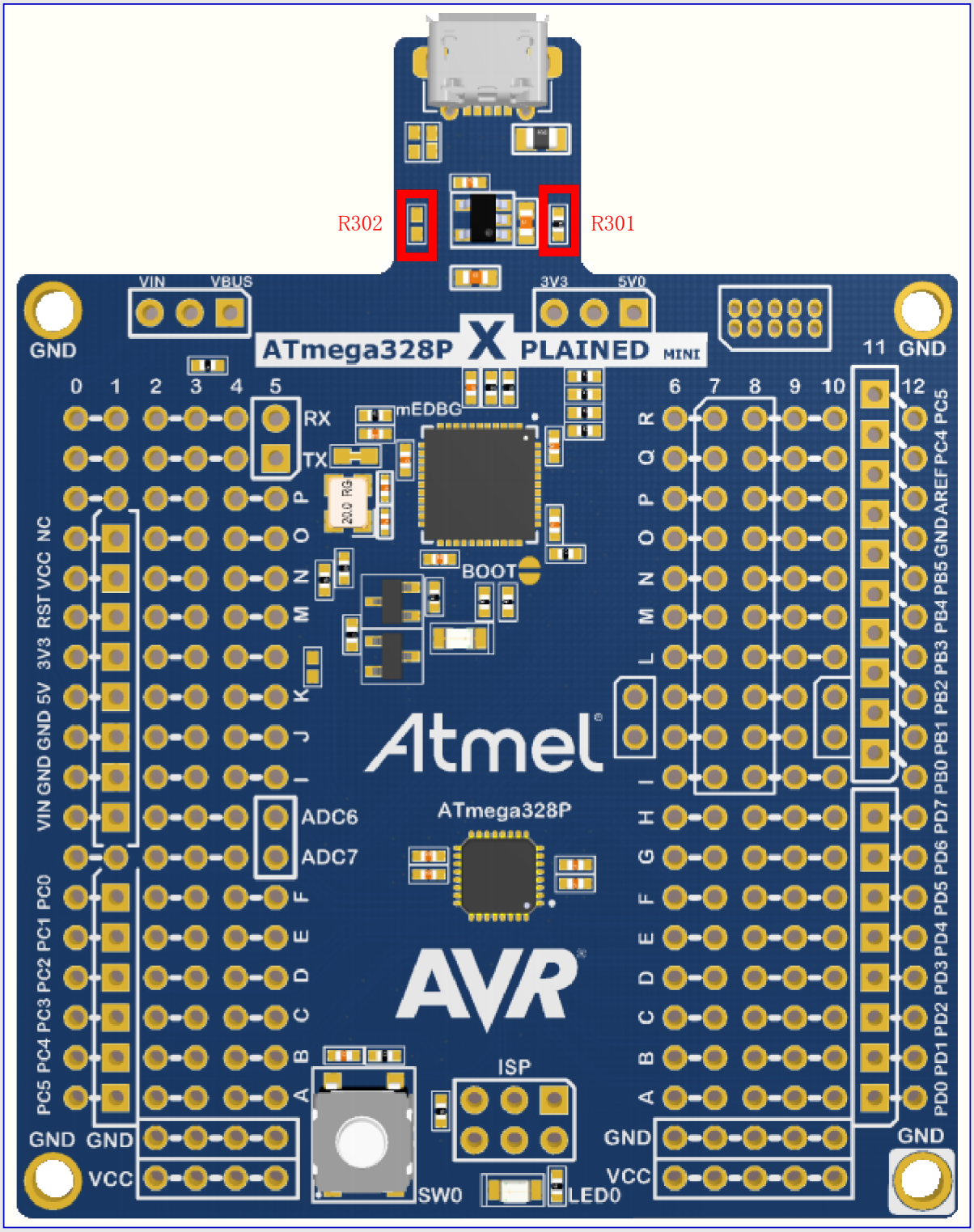
下図は3.3V電源回路図を示します。

図2-2. Xplained Mini 3.3V電源



Xpained Miniの電源を5Vから3.3Vに変更するには、R301の0Ω抵抗を取り外してR302の0Ω抵抗を追加してください。これらの抵抗は基板上のUSBコネクタ傍らに置かれ、下図で記されます。

図2-3. 電源抵抗器を持つXplained Mini基板

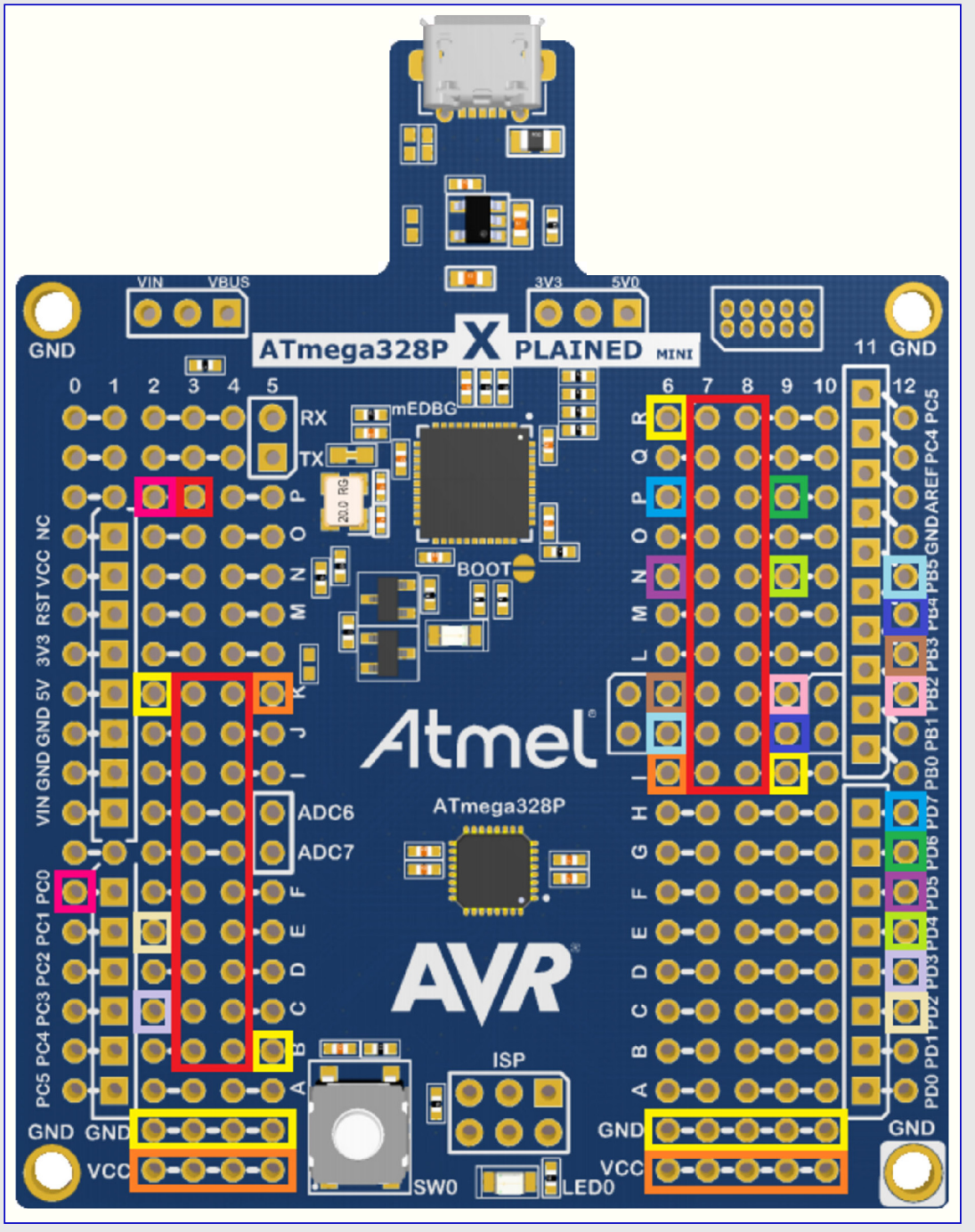


2.2.2. 拡張ﾍｯﾀﾞ

Xplained Mini基板に半田付けするのに2つの拡張ﾍｯﾀﾞが必要です。これは10×2のL型100milﾍｯﾀﾞであるべきで、それらは基板の裏側に半田付けされなければなりません。任意選択の単ﾋﾟﾝ ﾍｯﾀﾞは基板の表側に半田付けされるべきです。

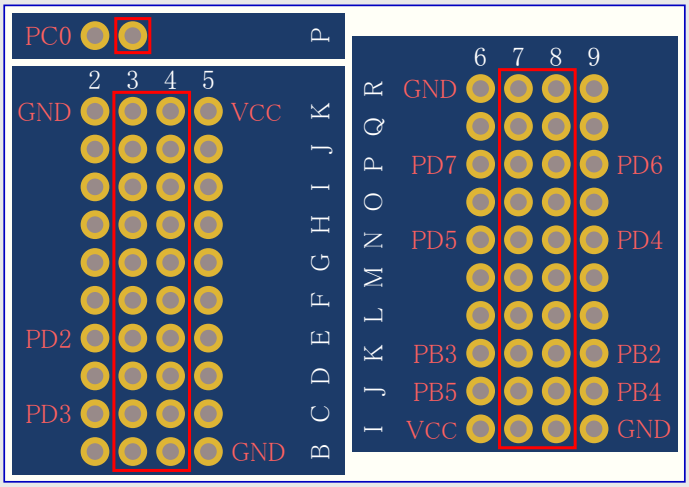
下図では行われるべき必要な接続が色分けされます。赤印はﾍｯﾀﾞが置かれるべき場所を示します。接続はﾋﾟﾝ ﾍｯﾀﾞの隣の穴から基板の側で対応する色で記された穴へ行われるべきです。これらは黄色と橙色で記されたGNDとVCCへのいくつかの接続です。これらの接続はﾋﾟﾝ ﾍｯﾀﾞの隣の穴から基板の裏側で記された何れかのGNDとVCCへ行くべきです。

図2-4. 接続を持つXplained Mini基板



右図ではどのﾋﾟﾝがどのﾋﾟﾝ ﾍｯﾀﾞに配線されるべきかがﾋﾟﾝの名称で記されます。例え拡張ﾍｯﾀﾞが基板の裏側に半田付けされるとは言え、図面は表側からなことに注意してください。半田付け処理を容易にするため、基板の表側に線を半田付けすることと、拡張ﾍｯﾀﾞの前に線を半田付けすることが推奨されます。

図2-5. ﾋﾟﾝ接続



2.3. ｿﾌﾄｳｪｱ

この実演応用で使われるｿﾌﾄｳｪｱは以下の項で記述される3つの部分から成ります。

- magaAVR応用
- ATWINC1500ﾌｧｰﾑｳｪｱ
- Android応用

応用の全般的な説明とそれの使用方法はこの後で与えられます。

2.3.1. megaAVR

Atmel Studioﾌﾟﾛｼﾞｪｸﾄはこの応用記述に付随するzipﾌｧｲﾙ内のStudioﾌﾟﾛｼﾞｪｸﾄ ﾌｫﾙﾀﾞで見つけることができ、それはWINC_megaと呼ばれます。ﾌﾟﾛｼﾞｪｸﾄ ﾌﾟﾛﾊﾟﾃｨ(Alt+F7)のﾃﾞﾊﾞｲｽ(Device)ﾀﾌﾞでﾃﾞﾊﾞｲｽを変更することにより、どのmegaAVR Xplained Mini基板に対しても再ｺﾝﾊﾟｲﾙすることができます。

megaAVRはSPI経由でATWINC1500単位部と通信し、UDPｶｹｯﾄを送る時の時間にﾀｲﾏ/ｶｳﾝﾀ1を使います。

Xplained Mini基板は基板上組み込みﾃﾞﾊﾞｯｶﾞとAtmel Studioでﾌﾟﾛｸﾞﾗﾐﾝｸﾞとﾃﾞﾊﾞｯｸﾞを行うことができます。Studio Project¥programmableﾌｫﾙﾀﾞで見つかるWINC_mega_328p.hexﾌｧｲﾙはATmegha328用にｺﾝﾊﾟｲﾙされていて、ATmega328ﾃﾞﾊﾞｲｽを使う場合にISP経由で直接書き込むことができます。

ﾋｭｰｽﾞ設定は次の様に設定されなければなりません。

- BOOTSZ = 256W_3F00
- BOOTRST = ☐
- RSTDISBL = ☐
- DWEN = ☐
- SPIEN = ☑
- WDTON = ☐
- EESAVE = ☐
- BODLEVEL = DISABLE
- CKDIV8 = ☐
- CKOUT = ☐
- SUT_CKSEL = EXTCLK_6CK_14CK_65MS

2.3.2. ATWINC1500

ATWINC1500ﾌｧｰﾑｳｪｱは実時間ｵﾍﾟﾚｰﾃｨﾝｸﾞ ｼｽﾃﾑ(RTOS)、軽量網階層、ﾐﾄﾞﾙｳｪｱ(中間ｿﾌﾄｳｪｱ)ｻｰﾋﾞｽ、MAC制御ﾄﾞﾗｲﾊﾞを含む部品の組の最上部で走行するﾎｽﾄｲﾝﾀｰﾌｪｰｽ層を含みます。この応用に対して支援されるATWINC1500ﾌｧｰﾑｳｪｱ版は19.3です。

ATWICN1500はﾋﾟｱtoﾋﾟｱ接続、LAN通信とｸﾗｳﾄﾞ ｻｰﾋﾞｽを持つ通信から広範囲の接続機能にｱｸｾｽすることを限られたﾒﾓﾘとCPU能力を持つﾎｽﾄMCU応用に許す直列ｲﾝﾀｰﾌｪｰｽを用いてﾎｽﾄMCUをｲﾝﾀｰﾌｪｰｽすることができます。

ATWICN1500ﾌｧｰﾑｳｪｱはSPIｲﾝﾀｰﾌｪｰｽ経由でﾎｽﾄ応用と2進ﾒｯｾｰｼﾞを交換します。このﾌｧｰﾑｳｪｱはﾃﾞﾊﾞｲｽ特有初期化、WLAN構成設定(走査、接続、切断)、BSD様のｿｹｯﾄ ｲﾝﾀｰﾌｪｰｽ(socket(),close(),bind(),listen(),accept(),send(),sendto(),recv(),recvfrom())を用いる網通行のようないくつかの型のﾎｽﾄ ｲﾝﾀｰﾌｪｰｽ(HIF)ﾒｯｾｰｼﾞをｶﾌﾟｾﾙ化します。

ATWINC1500のHIFはﾃﾞｰﾀと事象が先着(先着方式(FCFS))でATWICN1500ﾌｧｰﾑｳｪｱからﾎｽﾄMCUに押し出される、”押し出し”構造を使います。ATWINC1500ﾌｧｰﾑｳｪｱは1つ以上の事象が起こると必ず、専用GPIO線を用いてﾎｽﾄMCUに割り込みます。上の構造は以下の構成設定での走行を充分柔軟にすることをATWINC1500ﾌｧｰﾑｳｪｱに許します。

- **オペレーティング システムなし構成設定** – ホストMCU応用主繰り返しは保留中の事象を処理して呼び戻しを行うために特定関数を呼ぶことによってWINCホスト ドライバ状態機構を更新する責任があります。
- **オペレーティング システム構成設定** – 保留中の事象を処理して呼び戻しを行うため、特定関数を呼ぶことによって割り込み処理部から延期した作業を処理するのに専用タスクを実装することができます。

2.3.2.1. Wi-Fiソフトウェア API

ATWINC1500 Wi-Fi単位部はホストMCUに対するWi-Fiソフトウェア APIと共にやって来ます。このAPIの目的はホストMCUとATWINC1500 Wi-Fi単位部間で使われる2進規約の抽象化と同時に、どの使用者応用にも無線能力を追加するための容易で信頼に足る解決策の維持を提供することです。

ホストMCU応用はATWINC1500単位部から受信したデータと事象を処理する状態機構に基づくべきです。ATWINC1500ホスト ドライバは応用コードが属すホストMCUで走行するために以下のソフトウェア部品が必要とされます。

- **基板支援一括(BSP:Board Support Package)** – ホストMCU直列SPIバス ドライバとATWINC1500単位部用割り込み論理回路
- **ATWINC1500制御器ドライバ** – HIFメッセージ経由でATWINC1500チップとの通信を扱います。ホスト応用に対するC APIインターフェースも提供します。

2.3.2.2. 移植

変更や新しいファイルの追加とSPIバスの実装により、ATWINC1500 APIはどのMCUでも使うことができます。ATmega328PでATWINC1500を使うには以下のファイルが変更または追加されています。

- winc¥bsp¥include¥nm_bsp.h – 新しいデバイスを追加するのに更新され、特定デバイス用nm_bsp_*ヘッダ ファイルをインクルードしなければなりません。
- winc¥bsp¥include¥nm_bsp_mega.h – デバイス特有構成設定を追加されなければなりません。
- winc¥bsp¥source¥nm_bsp_mega.c – これはATWINC1500をリセットと許可に使われる出力ピンだけでなく、処理されることが必要なMCUへの合図のためにATWINC1500によって使われる割り込み入力ピンの構成設定も含みます。
- winc¥bus_wrapper¥source¥nm_bsp_wrapper_mega.c – nm_bsp_wrapper.hからの関数宣言を実装するために追加されなければなりません。これはATWINC1500と通信するためのSPIバスを初期化して使うための関数を含みます。
- config¥conf_winc.h – デバイスとハードウェア特有の構成設定を追加されなければなりません。

2.3.2.3. ファームウェア更新

ATWINC1500単位部は応用を動かすためにファームウェア版19.3に更新されなければなりません。

ファームウェア更新処理は最初にXplained Mini基板上のmegaAVRにシリアル橋渡し応用の書き込みから成ります。これはその後に仮想COMポートとATWINC1500単位部間のインターフェースとして働き、コンピュータからATWINC1500へファームウェアを転送します。

これを行うには、Xplained MiniがUSBケーブルでコンピュータに接続されていることと、ATWINC1500が基板の右側のピン ヘッダに接続されていることのを確実にしてください。この応用記述に付随する.zipファイル内のWINC1500_FW_UPDATEフォルダで見つかる.batスクリプトを走らせることにより、シリアル橋渡しがmegaAVRに書き込まれ、新しいファームウェアがATWINC1500へ転送されます。ATmega328Pに対して走らせるスクリプトは”mega328P_xplained_mini_firmware_update.bat”です。他のXplained基板に対する更新スクリプトもこのフォルダに含まれます。

障害対策

ファームウェア更新処理中に何れかの問題がある場合、以下の点を調べてください。

- megaAVRでデバッグWireが許可されている場合に更新処理が失敗することに注意してください。DWENヒューズが非プログラム(1)でSPIENヒューズがプログラム(0)されていることを確実にしてください。
- mEDBGJ仮想COMポートはPuTTyやTeraTermのような何れかの端末プログラムで開かれていなければなりません。
- mEDBGJ仮想COMポート ドライバがインストールされ、デバイス マネージャーに現れていることを確実にしてください。現れていなければ、Atmel Studioの拡張管理部経由で(Tools⇒Extension MamagerでAtmel USBインストーラを探して)ダウンロードしてください。

2.3.3. Android応用

IoT用AVRのAmdroid応用はこの応用記述に付随するzipファイル内のapkファイルとして見つかります。この応用をインストールするには、このapkファイルをAndroid機器に保存し、機器からそれを開いてください。これが応用をインストールします。インストールが妨げられる場合、設定⇒施錠画面と保護へ行って”未知の供給物”をONに切り替えてください。

応用ロゴが右図で示されます。

図2-6. IoT用AVR応用ロゴ

2.3.4. プログラムの流れ

初回に応用を走らせる時に、ATWINC1500は無線網に接続されることが必要です。ATWINC1500に利用可能なWi-FiのSSIDとパスワードを提供するため、準備動作に置かれなければなりません。これは電源投入時に基板上の押し釦を押さえ付けて置くことによって行われます。I/O1 Xplained上のLEDが点灯すると、釦を放すことができます。今やATWINC1500は準備動作に置かれ、AVR_for_IoTと呼ばれるそれ自身のアクセス ポイントを作成します。

準備動作はATWINC1500の組み込み機能で、ソフトウェアでの関数呼び出しで開始することができます。この動作では、ATWINC1500が網に資格情報を入力するのに使うことができる、ウェブ ページ表示の単独目的を持つアクセス ポイントとして働きます。応用はATWINC1500が資格情報を受信した時に呼び戻しを得て、これはその後に網への接続に使うことができます。

利用可能な網の使用者名とパスワードを供給するには2つの任意選択があります。最初のものはAVR_for_IoTに接続してウェブ ブラウザを開くのに電話機やコンピュータを使うことで、これは右図で示されるような準備ページを開きます。

網のSSIDとパスワードを入力して"Conect(接続)"釦を押してください。これはATWINC1500に資格情報を送ります。Device Name(装置名)領域が未使用なことに注意してください。ATWICN1500ファームウェアはアクセス ポイントを禁止して応用にSSIDとパスワードについての情報を送ります。アクセス ポイントが禁止されるため、電話機またはコンピュータはWi-Fiから切断され、異常がOFFに切り替えされます。応用はその後に与えられたWi-Fi網に接続を試みます。

図2-7. ATWINC1500準備ページ

AVR_for_IoTアクセス ポイントへの手動接続に代わる、他の任意選択はAndroid応用を使うことです。Androidアプリ(AVR for IoT)を開き、右図で示されるように"SEND WIFI CREDENTIALS TO WINC"釦を押してください。これを行う時に、電話機はATWINC1500が接続されるべき標準Wi-Fi網に接続されるべきです。

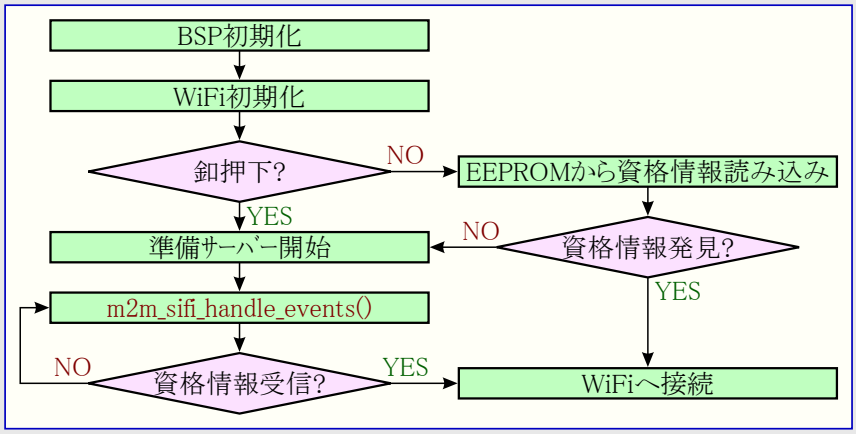
これはATWINC1500が接続すべき網のSSIDとパスワードをあなたに書かせるでしょう。再び"SEND WIFI CREDENTIALS TO WINC"釦が押されると、Android電話機は現在接続されていてAVR_for_IoTアクセス ポイントに接続する何れかのWi-Fiから接続され、ATWINC1500へ資格情報を送ります。これが行われると、以前に接続された網に再接続されるでしょう。

Wi-Fi資格情報が受信されてしまうと、MCUが電源ONにされる毎に準備処理を繰り返すのを不要にするため、MCUはそれをEEPROMに格納します。電源ON中にXplained Mini上の押し釦が押されなければ、SSIDとパスワードは存在するならばEEPROMから読まれ、応用はそれらの資格情報で直接的に接続を試みます。

下図は初期化と準備処理の流れ図を示します。

図2-8. Atmel交互切り替え実演主画面

図2-9. 初期化と準備処理の流れ図



応用がｱｸｾｽ ﾎﾟｲﾝﾄに接続されてIPｱﾄﾞﾚｽを得てしまうと、計時器を構成設定して4秒毎にUDPﾊﾟｹｯﾄを一斉送信するのにこれを使います。これらのﾊﾟｹｯﾄが網上で一斉送信されるため、同じｱｸｾｽ ﾎﾟｲﾝﾄに接続されるどのAndroidやｺﾝﾋﾟｭｰﾀもこれらのﾊﾟｹｯﾄを受信します。Android応用にXplained Miniからの UDPﾊﾟｹｯﾄを聴取させるには”SCAN FOR AVR FOR IOT DEVICES”釦を押してください。

これは概ね6秒間網で走査を始めて、その後にAndroid電話機として同じ無線網に接続されたATWINC1500装置を持つ何れかの近所のXplained MIniを示します利用可能などの装置のIPｱﾄﾞﾚｽも示され、これが押されると、I/O1 Xplained上のLEDが交互(ON/OFF)切り替えします。

Xplained Mini基板の左側のﾋﾟﾝ ﾍｯﾀﾞも交互切り替えし、ﾋﾟﾝによって制御することができる何か外部に接続される実演を許します。

下図は応用の主繰り返しの流れ図を示します。

図2-10. Atmel交互切替実演走査終了

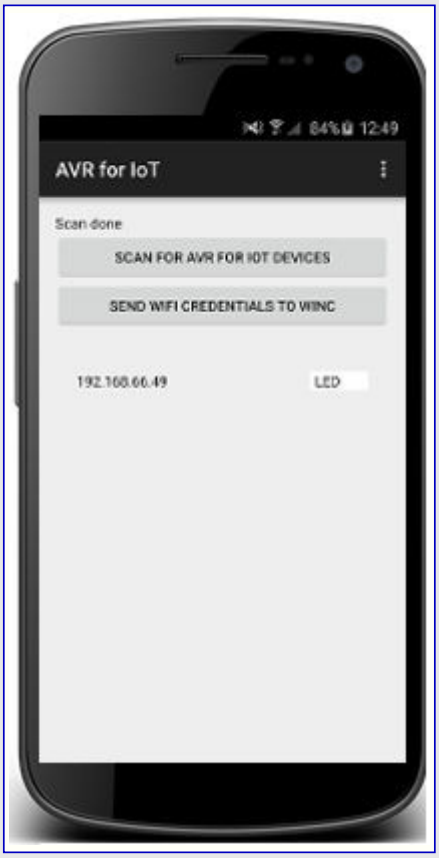
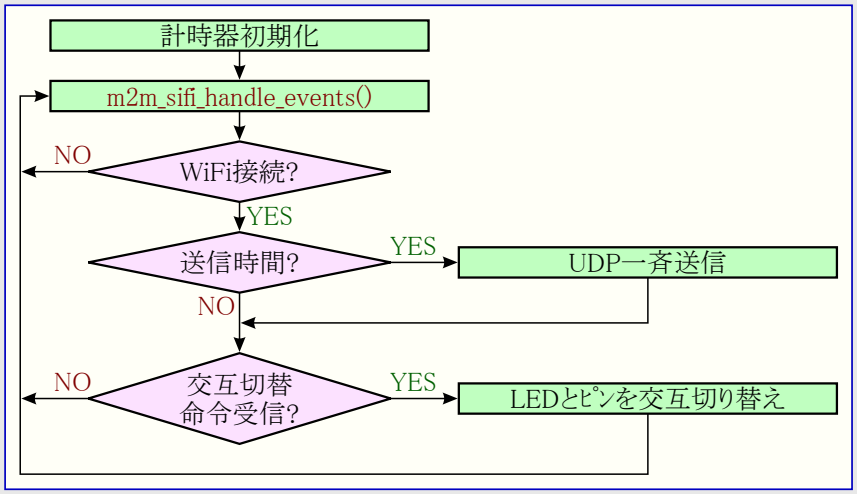


図2-11. 応用主繰り返し期化と準備処理の流れ図



2.3.5. 規約

4秒毎にmegaAVRで走行する応用は6789ﾎﾟｰﾄで一斉送信ｱﾄﾞﾚｽの255.255.255.255にUDPﾊﾟｹｯﾄを送出します。このﾊﾟｹｯﾄはAndroid応用によっても既知の予め定義された緩衝部から成ります。

```
/** 鍵緩衝部 - Android応用と一致しなければなりません。 */
uint8_t key_buf[] = KEY_BUFFER;
```

これはAndroid応用に実演応用からのUDP一斉送信ﾊﾟｹｯﾄと網上の何れかの他の一斉送信を区別できるようにさせます。鍵緩衝部に一致する一斉送信は走査後にAndroid応用表示で利用可能装置の一覧に追加されます。

megaAVRはWi-Fiに接続されると直ぐにやって来る要求を聴取するためにTCPｿｹｯﾄを開きます。Android応用で装置のIPｱﾄﾞﾚｽが押されると、1234ﾎﾟｰﾄ上のTCPｿｹｯﾄ経由でXplained Miniに接続し、語”toggle”を含むﾒｯｾｰｼﾞを送り、これはmegaAVRがTCPｿｹｯﾄでﾒｯｾｰｼﾞを受信する時に探すｷｰﾜｰﾄﾞです。megaAVRによってこれが受信されると、LEDを交互切り替えしてﾋﾟﾝを交互切り替えします。これが終わると、確認ﾒｯｾｰｼﾞをAndoroid応用に送り返します。このﾒｯｾｰｼﾞが受信されない、または送信中に何れかの他の異常が起きた場合、Android応用は何かが間違っていたことを示す赤箱でIPｱﾄﾞﾚｽを強調表示します。

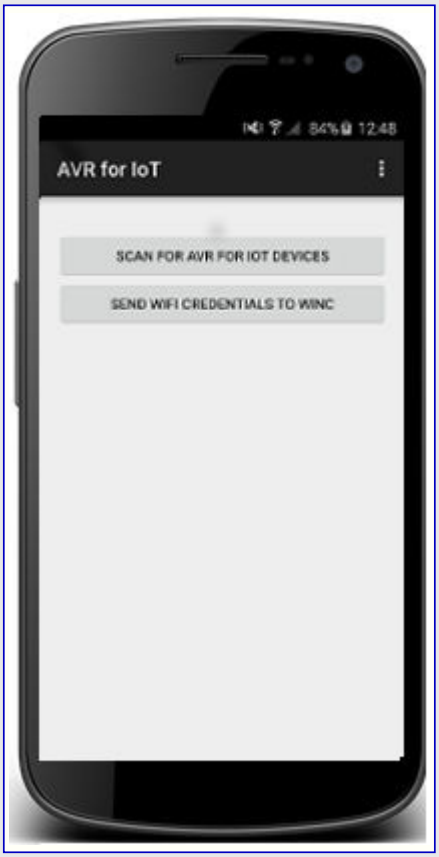
3. 応用の更なる開発

以下の項はAndroid応用に温度ﾃﾞｰﾀも送るのに応用が更にどう拡張され得るかを記述します。Android応用は既にこれ用に準備され、応用から許可することができます。これは右図で示されるように主画面の上の右角で3つの点を押すことによって行われ、settingsを選んで”Enable temperature reading”ﾁｪｯｸ枠をﾁｪｯｸします。

これは一覧内の各項目に別の釦を追加し、これはXplained Mini基板からの温度ﾃﾞｰﾀを要求するためにｸﾘｯｸすることができます。この釦が押されると、Android応用は”temp”の文字でATWINC1500にTCPﾊﾟｹｯﾄを送ります。このﾒｯｾｰｼﾞの主織はAtmel Studioﾌﾟﾛｼﾞｪｸﾄ内のWINC_mega.cﾌｧｲﾙで実装されることが必要です。”WINC_mega6_2.atsln”Atmel Studioﾌﾟﾛｼﾞｪｸﾄは応用記述と共に来る.zipﾌｧｲﾙ内の”Studio Project¥WINC_mega”ﾌｫﾙﾀﾞで見つけることができます。

行うべき最初の事はmegaAVRの内部温度感知器を読むためにADCを初期化するための1つとADC変換を行うための1つで、2つの関数をWINC_mega.cに追加することです。

図3-1. Android応用で温度読み取り許可




```
static void init_temp(void)
{
    sysclk_enable_module(POWER_RED_REG0, PRADC_bm);
    /* 内部温度感知器から読むためにADCを初期化 */
    ADMUX =  (1 << MUX3) | (1 << REFS0) | (1 << REFS1);
    /* ADC許可 */
    ADCSRA = 1 << ADEN | 1 << ADPS0 | 1 << ADPS1 | 1 << ADPS2;
}
```

```
static void read_temp(uint8_t *buffer)
{
    /* 変換開始 */
    ADCSRA |= 1 << ADSC;

    /* 変換終了待ち */
    while(!(ADCSRA & (1 << ADIF)));
    /* 変換結果を緩衝部に書き込み */
    ((uint16_t*)buffer)[0] = ADCW;
}
```

Android応用から温度を送るための要求が受信されたか否かに関わらず、主繰り返しに合図するための変数もまた必要です。この変数は全体変数で主関数の外側に置かれるべきです。

```
/** 温度読み取り要求が受信されたかを知るための主繰り返し用変数 */
volatile bool temperature = false;
```

主関数の始めで緩衝部が宣言されるべきです。これは要求された時にAndroid応用に送られるべき温度を格納するのに使われます。

```
/** 温度を格納するための緩衝部 */
uint8_t temp_buf[2];
```

m2m_tcp_socket_handler()ｿｹｯﾄ呼び戻し関数に於いて、Android応用からの新しいﾒｯｾｰｼﾞを処理する論理を追加します。前に述べられたように、ﾒｯｾｰｼﾞ送出は”temp”で、故にこの文字列が受信したﾒｯｾｰｼﾞに存在するかを調べることが必要です。そうならば、それが主繰り返し内で処理されるように、全体変数を真に設定します。switch case内でcase SOCKET_MSG_RECVを見つけて以下のｺｰﾄﾞで置き換えてください。

```
/* Message receive */
case SOCKET_MSG_RECV:
    /* Check command */
    if (strstr(gau8SocketTestBuffer, "toggle")) {
        toggle = true;
    } else if (strstr(gau8SocketTestBuffer, "temp")) {
        temperature = true;
    }
    break;
```

これを行った後、主関数からADC初期化関数を呼ぶことが必要で、これはwhile(1)繰り返しの先頭の前で行われなければなりません。

```
init_temp();
```

最後に行うべき事は温度読み取りを追加してそれが要求された時にそれをAndroid応用へ送り返すことです。以下のｺｰﾄﾞがLED交互切り替えに対する同様ｺｰﾄﾞ直後のif (wifi_connected == M2M_WIFI_CONNECTED)行内側のwhile(1)の内側に置かれるべきです。

```
/* 温度命令受信  */
if (temperature) {
    temperature = false;
    read_temp(temp_buf);
    /* 読み取り温度送り返し */
    send(tcp_client_socket, (void*)temp_buf, sizeof(temp_buf), 0);
}
```

AVRからAndroid応用へ送る値は温度感知部ADC採取からの生の値です。故にこれは℃での温度ではありませんが、AVRを温めたり冷やしたりする時に読み取りでの違いがあります。温度感知器についてのより多くの情報に関してはﾃﾞﾊﾞｲｽのﾃﾞｰﾀｼｰﾄを参照してください。

追加されるこれら全ての実装を持つWINC_mega.cﾌｧｲﾙは.zipﾌｧｲﾙ内の”Studio Project¥temperature”ﾌｫﾙﾀﾞで見つけることができ、参考として使うことができます。

4. 改訂履歴

文書改訂	日付	注釈
42552A	2015年9月	初版文書公開

Atmel | Enabling Unlimited Possibilities®

Atmel Corporation 1600 Technology Drive, San Jose, CA 95110 USA TEL:(+1)(408) 441-0311 FAX: (+1)(408) 436-4200 | www.atmel.com

---

本応用記述はAtmelのAT13041応用記述(Rev.42552A-09/2015)の翻訳日本語版です。日本語では不自然となる重複する形容表現は省略されている場合があります。日本語では難解となる表現は大幅に意訳されている部分もあります。必要に応じて一部加筆されています。頁割の変更により、原本より頁数が少なくなっています。

必要と思われる部分には( )内に英語表記や略称などを残す形で表記しています。

青字の部分はリンクとなっています。一般的に赤字の0,1は論理0,1を表します。その他の赤字は重要な部分を表します。